



MapReduce Service

Component Operation Guide

Date 2022-01-06

Contents

1 Using Alluxio.....	1
1.1 Configuring an Underlying Storage System.....	1
1.2 Accessing Alluxio Using a Data Application.....	2
1.3 Common Operations of Alluxio.....	5
2 Using CarbonData (for Versions Earlier Than MRS 3.x).....	8
2.1 Getting Started with CarbonData.....	8
2.2 About CarbonData Table.....	10
2.3 Creating a CarbonData Table.....	11
2.4 Deleting a CarbonData Table.....	13
3 Using CarbonData (for MRS 3.x or Later).....	15
3.1 Overview.....	15
3.1.1 CarbonData Overview.....	15
3.1.2 Main Specifications of CarbonData.....	18
3.2 Configuration Reference.....	20
3.3 CarbonData Operation Guide.....	32
3.3.1 CarbonData Quick Start.....	32
3.3.2 CarbonData Table Management.....	35
3.3.2.1 About CarbonData Table.....	35
3.3.2.2 Creating a CarbonData Table.....	37
3.3.2.3 Deleting a CarbonData Table.....	39
3.3.2.4 Modify the CarbonData Table.....	39
3.3.3 CarbonData Table Data Management.....	40
3.3.3.1 Loading Data.....	40
3.3.3.2 Deleting Segments.....	40
3.3.3.3 Combining Segments.....	42
3.3.4 CarbonData Data Migration.....	45
3.3.5 Migrating Data on CarbonData from Spark 1.5 to Spark2x.....	47
3.4 CarbonData Performance Tuning.....	48
3.4.1 Tuning Guidelines.....	49
3.4.2 Suggestions for Creating CarbonData Tables.....	51
3.4.3 Configurations for Performance Tuning.....	53
3.5 CarbonData Access Control.....	56

3.6 CarbonData Syntax Reference.....	58
3.6.1 DDL.....	58
3.6.1.1 CREATE TABLE.....	58
3.6.1.2 CREATE TABLE As SELECT.....	61
3.6.1.3 DROP TABLE.....	62
3.6.1.4 SHOW TABLES.....	63
3.6.1.5 ALTER TABLE COMPACTION.....	64
3.6.1.6 TABLE RENAME.....	65
3.6.1.7 ADD COLUMNS.....	66
3.6.1.8 DROP COLUMNS.....	67
3.6.1.9 CHANGE DATA TYPE.....	68
3.6.1.10 REFRESH TABLE.....	69
3.6.1.11 REGISTER INDEX TABLE.....	70
3.6.1.12 REFRESH INDEX.....	71
3.6.2 DML.....	72
3.6.2.1 LOAD DATA.....	72
3.6.2.2 UPDATE CARBON TABLE.....	77
3.6.2.3 DELETE RECORDS from CARBON TABLE.....	78
3.6.2.4 INSERT INTO CARBON TABLE.....	80
3.6.2.5 DELETE SEGMENT by ID.....	81
3.6.2.6 DELETE SEGMENT by DATE.....	81
3.6.2.7 SHOW SEGMENTS.....	82
3.6.2.8 CREATE SECONDARY INDEX.....	83
3.6.2.9 SHOW SECONDARY INDEXES.....	84
3.6.2.10 DROP SECONDARY INDEX.....	85
3.6.2.11 CLEAN FILES.....	85
3.6.2.12 SET/RESET.....	86
3.6.3 Operation Concurrent Execution.....	89
3.6.4 API.....	93
3.6.5 Spatial Indexes.....	95
3.7 CarbonData Troubleshooting.....	109
3.7.1 Filter Result Is not Consistent with Hive when a Big Double Type Value Is Used in Filter.....	109
3.7.2 Query Performance Deterioration.....	109
3.8 CarbonData FAQ.....	110
3.8.1 Why Is Incorrect Output Displayed When I Perform Query with Filter on Decimal Data Type Values?.....	110
3.8.2 How to Avoid Minor Compaction for Historical Data?.....	111
3.8.3 How to Change the Default Group Name for CarbonData Data Loading?.....	111
3.8.4 Why Does INSERT INTO CARBON TABLE Command Fail?.....	112
3.8.5 Why Is the Data Logged in Bad Records Different from the Original Input Data with Escape Characters?.....	112
3.8.6 Why Data Load Performance Decreases due to Bad Records?.....	113

3.8.7 Why INSERT INTO/LOAD DATA Task Distribution Is Incorrect and the Opened Tasks Are Less Than the Available Executors when the Number of Initial Executors Is Zero?.....	113
3.8.8 Why Does CarbonData Require Additional Executors Even Though the Parallelism Is Greater Than the Number of Blocks to Be Processed?.....	114
3.8.9 Why Data loading Fails During off heap?.....	114
3.8.10 Why Do I Fail to Create a Hive Table?.....	115
3.8.11 Why CarbonData tables created in V100R002C50RC1 not reflecting the privileges provided in Hive Privileges for non-owner?.....	115
3.8.12 How Do I Logically Split Data Across Different Namespaces?.....	116
3.8.13 Why Missing Privileges Exception is Reported When I Perform Drop Operation on Databases?....	117
3.8.14 Why the UPDATE Command Cannot Be Executed in Spark Shell?.....	117
3.8.15 How Do I Configure Unsafe Memory in CarbonData?.....	118
3.8.16 Why Exception Occurs in CarbonData When Disk Space Quota is Set for Storage Directory in HDFS?.....	118
3.8.17 Why Does Data Query or Loading Fail and "org.apache.carbondata.core.memory.MemoryException: Not enough memory" Is Displayed?.....	119
4 Using ClickHouse.....	120
4.1 Using ClickHouse from Scratch.....	120
4.2 ClickHouse Cluster Configuration.....	126
4.3 Setting the ClickHouse Username and Password.....	129
4.4 ClickHouse Table Engine Overview.....	131
4.5 Creating and Using ClickHouse Replicated Tables and Distributed Tables.....	138
4.6 ClickHouse Log Overview.....	143
5 Using DBService.....	146
5.1 DBService Log Overview.....	146
6 Using Flink.....	150
6.1 Using Flink from Scratch.....	150
6.2 Viewing Flink Job Information.....	157
6.3 Flink Configuration Management.....	157
6.3.1 Configuring Parameter Paths.....	157
6.3.2 JobManager & TaskManager.....	158
6.3.3 Blob.....	170
6.3.4 Distributed Coordination (via Akka).....	172
6.3.5 SSL.....	178
6.3.6 Network communication (via Netty).....	182
6.3.7 JobManager Web Frontend.....	183
6.3.8 File Systems.....	188
6.3.9 State Backend.....	189
6.3.10 Kerberos-based Security.....	191
6.3.11 HA.....	193
6.3.12 Environment.....	196
6.3.13 Yarn.....	197
6.3.14 Pipeline.....	199

6.4 Security Configuration.....	200
6.4.1 Security Features.....	200
6.4.2 Configuring Kafka.....	201
6.4.3 Configuring Pipeline.....	202
6.5 Security Hardening.....	203
6.5.1 Authentication and Encryption.....	203
6.5.2 ACL Control.....	211
6.5.3 Web Security.....	211
6.6 Security Statement.....	214
6.7 Flink Log Overview.....	214
6.8 Flink Performance Tuning.....	216
6.8.1 Optimization DataStream.....	216
6.8.1.1 Memory Configuration Optimization.....	217
6.8.1.2 Configuring DOP.....	217
6.8.1.3 Configuring Process Parameters.....	218
6.8.1.4 Optimizing the Design of Partitioning Method.....	219
6.8.1.5 Configuring the Netty Network Communication.....	220
6.8.1.6 Summarization.....	221
6.9 Common Flink Shell Commands.....	221
7 Using Flume.....	227
7.1 Using Flume from Scratch.....	227
7.2 Overview.....	233
7.3 Installing the Flume Client.....	236
7.3.1 Installing the Flume Client on Clusters of Versions Earlier Than MRS 3.x.....	236
7.3.2 Installing the Flume Client on Clusters of MRS 3.x or a Later Version.....	239
7.4 Viewing Flume Client Logs.....	241
7.5 Stopping or Uninstalling the Flume Client.....	242
7.6 Using the Encryption Tool of the Flume Client.....	243
7.7 Flume Service Configuration Guide.....	244
7.8 Flume Configuration Parameter Description.....	277
7.9 Using Environment Variables in the properties.properties File.....	292
7.10 Non-Encrypted Transmission.....	293
7.10.1 Configuring Non-encrypted Transmission.....	294
7.10.2 Typical Scenario: Collecting Local Static Logs and Uploading Them to Kafka.....	296
7.10.3 Typical Scenario: Collecting Local Static Logs and Uploading Them to HDFS.....	303
7.10.4 Typical Scenario: Collecting Local Dynamic Logs and Uploading Them to HDFS.....	311
7.10.5 Typical Scenario: Collecting Logs from Kafka and Uploading Them to HDFS.....	319
7.10.6 Typical Scenario: Collecting Logs from Kafka and Uploading Them to HDFS Through the Flume Client.....	327
7.10.7 Typical Scenario: Collecting Local Static Logs and Uploading Them to HBase.....	331
7.11 Encrypted Transmission.....	340
7.11.1 Configuring the Encrypted Transmission.....	340

7.11.2 Typical Scenario: Collecting Local Static Logs and Uploading Them to HDFS.....	350
7.12 Viewing Flume Client Monitoring Information.....	365
7.13 Connecting Flume to Kafka in Security Mode.....	365
7.14 Connecting Flume with Hive in Security Mode.....	366
7.15 Configuring the Flume Service Model.....	369
7.15.1 Overview.....	369
7.15.2 Service Model Configuration Guide.....	369
7.16 Introduction to Flume Logs.....	375
7.17 Flume Client Cgroup Usage Guide.....	378
7.18 Secondary Development Guide for Flume Third-Party Plug-ins.....	379
7.19 Common Issues About Flume.....	380
8 Using HBase.....	382
8.1 Using HBase from Scratch.....	382
8.2 Using an HBase Client.....	387
8.3 Creating HBase Roles.....	389
8.4 Configuring HBase Replication.....	391
8.5 Configuring HBase Parameters.....	401
8.6 Enabling Cross-Cluster Copy.....	402
8.7 Using the ReplicationSyncUp Tool.....	403
8.8 Using HIndex.....	405
8.8.1 Introduction to HIndex.....	405
8.8.2 Loading Index Data in Batches.....	415
8.8.3 Using an Index Generation Tool.....	418
8.8.4 Migrating Index Data.....	420
8.9 Configuring HBase DR.....	423
8.10 Performing an HBase DR Service Switchover.....	432
8.11 Performing an HBase DR Active/Standby Cluster Switchover.....	434
8.12 Community BulkLoad Tool.....	436
8.13 Configuring the MOB.....	436
8.14 Configuring Secure HBase Replication.....	438
8.15 Configuring Region In Transition Recovery Chore Service.....	439
8.16 Using a Secondary Index.....	439
8.17 HBase Log Overview.....	441
8.18 HBase Performance Tuning.....	444
8.18.1 Improving the BulkLoad Efficiency.....	444
8.18.2 Improving Put Performance.....	445
8.18.3 Optimizing Put and Scan Performance.....	446
8.18.4 Improving Real-time Data Write Performance.....	450
8.18.5 Improving Real-time Data Read Performance.....	459
8.18.6 Optimizing JVM Parameters.....	466
8.19 Common Issues About HBase.....	467
8.19.1 Why Does a Client Keep Failing to Connect to a Server for a Long Time?.....	467

8.19.2 Operation Failures Occur in Stopping BulkLoad On the Client.....	469
8.19.3 Why May a Table Creation Exception Occur When HBase Deletes or Creates the Same Table Consecutively?.....	469
8.19.4 Why Other Services Become Unstable If HBase Sets up A Large Number of Connections over the Network Port?.....	470
8.19.5 Why Does the HBase BulkLoad Task (One Table Has 26 TB Data) Consisting of 210,000 Map Tasks and 10,000 Reduce Tasks Fail?.....	471
8.19.6 How Do I Restore a Region in the RIT State for a Long Time?.....	471
8.19.7 Why Does HMaster Exits Due to Timeout When Waiting for the Namespace Table to Go Online?	472
8.19.8 Why Does SocketTimeoutException Occur When a Client Queries HBase?.....	473
8.19.9 Why Modified and Deleted Data Can Still Be Queried by Using the Scan Command?.....	474
8.19.10 Why "java.lang.UnsatisfiedLinkError: Permission denied" exception thrown while starting HBase shell?.....	475
8.19.11 When does the RegionServers listed under "Dead Region Servers" on HMaster WebUI gets cleared?.....	475
8.19.12 Why Are Different Query Results Returned After I Use Same Query Criteria to Query Data Successfully Imported by HBase bulkload?.....	476
8.19.13 What Should I Do If I Fail to Create Tables Due to the FAILED_OPEN State of Regions?.....	476
8.19.14 How Do I Delete Residual Table Names in the /hbase/table-lock Directory of ZooKeeper?.....	477
8.19.15 Why Does HBase Become Faulty When I Set a Quota for the Directory Used by HBase in HDFS?	477
8.19.16 Why HMaster Times Out While Waiting for Namespace Table to be Assigned After Rebuilding Meta Using OfflineMetaRepair Tool and Startups Failed.....	479
8.19.17 Why Messages Containing FileNotFoundException and no lease Are Frequently Displayed in the HMaster Logs During the WAL Splitting Process?.....	479
8.19.18 Insufficient Rights When a Tenant Accesses Phoenix.....	481
8.19.19 What Can I Do When HBase Fails to Recover a Task and a Message Is Displayed Stating "Rollback recovery failed"?.....	481
8.19.20 How Do I Fix Region Overlapping?.....	482
8.19.21 Why Does RegionServer Fail to Be Started When GC Parameters Xms and Xmx of HBase RegionServer Are Set to 31 GB?.....	483
8.19.22 Why Does the LoadIncrementalHFiles Tool Fail to Be Executed and "Permission denied" Is Displayed When Nodes in a Cluster Are Used to Import Data in Batches?.....	483
8.19.23 Why Is the Error Message "import argparse" Displayed When the Phoenix sqlline Script Is Used?	485
8.19.24 How Do I Deal with the Restrictions of the Phoenix BulkLoad Tool?.....	485
8.19.25 Why a Message Is Displayed Indicating that the Permission is Insufficient When CTBase Connects to the Ranger Plug-ins?.....	486
9 Using HDFS.....	488
9.1 Using Hadoop from Scratch.....	488
9.2 Configuring Memory Management.....	490
9.3 Creating an HDFS Role.....	491
9.4 Using the HDFS Client.....	493
9.5 Running the DistCp Command.....	495
9.6 Overview of HDFS File System Directories.....	500

9.7 Changing the DataNode Storage Directory.....	508
9.8 Configuring HDFS Directory Permission.....	511
9.9 Configuring NFS.....	512
9.10 Planning HDFS Capacity.....	513
9.11 Configuring ulimit for HBase and HDFS.....	516
9.12 Balancing DataNode Capacity.....	517
9.13 Configuring Replica Replacement Policy for Heterogeneous Capacity Among DataNodes.....	522
9.14 Configuring the Number of Files in a Single HDFS Directory	523
9.15 Configuring the Recycle Bin Mechanism.....	524
9.16 Setting Permissions on Files and Directories.....	525
9.17 Setting the Maximum Lifetime and Renewal Interval of a Token.....	526
9.18 Configuring the Damaged Disk Volume.....	527
9.19 Configuring Encrypted Channels.....	527
9.20 Reducing the Probability of Abnormal Client Application Operation When the Network Is Not Stable	529
9.21 Configuring the NameNode Blacklist.....	529
9.22 Optimizing HDFS NameNode RPC QoS.....	532
9.23 Optimizing HDFS DataNode RPC QoS.....	535
9.24 Configuring Reserved Percentage of Disk Usage on DataNodes.....	535
9.25 Configuring HDFS NodeLabel.....	536
9.26 Configuring HDFS Mover.....	542
9.27 Using HDFS AZ Mover.....	544
9.28 Configuring HDFS DiskBalancer.....	545
9.29 Configuring HDFS EC Storage.....	548
9.30 Configuring the Observer NameNode to Process Read Requests.....	557
9.31 Performing Concurrent Operations on HDFS Files.....	558
9.32 Introduction to HDFS Logs.....	561
9.33 HDFS Performance Tuning.....	565
9.33.1 Improving Write Performance.....	565
9.33.2 Improving Read Performance Using Client Metadata Cache.....	566
9.33.3 Improving the Connection Between the Client and NameNode Using Current Active Cache.....	568
9.34 FAQ.....	569
9.34.1 NameNode Startup Is Slow.....	570
9.34.2 DataNode Is Normal but Cannot Report Data Blocks.....	570
9.34.3 HDFS WebUI Cannot Properly Update Information About Damaged Data.....	571
9.34.4 Why Does the Distcp Command Fail in the Secure Cluster, Causing an Exception?.....	572
9.34.5 Why Does DataNode Fail to Start When the Number of Disks Specified by dfs.datanode.data.dir Equals dfs.datanode.failed.volumes.tolerated?.....	572
9.34.6 Why Does an Error Occur During DataNode Capacity Calculation When Multiple data.dir Are Configured in a Partition?.....	573
9.34.7 Standby NameNode Fails to Be Restarted When the System Is Powered off During Metadata (Namespace) Storage.....	574
9.34.8 Why Data in the Buffer Is Lost If a Power Outage Occurs During Storage of Small Files.....	575

9.34.9 Why Does Array Border-crossing Occur During FileInputFormat Split?.....	575
9.34.10 Why Is the Storage Type of File Copies DISK When the Tiered Storage Policy Is LAZY_PERSIST?	576
9.34.11 The HDFS Client Is Unresponsive When the NameNode Is Overloaded for a Long Time.....	576
9.34.12 Can I Delete or Modify the Data Storage Directory in DataNode?.....	577
9.34.13 Blocks Miss on the NameNode UI After the Successful Rollback.....	578
9.34.14 Why Is "java.net.SocketException: No buffer space available" Reported When Data Is Written to HDFS.....	579
9.34.15 Why are There Two Standby NameNodes After the active NameNode Is Restarted?.....	580
9.34.16 When Does a Balance Process in HDFS, Shut Down and Fail to be Executed Again?.....	582
9.34.17 "This page can't be displayed" Is Displayed When Internet Explorer Fails to Access the Native HDFS UI.....	582
9.34.18 NameNode Fails to Be Restarted Due to EditLog Discontinuity.....	583
10 Using Hive.....	585
10.1 Using Hive from Scratch.....	585
10.2 Configuring Hive Parameters.....	589
10.3 Hive SQL.....	591
10.4 Permission Management.....	594
10.4.1 Hive Permission.....	594
10.4.2 Creating a Hive Role.....	598
10.4.3 Configuring Permissions for Hive Tables, Columns, or Databases.....	602
10.4.4 Configuring Permissions to Use Other Components for Hive.....	606
10.5 Using a Hive Client.....	610
10.6 Using HDFS Colocation to Store Hive Tables.....	614
10.7 Using the Hive Column Encryption Function.....	615
10.8 Customizing Row Separators.....	616
10.9 Deleting Single-Row Records from Hive on HBase.....	617
10.10 Configuring HTTPS/HTTP-based REST APIs.....	617
10.11 Enabling or Disabling the Transform Function.....	618
10.12 Access Control of a Dynamic Table View on Hive.....	619
10.13 Specifying Whether the ADMIN Permissions Is Required for Creating Temporary Functions.....	620
10.14 Using Hive to Read Data in a Relational Database.....	621
10.15 Supporting Traditional Relational Database Syntax in Hive.....	622
10.16 Switching Between Multiple Hive Instances in Hive Editor.....	624
10.17 Creating User-Defined Hive Functions.....	625
10.18 Enhancing beeline Reliability.....	627
10.19 Viewing Table Structures Using the show create Statement as Users with the select Permission..	628
10.20 Writing a Directory into Hive with the Old Data Removed to the Recycle Bin.....	629
10.21 Inserting Data to a Directory That Does Not Exist.....	630
10.22 Creating Databases and Creating Tables in the Default Database Only as the Hive Administrator	631
10.23 Disabling of Specifying the location Keyword When Creating an Internal Hive Table.....	632
10.24 Enabling the Function of Creating a Foreign Table in a Directory That Can Only Be Read.....	633

10.25 Authorizing Over 32 Roles in Hive.....	634
10.26 Restricting the Maximum Number of Maps for Hive Tasks.....	635
10.27 HiveServer Lease Isolation.....	635
10.28 Hive Supporting Transactions.....	636
10.29 Switching the Hive Execution Engine to Tez.....	642
10.30 Hive Materialized View.....	643
10.31 Hive Log Overview.....	646
10.32 Hive Performance Tuning.....	650
10.32.1 Creating Table Partitions.....	650
10.32.2 Optimizing Join.....	651
10.32.3 Optimizing Group By.....	653
10.32.4 Optimizing Data Storage.....	654
10.32.5 Optimizing SQL Statements.....	655
10.32.6 Optimizing the Query Function Using Hive CBO.....	656
10.33 Common Issues About Hive.....	657
10.33.1 How Do I Delete UDFs on Multiple HiveServers at the Same Time?.....	658
10.33.2 Why Cannot the DROP operation Be Performed on a Backed-up Hive Table?.....	659
10.33.3 How to Perform Operations on Local Files with Hive User-Defined Functions.....	660
10.33.4 How Do I Forcibly Stop MapReduce Jobs Executed by Hive?.....	660
10.33.5 How Do I Monitor the Hive Table Size?.....	661
10.33.6 How Do I Prevent Key Directories from Data Loss Caused by Misoperations of the insert overwrite Statement?.....	661
10.33.7 Why Is Hive on Spark Task Freezing When HBase Is Not Installed?.....	662
10.33.8 Error Reported When the WHERE Condition Is Used to Query Tables with Excessive Partitions in FusionInsight Hive.....	663
10.33.9 Why Cannot I Connect to HiveServer When I Use IBM JDK to Access the Beeline Client?.....	663
10.33.10 Description of Hive Table Location (Either Be an OBS or HDFS Path).....	664
10.33.11 Why Cannot Data Be Queried After the MapReduce Engine Is Switched After the Tez Engine Is Used to Execute Union-related Statements?.....	664
10.33.12 Why Does Hive Not Support Concurrent Data Writing to the Same Table or Partition?.....	664
10.33.13 Why Does Hive Not Support Vectorized Query?.....	665
10.33.14 Why Does Metadata Still Exist When the HDFS Data Directory of the Hive Table Is Deleted by Mistake?.....	665
10.33.15 Hive Configuration Problems.....	665
11 Using Hue (Versions Earlier Than MRS 3.x).....	667
11.1 Using Hue from Scratch.....	667
11.2 Accessing the Hue Web UI.....	668
11.3 Hue Common Parameters.....	669
11.4 Using HiveQL Editor on the Hue Web UI.....	670
11.5 Using the Metadata Browser on the Hue Web UI.....	672
11.6 Using File Browser on the Hue Web UI.....	676
11.7 Using Job Browser on the Hue Web UI.....	679
12 Using Hue (MRS 3.x or Later).....	681

12.1 Using Hue from Scratch.....	681
12.2 Accessing the Hue Web UI.....	682
12.3 Hue Common Parameters.....	683
12.4 Using HiveQL Editor on the Hue Web UI.....	684
12.5 Using the SparkSql Editor on the Hue Web UI.....	686
12.6 Using the Metadata Browser on the Hue Web UI.....	688
12.7 Using File Browser on the Hue Web UI.....	689
12.8 Using Job Browser on the Hue Web UI.....	692
12.9 Using HBase on the Hue Web UI.....	693
12.10 Typical Scenarios.....	694
12.10.1 HDFS on Hue.....	694
12.10.2 Hive on Hue.....	698
12.10.3 Oozie on Hue.....	699
12.11 Hue Log Overview.....	701
12.12 Common Issues About Hue.....	703
12.12.1 How Do I Solve the Problem that HQL Fails to Be Executed in Hue Using Internet Explorer?.....	704
12.12.2 Why Does the use database Statement Become Invalid When Hive Is Used?.....	704
12.12.3 What Can I Do If HDFS Files Fail to Be Accessed Using Hue WebUI?.....	704
12.12.4 What Can I Do If a Large File Fails to Be Uploaded on the Hue Page?.....	705
12.12.5 Why Is the Hue Native Page Cannot Be Properly Displayed If the Hive Service Is Not Installed in a Cluster?.....	706
13 Using Impala.....	707
13.1 Using Impala from Scratch.....	707
13.2 Accessing the Impala Web UI.....	709
13.3 Using Impala to Operate Kudu.....	710
13.4 Interconnecting Impala with External LDAP.....	712
14 Using Kafka.....	714
14.1 Using Kafka from Scratch.....	714
14.2 Managing Kafka Topics.....	716
14.3 Querying Kafka Topics.....	720
14.4 Managing Kafka User Permissions.....	720
14.5 Managing Messages in Kafka Topics.....	724
14.6 Synchronizing Binlog-based MySQL Data to the MRS Cluster.....	725
14.7 Creating a Kafka Role.....	731
14.8 Kafka Common Parameters.....	732
14.9 Safety Instructions on Using Kafka.....	737
14.10 Kafka Specifications.....	740
14.11 Using the Kafka Client.....	741
14.12 Configuring Kafka HA and High Reliability Parameters.....	742
14.13 Changing the Broker Storage Directory.....	748
14.14 Checking the Consumption Status of Consumer Group.....	750
14.15 Kafka Balancing Tool Instructions.....	751

14.16	Balancing Data After Kafka Node Scale-Out.....	754
14.17	Kafka Token Authentication Mechanism Tool Usage.....	757
14.18	Introduction to Kafka Logs.....	758
14.19	Performance Tuning.....	761
14.19.1	Kafka Performance Tuning.....	761
14.20	Common Issues About Kafka.....	762
14.20.1	How Do I Solve the Problem that Kafka Topics Cannot Be Deleted?.....	762
14.21	Kafka Feature Description.....	763
15	Using KafkaManager.....	766
15.1	Introduction to KafkaManager.....	766
15.2	Accessing the KafkaManager Web UI.....	766
15.3	Managing Kafka Clusters.....	767
15.4	Kafka Cluster Monitoring Management.....	769
16	Using Kudu.....	777
16.1	Using Kudu from Scratch.....	777
16.2	Accessing the Kudu Web UI.....	778
17	Using Loader.....	780
17.1	Using Loader from Scratch.....	780
17.2	How to Use Loader.....	781
17.3	Loader Link Configuration.....	782
17.4	Managing Loader Links (Versions Earlier Than MRS 3.x).....	784
17.5	Source Link Configurations of Loader Jobs.....	786
17.6	Destination Link Configurations of Loader Jobs.....	789
17.7	Managing Loader Jobs.....	792
17.8	Preparing a Driver for MySQL Database Link.....	795
17.9	Loader Log Overview.....	797
17.10	Example: Using Loader to Import Data from OBS to HDFS.....	800
17.11	Common Issues About Loader.....	801
17.11.1	How to Resolve the Problem that Failed to Save Data When Using Internet Explorer 10 or Internet Explorer 11 ?.....	801
17.11.2	Differences Among Connectors Used During the Process of Importing Data from the Oracle Database to HDFS.....	802
18	Using MapReduce.....	804
18.1	Configuring the Log Archiving and Clearing Mechanism.....	804
18.2	Reducing Client Application Failure Rate.....	806
18.3	Transmitting MapReduce Tasks from Windows to Linux.....	807
18.4	Configuring the Distributed Cache.....	807
18.5	Configuring the MapReduce Shuffle Address.....	810
18.6	Configuring the Cluster Administrator List.....	810
18.7	Introduction to MapReduce Logs.....	811
18.8	MapReduce Performance Tuning.....	814

18.8.1 Optimization Configuration for Multiple CPU Cores.....	814
18.8.2 Determining the Job Baseline.....	819
18.8.3 Streamlining Shuffle.....	821
18.8.4 AM Optimization for Big Tasks.....	824
18.8.5 Speculative Execution.....	825
18.8.6 Using Slow Start.....	826
18.8.7 Optimizing Performance for Committing MR Jobs.....	826
18.9 Common Issues About MapReduce.....	827
18.9.1 Why Does It Take a Long Time to Run a Task Upon ResourceManager Active/Standby Switchover?	827
18.9.2 Why Does a MapReduce Task Stay Unchanged for a Long Time?.....	828
18.9.3 Why the Client Hangs During Job Running?.....	828
18.9.4 Why Cannot HDFS_DELEGATION_TOKEN Be Found in the Cache?.....	829
18.9.5 How Do I Set the Task Priority When Submitting a MapReduce Task?.....	829
18.9.6 Why Physical Memory Overflow Occurs If a MapReduce Task Fails?.....	830
18.9.7 After the Address of MapReduce JobHistoryServer Is Changed, Why the Wrong Page is Displayed When I Click the Tracking URL on the ResourceManager WebUI?.....	831
18.9.8 MapReduce Job Failed in Multiple NameService Environment.....	831
18.9.9 Why a Fault MapReduce Node Is Not Blacklisted?.....	832
19 Using Oozie.....	833
19.1 Using Oozie from Scratch.....	833
19.2 Using the Oozie Client.....	834
19.3 Using Oozie Client to Submit an Oozie Job.....	836
19.3.1 Submitting a Hive Job.....	836
19.3.2 Submitting a Spark2x Job.....	838
19.3.3 Submitting a Loader Job.....	840
19.3.4 Submitting Other Jobs.....	842
19.4 Using Hue to Submit an Oozie Job.....	845
19.4.1 Creating a Workflow.....	845
19.4.2 Submitting a Workflow Job.....	846
19.4.2.1 Submitting a Hive2 Job.....	846
19.4.2.2 Submitting a Spark2x Job.....	847
19.4.2.3 Submitting a Java Job.....	849
19.4.2.4 Submitting a Loader Job.....	849
19.4.2.5 Submitting a MapReduce Job.....	850
19.4.2.6 Submitting a Sub-workflow Job.....	851
19.4.2.7 Submitting a Shell Job.....	852
19.4.2.8 Submitting an HDFS Job.....	853
19.4.2.9 Submitting a Streaming Job.....	854
19.4.2.10 Submitting a DistCp Job.....	855
19.4.2.11 Example of Mutual Trust Operations.....	855
19.4.2.12 Submitting an SSH Job.....	856
19.4.2.13 Submitting a Hive Script.....	857

19.4.3 Submitting a Coordinator Periodic Scheduling Job.....	858
19.4.4 Submitting a Bundle Batch Processing Job.....	859
19.4.5 Querying the Operation Results.....	860
19.5 Oozie Log Overview.....	860
19.6 Common Issues About Oozie.....	863
19.6.1 Oozie Scheduled Tasks Are Not Executed on Time.....	863
19.6.2 Why Update of the share lib Directory of Oozie on HDFS Does Not Take Effect?.....	863
20 Using Presto.....	864
20.1 Accessing the Presto Web UI.....	864
20.2 Using a Client to Execute Query Statements.....	865
20.3 Using Presto to Dump Data in DLF.....	867
20.4 Configuring Presto Permissions.....	869
21 Using Ranger (MRS 1.9.2).....	871
21.1 Creating a Ranger Cluster.....	871
21.2 Accessing the Ranger Web UI and Synchronizing Unix Users to the Ranger Web UI.....	872
21.3 Configuring Hive Access Permissions in Ranger.....	874
21.4 Configuring HBase Access Permissions in Ranger.....	879
22 Using Ranger (MRS 3.x).....	886
22.1 Logging In to the Ranger Web UI.....	886
22.2 Enabling Ranger Authentication.....	888
22.3 Configuring Component Permission Policies.....	888
22.4 Viewing Ranger Audit Information.....	890
22.5 Configuring a Security Zone.....	891
22.6 Changing the Ranger Data Source to LDAP for a Normal Cluster.....	895
22.7 Viewing Ranger Permission Information.....	896
22.8 Adding a Ranger Access Permission Policy for HDFS.....	897
22.9 Adding a Ranger Access Permission Policy for HBase.....	901
22.10 Adding a Ranger Access Permission Policy for Hive.....	905
22.11 Adding a Ranger Access Permission Policy for Yarn.....	915
22.12 Adding a Ranger Access Permission Policy for Spark2x.....	917
22.13 Adding a Ranger Access Permission Policy for Kafka.....	927
22.14 Adding a Ranger Access Permission Policy for Storm.....	936
22.15 Ranger Log Overview.....	939
22.16 Common Issues About Ranger.....	942
22.16.1 Why Ranger Startup Fails During the Cluster Installation?.....	942
22.16.2 How Do I Determine Whether the Ranger Authentication Is Used for a Service?.....	942
22.16.3 Why Cannot a New User Log In to Ranger After Changing the Password?.....	943
22.16.4 When an HBase Policy Is Added or Modified on Ranger, Wildcard Characters Cannot Be Used to Search for Existing HBase Tables.....	943
23 Using Spark.....	945
23.1 Precautions.....	945

23.2 Getting Started with Spark.....	945
23.3 Getting Started with Spark SQL.....	947
23.4 Using the Spark Client.....	950
23.5 Accessing the Spark Web UI.....	950
23.6 Interconnecting Spark with OpenTSDB.....	952
23.6.1 Creating a Table and Associating It with OpenTSDB.....	952
23.6.2 Inserting Data to the OpenTSDB Table.....	953
23.6.3 Querying an OpenTSDB Table.....	954
23.6.4 Modifying the Default Configuration Data.....	954
24 Using Spark2x.....	956
24.1 Precautions.....	956
24.2 Basic Operation.....	956
24.2.1 Getting Started.....	956
24.2.2 Configuring Parameters Rapidly.....	959
24.2.3 Common Parameters.....	968
24.2.4 Spark on HBase Overview and Basic Applications.....	992
24.2.5 Spark on HBase V2 Overview and Basic Applications.....	994
24.2.6 SparkSQL Permission Management(Security Mode).....	996
24.2.6.1 Spark SQL Permissions.....	996
24.2.6.2 Creating a Spark SQL Role.....	1001
24.2.6.3 Configuring Permissions for SparkSQL Tables, Columns, and Databases.....	1005
24.2.6.4 Configuring Permissions for SparkSQL to Use Other Components.....	1007
24.2.6.5 Configuring the Client and Server.....	1009
24.2.7 Scenario-Specific Configuration.....	1011
24.2.7.1 Configuring Multi-active Instance Mode.....	1011
24.2.7.2 Configuring the Multi-tenant Mode.....	1012
24.2.7.3 Configuring the Switchover Between the Multi-active Instance Mode and the Multi-tenant Mode.....	1014
24.2.7.4 Configuring the Size of the Event Queue.....	1015
24.2.7.5 Configuring Executor Off-Heap Memory.....	1016
24.2.7.6 Enhancing Stability in a Limited Memory Condition.....	1017
24.2.7.7 Viewing Aggregated Container Logs on the Web UI.....	1018
24.2.7.8 Configuring Environment Variables in Yarn-Client and Yarn-Cluster Modes.....	1020
24.2.7.9 Configuring the Default Number of Data Blocks Divided by SparkSQL.....	1022
24.2.7.10 Configuring the Compression Format of a Parquet Table.....	1022
24.2.7.11 Configuring the Number of Lost Executors Displayed in WebUI.....	1023
24.2.7.12 Setting the Log Level Dynamically.....	1023
24.2.7.13 Configuring Whether Spark Obtains HBase Tokens.....	1025
24.2.7.14 Configuring LIFO for Kafka.....	1026
24.2.7.15 Configuring Reliability for Connected Kafka.....	1028
24.2.7.16 Configuring Streaming Reading of Driver Execution Results.....	1029
24.2.7.17 Filtering Partitions without Paths in Partitioned Tables.....	1031

24.2.7.18 Configuring Spark2x Web UI ACLs.....	1032
24.2.7.19 Configuring Vector-based ORC Data Reading.....	1034
24.2.7.20 Broaden Support for Hive Partition Pruning Predicate Pushdown.....	1036
24.2.7.21 Hive Dynamic Partition Overwriting Syntax.....	1036
24.2.7.22 Configuring the Column Statistics Histogram to Enhance the CBO Accuracy.....	1037
24.2.7.23 Configuring Local Disk Cache for JobHistory.....	1040
24.2.7.24 Configuring Spark SQL to Enable the Adaptive Execution Feature.....	1040
24.2.7.25 Configuring Event Log Rollover.....	1043
24.2.8 Adapting to the Third-party JDK When Ranger Is Used.....	1045
24.3 Spark2x Logs.....	1045
24.4 Obtaining Container Logs of a Running Spark Application.....	1049
24.5 Small File Combination Tools.....	1049
24.6 Using CarbonData for First Query.....	1052
24.7 Spark2x Performance Tuning.....	1053
24.7.1 Spark Core Tuning.....	1054
24.7.1.1 Data Serialization.....	1054
24.7.1.2 Optimizing Memory Configuration.....	1055
24.7.1.3 Setting the DOP.....	1055
24.7.1.4 Using Broadcast Variables.....	1056
24.7.1.5 Using the external shuffle service to improve performance.....	1056
24.7.1.6 Configuring Dynamic Resource Scheduling in Yarn Mode.....	1057
24.7.1.7 Configuring Process Parameters.....	1058
24.7.1.8 Designing the Direction Acyclic Graph (DAG).....	1060
24.7.1.9 Experience.....	1063
24.7.2 Spark SQL and DataFrame Tuning.....	1064
24.7.2.1 Optimizing the Spark SQL Join Operation.....	1064
24.7.2.2 Improving Spark SQL Calculation Performance Under Data Skew.....	1067
24.7.2.3 Optimizing Spark SQL Performance in the Small File Scenario.....	1068
24.7.2.4 Optimizing the INSERT...SELECT Operation.....	1069
24.7.2.5 Multiple JDBC Clients Concurrently Connecting to JDBCServer.....	1070
24.7.2.6 Optimizing Memory when Data Is Inserted into Dynamic Partitioned Tables.....	1071
24.7.2.7 Optimizing Small Files.....	1071
24.7.2.8 Optimizing the Aggregate Algorithms.....	1072
24.7.2.9 Optimizing Datasource Tables.....	1073
24.7.2.10 Merging CBO.....	1074
24.7.2.11 Optimizing SQL Query of Data of Multiple Sources.....	1076
24.7.2.12 SQL Optimization for Multi-level Nesting and Hybrid Join.....	1079
24.7.3 Spark Streaming Tuning.....	1081
24.8 Common Issues About Spark2x.....	1083
24.8.1 Spark Core.....	1083
24.8.1.1 How Do I View Aggregated Spark Application Logs?.....	1083
24.8.1.2 Why Is the Return Code of Driver Inconsistent with Application State Displayed on ResourceManager WebUI?.....	1083

24.8.1.3 Why Cannot Exit the Driver Process?.....	1084
24.8.1.4 Why Does FetchFailedException Occur When the Network Connection Is Timed out.....	1084
24.8.1.5 How to Configure Event Queue Size If Event Queue Overflows?.....	1086
24.8.1.6 What Can I Do If the getApplicationReport Exception Is Recorded in Logs During Spark Application Execution and the Application Does Not Exit for a Long Time?.....	1086
24.8.1.7 What Can I Do If "Connection to ip:port has been quiet for xxx ms while there are outstanding requests" Is Reported When Spark Executes an Application and the Application Ends?.....	1087
24.8.1.8 Why Do Executors Fail to be Removed After the NodeManager Is Shut Down?.....	1089
24.8.1.9 What Can I Do If the Message "Password cannot be null if SASL is enabled" Is Displayed?.....	1089
24.8.1.10 What Should I Do If the Message "Failed to CREATE_FILE" Is Displayed in the Restarted Tasks When Data Is Inserted Into the Dynamic Partition Table?.....	1090
24.8.1.11 Why Tasks Fail When Hash Shuffle Is Used?.....	1090
24.8.1.12 What Can I Do If the Error Message "DNS query failed" Is Displayed When I Access the Aggregated Logs Page of Spark Applications?.....	1091
24.8.1.13 What Can I Do If Shuffle Fetch Fails Due to the "Timeout Waiting for Task" Exception?.....	1093
24.8.1.14 Why Does the Stage Retry due to the Crash of the Executor?.....	1093
24.8.1.15 Why Do the Executors Fail to Register Shuffle Services During the Shuffle of a Large Amount of Data?.....	1093
24.8.1.16 Why Does the Out of Memory Error Occur in NodeManager During the Execution of Spark Applications.....	1095
24.8.1.17 Why Does the Realm Information Fail to Be Obtained When SparkBench is Run on HiBench for the Cluster in Security Mode?.....	1096
24.8.2 Spark SQL and DataFrame.....	1097
24.8.2.1 What Do I have to Note When Using Spark SQL ROLLUP and CUBE?.....	1097
24.8.2.2 Why Spark SQL Is Displayed as a Temporary Table in Different Databases?.....	1098
24.8.2.3 How to Assign a Parameter Value in a Spark Command?.....	1099
24.8.2.4 What Directory Permissions Do I Need to Create a Table Using SparkSQL?.....	1099
24.8.2.5 Why Do I Fail to Delete the UDF Using Another Service?.....	1100
24.8.2.6 Why Cannot I Query Newly Inserted Data in a Parquet Hive Table Using SparkSQL?.....	1101
24.8.2.7 How to Use Cache Table?.....	1101
24.8.2.8 Why Are Some Partitions Empty During Repartition?.....	1102
24.8.2.9 Why Does 16 Terabytes of Text Data Fails to Be Converted into 4 Terabytes of Parquet Data?.....	1103
24.8.2.10 Why the Operation Fails When the Table Name Is TABLE?.....	1104
24.8.2.11 Why Is a Task Suspended When the ANALYZE TABLE Statement Is Executed and Resources Are Insufficient?.....	1104
24.8.2.12 If I Access a parquet Table on Which I Do not Have Permission, Why a Job Is Run Before "Missing Privileges" Is Displayed?.....	1105
24.8.2.13 Why Do I Fail to Modify MetaData by Running the Hive Command?.....	1106
24.8.2.14 Why Is "RejectedExecutionException" Displayed When I Exit Spark SQL?.....	1106
24.8.2.15 What Should I Do If the JDBCServer Process is Mistakenly Killed During a Health Check?.....	1106
24.8.2.16 Why No Result Is found When 2016-6-30 Is Set in the Date Field as the Filter Condition?.....	1107
24.8.2.17 Why Does the "--hivevar" Option I Specified in the Command for Starting spark-beeline Fail to Take Effect?.....	1108
24.8.2.18 Why Does the "Permission denied" Exception Occur When I Create a Temporary Table or View in Spark-beeline?.....	1108

24.8.2.19 Why Is the "Code of method ... grows beyond 64 KB" Error Message Displayed When I Run Complex SQL Statements?.....	1109
24.8.2.20 Why Is Memory Insufficient if 10 Terabytes of TPCDS Test Suites Are Consecutively Run in Beeline/JDBCServer Mode?.....	1109
24.8.2.21 Why Are Some Functions Not Available when Another JDBCServer Is Connected?.....	1110
24.8.2.22 Why Does Spark2x Have No Access to DataSource Tables Created by Spark1.5?.....	1111
24.8.2.23 Why Does Spark-beeline Fail to Run and Error Message "Failed to create ThriftService instance" Is Displayed?.....	1112
24.8.3 Spark Streaming.....	1113
24.8.3.1 What Can I Do If Spark Streaming Tasks Are Blocked?.....	1113
24.8.3.2 What Should I Pay Attention to When Optimizing Spark Streaming Task Parameters?.....	1114
24.8.3.3 Why Does the Spark Streaming Application Fail to Be Submitted After the Token Validity Period Expires?.....	1115
24.8.3.4 Why does Spark Streaming Application Fail to Restart from Checkpoint When It Creates an Input Stream Without Output Logic?.....	1116
24.8.3.5 Why Is the Input Size Corresponding to Batch Time on the Web UI Set to 0 Records When Kafka Is Restarted During Spark Streaming Running?.....	1117
24.8.4 Why the Job Information Obtained from the restful Interface of an Ended Spark Application Is Incorrect?.....	1118
24.8.5 Why Cannot I Switch from the Yarn Web UI to the Spark Web UI?.....	1119
24.8.6 What Can I Do If an Error Occurs when I Access the Application Page Because the Application Cached by HistoryServer Is Recycled?.....	1120
24.8.7 Why Is not an Application Displayed When I Run the Application with the Empty Part File?.....	1121
24.8.8 Why Does Spark2x Fail to Export a Table with the Same Field Name?.....	1122
24.8.9 Why JRE fatal error after running Spark application multiple times?.....	1122
24.8.10 "This page can't be displayed" Is Displayed When Internet Explorer Fails to Access the Native Spark2x UI.....	1122
24.8.11 How Does Spark2x Access External Cluster Components?.....	1123
24.8.12 Why Does the Foreign Table Query Fail When Multiple Foreign Tables Are Created in the Same Directory?.....	1125
24.8.13 What Should I Do If the Native Page of an Application of Spark2x JobHistory Fails to Display During Access to the Page.....	1126
24.8.14 Why Do I Fail to Create a Table in the Specified Location on OBS After Logging to spark-beeline?.....	1126
24.8.15 Spark Shuffle Exception Handling.....	1127
25 Using Storm.....	1129
25.1 Using Storm from Scratch.....	1129
25.2 Using the Storm Client.....	1130
25.3 Submitting Storm Topologies on the Client.....	1131
25.4 Accessing the Storm Web UI.....	1132
25.5 Managing Storm Topologies.....	1134
25.6 Querying Storm Topology Logs.....	1135
25.7 Storm Common Parameters.....	1135
25.8 Configuring a Storm Service User Password Policy.....	1137
25.9 Migrating Storm Services to Flink.....	1139

25.9.1 Overview.....	1139
25.9.2 Completely Migrating Storm Services.....	1139
25.9.3 Performing Embedded Service Migration.....	1141
25.9.4 Migrating Services of External Security Components Interconnected with Storm.....	1141
25.10 Storm Log Introduction.....	1142
25.11 Performance Tuning.....	1147
25.11.1 Storm Performance Tuning.....	1147
26 Using Tez.....	1150
26.1 Precautions.....	1150
26.2 Common Tez Parameters.....	1150
26.3 Accessing TezUI.....	1150
26.4 Log Overview.....	1151
26.5 Common Issues.....	1153
26.5.1 TezUI Cannot Display Tez Task Execution Details.....	1153
26.5.2 Error Occurs When a User Switches to the Tez Web UI.....	1153
26.5.3 Yarn Logs Cannot Be Viewed on the TezUI Page.....	1154
26.5.4 Table Data Is Empty on the TezUI HiveQueries Page.....	1155
27 Using Yarn.....	1156
27.1 Common Yarn Parameters.....	1156
27.2 Creating Yarn Roles.....	1160
27.3 Using the Yarn Client.....	1162
27.4 Configuring Resources for a NodeManager Role Instance.....	1164
27.5 Changing NodeManager Storage Directories.....	1165
27.6 Configuring Strict Permission Control for Yarn.....	1169
27.7 Configuring Container Log Aggregation.....	1171
27.8 Using CGroups with YARN.....	1178
27.9 Configuring the Number of ApplicationMaster Retries.....	1180
27.10 Configure the ApplicationMaster to Automatically Adjust the Allocated Memory.....	1180
27.11 Configuring the Access Channel Protocol.....	1182
27.12 Configuring Memory Usage Detection.....	1183
27.13 Configuring the Additional Scheduler WebUI.....	1184
27.14 Configuring Yarn Restart.....	1185
27.15 Configuring ApplicationMaster Work Preserving.....	1187
27.16 Configuring the Localized Log Levels.....	1188
27.17 Configuring Users That Run Tasks.....	1189
27.18 Yarn Log Overview.....	1190
27.19 Yarn Performance Tuning.....	1193
27.19.1 Preempting a Task.....	1193
27.19.2 Setting the Task Priority.....	1196
27.19.3 Optimizing Node Configuration.....	1197
27.20 Common Issues About Yarn.....	1203

27.20.1 Why Mounted Directory for Container is Not Cleared After the Completion of the Job While Using CGroups?.....	1203
27.20.2 Why the Job Fails with HDFS_DELEGATION_TOKEN Expired Exception?.....	1204
27.20.3 Why Are Local Logs Not Deleted After YARN Is Restarted?.....	1204
27.20.4 Why the Task Does Not Fail Even Though AppAttempts Restarts for More Than Two Times?...	1205
27.20.5 Why Is an Application Moved Back to the Original Queue After ResourceManager Restarts?...	1205
27.20.6 Why Does Yarn Not Release the Blacklist Even All Nodes Are Added to the Blacklist?	1205
27.20.7 Why Does the Switchover of ResourceManager Occur Continuously?.....	1206
27.20.8 Why Does a New Application Fail If a NodeManager Has Been in Unhealthy Status for 10 Minutes?.....	1207
27.20.9 Why Does an Error Occur When I Query the ApplicationID of a Completed or Non-existing Application Using the RESTful APIs?.....	1207
27.20.10 Why May A Single NodeManager Fault Cause MapReduce Task Failures in the Superior Scheduling Mode?.....	1208
27.20.11 Why Are Applications Suspended After They Are Moved From Lost_and_Found Queue to Another Queue?.....	1208
27.20.12 How Do I Limit the Size of Application Diagnostic Messages Stored in the ZKstore?.....	1209
27.20.13 Why Does a MapReduce Job Fail to Run When a Non-ViewFS File System Is Configured as ViewFS?.....	1210
27.20.14 Why Do Reduce Tasks Fail to Run in Some OSs After the Native Task Feature is Enabled?.....	1211
28 Using ZooKeeper.....	1212
28.1 Configuring the ZooKeeper Permissions.....	1212
29 Appendix.....	1217
29.1 Modifying Cluster Service Configuration Parameters.....	1217
29.2 Accessing Manager.....	1218
29.2.1 Accessing MRS Manager (Versions Earlier Than MRS 3.x).....	1219
29.2.2 Accessing FusionInsight Manager (MRS 3.x or Later).....	1221
29.3 Using an MRS Client.....	1223
29.3.1 Installing a Client (Version 3.x or Later).....	1223
29.3.2 Installing a Client (Versions Earlier Than 3.x).....	1227
29.3.3 Updating a Client (Version 3.x or Later).....	1232
29.3.4 Updating a Client (Versions Earlier Than 3.x).....	1234

1 Using Alluxio

1.1 Configuring an Underlying Storage System

If you want to use a unified client API and a global namespace to access persistent storage systems including HDFS and OBS to separate computing from storage, you can configure the underlying storage system of Alluxio on MRS Manager. After a cluster is created, the default underlying storage address is **hdfs://hacluster/**, that is, the HDFS root directory is mapped to Alluxio.

Prerequisites

- Alluxio has been installed in a cluster.
- The password of user **admin** has been obtained. The password of user **admin** is specified by the user during MRS cluster creation.

Configuring HDFS as the Underlying File System of Alluxio

NOTE

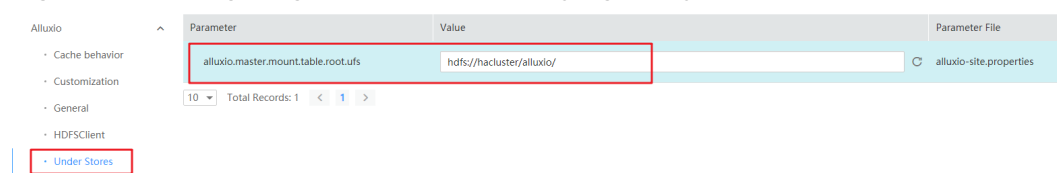
Security clusters with Kerberos authentication enabled do not support this function.

Step 1 Go to the **All Configurations** page of Alluxio. See [Modifying Cluster Service Configuration Parameters](#).

Step 2 In the left pane, choose **Alluxio > Under Stores**, and modify the value of **alluxio.master.mount.table.root.ufs** to **hdfs://hacluster/XXX/**.

For example, if you want to use *HDFS root directory/alluxio/* as the root directory of Alluxio, modify the value of **alluxio.master.mount.table.root.ufs** to **hdfs://hacluster/alluxio/**.

Figure 1-1 Configuring HDFS as the underlying file system of Alluxio



Step 3 Click **Save Configuration**. In the displayed dialog box, select **Restart the affected services or instances**.

Step 4 Click **OK** to restart Alluxio.

----End

1.2 Accessing Alluxio Using a Data Application

The port number used for accessing the Alluxio file system is 19998, and the access address is **alluxio://<Master node IP address of Alluxio>:19998/<PATH>**. This section uses examples to describe how to access the Alluxio file system using data applications (Spark, Hive, Hadoop MapReduce, and Presto).

Using Alluxio as the Input and Output of a Spark Application

Step 1 Log in to the Master node in a cluster as user **root** using the password set during cluster creation.

Step 2 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 3 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step:

```
kinit MRS cluster user
```

Example: **kinit admin**

Step 4 Prepare an input file and copy local data to the Alluxio file system.

For example, prepare the input file **test_input.txt** in the local **/home** directory, and run the following command to save the **test_input.txt** file to Alluxio:

```
alluxio fs copyFromLocal /home/test_input.txt /input
```

Step 5 Run the following commands to start **spark-shell**:

```
spark-shell
```

Step 6 Run the following commands in **spark-shell**:

```
val s = sc.textFile("alluxio://<Name of the Alluxio node>:19998/input")
```

```
val double = s.map(line => line + line)
```

```
double.saveAsTextFile("alluxio://<Name of the Alluxio node>:19998/output")
```

NOTE

Replace ***Name of the Alluxio node*:19998** with the actual node name and port numbers of all nodes where the AlluxioMaster instance is deployed. Use commas (,) to separate the node name and port number, for example, **node-ana-coremspb.mrs-m0va.com:19998,node-master2kiww.mrs-m0va.com:19998,node-master1cqwv.mrs-m0va.com:19998**.

Step 7 Press **Ctrl+C** to exit **spark-shell**.

Step 8 Run the `alluxio fs ls /` command to check whether the output directory `/output` containing double content of the input file exists in the root directory of Alluxio.

----End

Creating a Hive Table on Alluxio

Step 1 Log in to the Master node in a cluster as user `root` using the password set during cluster creation.

Step 2 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 3 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step:

```
kinit MRS cluster user
```

Example: `kinit admin`

Step 4 Prepare an input file. For example, prepare the `hive_load.txt` input file in the local `/home` directory. The file content is as follows:

```
1, Alice, company A  
2, Bob, company B
```

Step 5 Run the following command to import the `hive_load.txt` file to Alluxio:

```
alluxio fs copyFromLocal /home/hive_load.txt /hive_input
```

Step 6 Run the following command to start the Hive beeline:

```
beeline
```

Step 7 Run the following commands in beeline to create a table based on the input file in Alluxio:

```
CREATE TABLE u_user(id INT, name STRING, company STRING) ROW FORMAT  
DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE;
```

```
LOAD DATA INPATH 'alluxio://<Name of the Alluxio node>:19998/hive_input'  
INTO TABLE u_user;
```

NOTE

Replace `<Name of the Alluxio node>:19998` with the actual node name and port numbers of all nodes where the AlluxioMaster instance is deployed. Use commas (,) to separate the node name and port number, for example, `node-ana-coremspb.mrs-m0va.com:19998,node-master2kiww.mrs-m0va.com:19998,node-master1cqww.mrs-m0va.com:19998`.

Step 8 Run the following command to view the created table:

```
select * from u_user;
```

----End

Running Hadoop Wordcount in Alluxio

Step 1 Log in to the Master node in a cluster as user **root** using the password set during cluster creation.

Step 2 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 3 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step:

```
kinit MRS cluster user
```

Example: **kinit admin**

Step 4 Prepare an input file and copy local data to the Alluxio file system.

For example, prepare the input file **test_input.txt** in the local **/home** directory, and run the following command to save the **test_input.txt** file to Alluxio:

```
alluxio fs copyFromLocal /home/test_input.txt /input
```

Step 5 Run the following command to execute the wordcount job:

```
yarn jar /opt/share/hadoop-mapreduce-examples-<Hadoop version>-mrs-  
<MRS cluster version>/hadoop-mapreduce-examples-<Hadoop version>-mrs-  
<MRS cluster version>.jar wordcount alluxio://<Name of the Alluxio node>:  
19998/input alluxio://<Name of the Alluxio node>:19998/output
```

NOTE

- Replace **<Hadoop version>** with the actual one.
- Replace **<MRS cluster version>** with the major version of MRS. For example, for a cluster of MRS 1.9.2, **mrs-1.9.0** is used.
- Replace **Name of the Alluxio node>:19998** with the actual node name and port numbers of all nodes where the AlluxioMaster instance is deployed. Use commas (,) to separate the node name and port number, for example, **node-ana-coremspb.mrs-m0va.com:19998,node-master2kiww.mrs-m0va.com:19998,node-master1cqvw.mrs-m0va.com:19998**.

Step 6 Run the **alluxio fs ls /** command to check whether the output directory **/output** containing the wordcount result exists in the root directory of Alluxio.

----End

Using Presto to Query Tables in Alluxio

Step 1 Log in to the Master node in a cluster as user **root** using the password set during cluster creation.

Step 2 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 3 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step:

kinit *MRS cluster user*

Example: **kinit admin**

Step 4 Run the following commands to start Hive Beeline to create a table on Alluxio.

beeline

```
CREATE TABLE u_user (id int, name string, company string) ROW FORMAT  
DELIMITED FIELDS TERMINATED BY ',' LOCATION 'alluxio://<Name of the  
Alluxio node>:19998/u_user';
```

```
insert into u_user values(1,'Alice','Company A'),(2, 'Bob', 'Company B');
```

 **NOTE**

Replace *Name of the Alluxio node*:19998 with the actual node name and port numbers of all nodes where the AlluxioMaster instance is deployed. Use commas (,) to separate the node name and port number, for example, **node-ana-coremspb.mrs-m0va.com:19998,node-master2kiww.mrs-m0va.com:19998,node-master1cqww.mrs-m0va.com:19998**.

Step 5 Start the Presto client. For details, see [Step 2](#) to [Step 8](#) in [Using a Client to Execute Query Statements](#).

Step 6 On the Presto client, run the **select * from hive.default.u_user;** statement to query the table created in Alluxio:

Figure 1-2 Using Presto to query the table created in Alluxio

```
presto> select * from hive.default.u_user;  
id | name | company  
----+-----+-----  
 1 | Alice | Company A  
 2 | Bob   | Company B  
(2 rows)
```

----End

1.3 Common Operations of Alluxio

Preparations

1. Create a cluster with Alluxio installed.
2. Log in to the active Master node in a cluster as user **root** using the password set during cluster creation.
3. Run the following command to configure environment variables:
source /opt/client/bigdata_env

Using the Alluxio Shell

The [Alluxio shell](#) contains multiple command line operations that interact with Alluxio.

- View a file system operation command list:
alluxio fs

- Run the **ls** command to list the files in Alluxio. For example, list all files in the root directory:
alluxio fs ls /
- Run the **copyFromLocal** command to copy local files to Alluxio:
alluxio fs copyFromLocal /home/test_input.txt /test_input.txt
Command output:
Copied file:///home/test_input.txt to /test_input.txt
- Run the **ls** command again to list the files in Alluxio. The copied **test_input.txt** file is listed:
alluxio fs ls /
Command output:
12 PERSISTED 11-28-2019 17:10:17:449 100% /test_input.txt
The **test_input.txt** file is displayed in Alluxio. The parameters in the file indicate the file size, whether the file is persistent, creation date, cache ratio of the file in Alluxio, and file name.
- Run the **cat** command to print file content:
alluxio fs cat /test_input.txt
Command output:
Test Alluxio

Mounting Function of Alluxio

Alluxio uses a unified namespace feature to unify the access to storage systems. For details, see <https://docs.alluxio.io/os/user/2.0/en/advanced/Namespcae-Management.html>.

This feature allows users to mount different storage systems to an Alluxio namespace and seamlessly access files across storage systems through the Alluxio namespace.

1. Create a directory as a mount point in Alluxio.

alluxio fs mkdir /mnt

Successfully created directory /mnt

2. Mount an existing OBS file system to Alluxio. (Prerequisite: An agency with the **OBS OperateAccess** permission has been configured for the cluster. The **obs-mrstest** file system is used as an example. Replace the file system name with the actual one.

alluxio fs mount /mnt/obs obs:///obs-mrstest/data

Mounted obs:///obs-mrstest/data at /mnt/obs

3. List files in the OBS file system using the Alluxio namespace. Run the **ls** command to list the files in the OBS mount directory.

alluxio fs ls /mnt/obs

38 PERSISTED 11-28-2019 17:42:54:554 0% /mnt/obs/hive_load.txt

12 PERSISTED 11-28-2019 17:43:07:743 0% /mnt/obs/test_input.txt

You can also view the newly mounted files and directories on the Alluxio web UI.

4. After the mounting is complete, you can seamlessly exchange data between different storage systems through the unified namespace of Alluxio. For example, run the **ls -R** command to list all files in a directory recursively:

alluxio fs ls -R /

```

0    PERSISTED 11-28-2019 11:15:19:719 DIR /app-logs
1    PERSISTED 11-28-2019 11:18:36:885 DIR /apps
1    PERSISTED 11-28-2019 11:18:40:209 DIR /apps/templeton
239440292  PERSISTED 11-28-2019 11:18:40:209 0% /apps/templeton/hive.tar.gz
.....
1    PERSISTED 11-28-2019 19:00:23:879 DIR /mnt
2    PERSISTED 11-28-2019 19:00:23:879 DIR /mnt/obs
38   PERSISTED 11-28-2019 17:42:54:554 0% /mnt/obs/hive_load.txt
12   PERSISTED 11-28-2019 17:43:07:743 0% /mnt/obs/test_input.txt
.....

```

The command output shows all files that are from the mounted storage system in the root directory of the Alluxio file system (the default directory is the HDFS root directory, that is, **hdfs://hacluster/**). The **/app-logs** and **/apps** directories are in HDFS, and the **/mnt/obs/** directory is in OBS.

Using Alluxio to Accelerate Data Access

Alluxio can accelerate data access, because it uses memory to store data. Example commands are provided as follows:

1. Upload the **test_data.csv** file (a sample that records recipes) to the **/data** directory of the **obs-mrtest** file system. Run the **ls** command to display the file status.

```
alluxio fs ls /mnt/obs/test_data.csv
```

```
294520189  PERSISTED 11-28-2019 19:38:55:000 0% /mnt/obs/test_data.csv
```

The output indicates that the cache percentage of the file in Alluxio is 0%, that is, the file is not in Alluxio memory.

2. Count the occurrence times of the word "milk" in the file, and calculate the time consumed.

```
time alluxio fs cat /mnt/obs/test_data.csv | grep -c milk
```

```
52180
```

```
real 0m10.765s
user 0m5.540s
sys 0m0.696s
```

3. Data is stored in memory after being read for the first time. When Alluxio reads data again, the data access speed is increased. For example, after running the **cat** command to obtain a file, run the **ls** command to check the file status.

```
alluxio fs ls /mnt/obs/test_data.csv
```

```
294520189  PERSISTED 11-28-2019 19:38:55:000 100% /mnt/obs/test_data.csv
```

The output shows that the file has been fully loaded to Alluxio.

4. Access the file again, count the occurrence times of the word "eggs", and calculate the time consumed.

```
time alluxio fs cat /mnt/obs/test_data.csv | grep -c eggs
```

```
59510
```

```
real 0m5.777s
user 0m5.992s
sys 0m0.592s
```

According to the comparison of the two time consumption records, the time consumed for accessing data stored in Alluxio memory is significantly reduced.

2 Using CarbonData (for Versions Earlier Than MRS 3.x)

2.1 Getting Started with CarbonData

This section is for MRS 3.x or earlier. For MRS 3.x or later, see [Using CarbonData \(for MRS 3.x or Later\)](#).

This section describes the procedure of using Spark CarbonData. All tasks are based on the Spark-beeline environment. The tasks include:

1. Connecting to Spark
Before performing any operation on CarbonData, users must connect CarbonData to Spark.
2. Creating a CarbonData table
After connecting to Spark, users must create a CarbonData table to load and query data.
3. Loading data to the CarbonData table
Users load data from CSV files in HDFS to the CarbonData table.
4. Querying data from the CarbonData table
After data is loaded to the CarbonData table, users can run query commands such as **groupby** and **where**.

Prerequisites

A client has been installed. For details, see [Using an MRS Client](#).

Procedure

Step 1 Connect CarbonData to Spark.

1. Prepare a client based on service requirements and use user **root** to log in to the node where the client is installed.
For example, if you have updated the client on the Master2 node, log in to the Master2 node to use the client. For details, see [Using an MRS Client](#).

2. Run the following commands to switch the user and configure environment variables:

```
sudo su - omm  
source /opt/client/bigdata_env
```

3. For clusters with Kerberos authentication enabled, run the following command to authenticate the user. For clusters with Kerberos authentication disabled, skip this step.

```
kinit Spark username
```

 **NOTE**

The user needs to be added to user groups **hadoop** (primary group) and **hive**.

4. Run the following command to connect to the Spark environment.

```
spark-beeline
```

Step 2 Create a CarbonData table.

Run the following command to create a CarbonData table, which is used to load and query data.

```
CREATE TABLE x1 (imei string, deviceInformationId int, mac string,  
productdate timestamp, updatetime timestamp, gamePointId double,  
contractNumber double)  
  
STORED BY 'org.apache.carbondata.format'  
  
TBLPROPERTIES  
('DICTIONARY_EXCLUDE'='mac','DICTIONARY_INCLUDE'='deviceInformationId'  
);
```

The command output is as follows:

```
+-----+  
| result |  
+-----+  
+-----+  
No rows selected (1.551 seconds)
```

Step 3 Load data from CSV files to the CarbonData table.

Run the command to load data from CSV files based on the required parameters. Only CSV files are supported. The CSV column name and sequence configured in the **LOAD** command must be consistent with those in the CarbonData table. The data formats and number of data columns in the CSV files must also be the same as those in the CarbonData table.

The CSV files must be stored on HDFS. You can upload the files to OBS and import them from OBS to HDFS on the **Files** page of the MRS console.

If Kerberos authentication is enabled, prepare the CSV files in the work environment and import them to HDFS using open-source HDFS commands. In addition, assign the Spark user with the read and execute permissions of the files on HDFS by referring to [5](#).

For example, the **data.csv** file is saved in the **tmp** directory of HDFS with the following contents:

```
x123,111,dd,2017-04-20 08:51:27,2017-04-20 07:56:51,2222,33333
```

The command for loading data from that file is as follows:

```
LOAD DATA inpath 'hdfs://hacluster/tmp/data.csv' into table x1
options('DELIMITER',';', 'QUOTECHAR','"', 'FILEHEADER='imei,
deviceinformationid,mac,productdate,updatetime,gamepointid,contractnumb
er');
```

The command output is as follows:

```
+-----+
| Result |
+-----+
+-----+
No rows selected (3.039 seconds)
```

Step 4 Query data from the CarbonData.

- **Obtaining the number of records**

Run the following command to obtain the number of records in the CarbonData table:

```
select count(*) from x1;
```

- **Querying with the groupby condition**

Run the following command to obtain the **deviceinformationid** records without repetition in the CarbonData table:

```
select deviceinformationid,count (distinct deviceinformationid) from x1
group by deviceinformationid;
```

- **Querying with the where condition**

Run the following command to obtain specific **deviceinformationid** records:

```
select * from x1 where deviceinformationid='111';
```

 **NOTE**

If the query result have other non-English characters, the columns in the query result may not be aligned. This is because characters of different languages occupy different widths.

Step 5 Run the following command to exit the Spark environment.

```
!quit
```

```
----End
```

2.2 About CarbonData Table

Description

CarbonData tables are similar to tables in the relational database management system (RDBMS). RDBMS tables consist of rows and columns to store data. CarbonData tables have fixed columns and also store structured data. In CarbonData, data is saved in entity files.

Supported Data Types

CarbonData tables support the following data types:

- Int
- String
- BigInt
- Decimal
- Double
- TimeStamp

Table 2-1 describes the details about each data type.

Table 2-1 CarbonData data types

Data Type	Description
Int	4-byte signed integer ranging from -2,147,483,648 to 2,147,483,647 NOTE If a non-dictionary column is of the int data type, it is internally stored as the BigInt type.
String	The maximum character string length is 100000.
BigInt	Data is saved using the 64-bit technology. The value ranges from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807.
Decimal	The default value is (10,0) and maximum value is (38,38). NOTE When query with filters, append BD to the number to achieve accurate results. For example, select * from carbon_table where num = 1234567890123456.22BD .
Double	Data is saved using the 64-bit technology. The value ranges from 4.9E-324 to 1.7976931348623157E308.
TimeStamp	yyyy-MM-dd HH:mm:ss format is used by default.

 **NOTE**

Measurement of all Integer data is processed and displayed using the **BigInt** data type.

2.3 Creating a CarbonData Table

Scenario

A CarbonData table must be created to load and query data.

Creating a Table with Self-Defined Columns

Users can create a table by specifying its columns and data types. For analysis clusters with Kerberos authentication enabled, if a user wants to create a CarbonData table in a database other than the **default** database, the **Create**

permission of the database must be added to the role to which the user is bound in Hive role management.

Sample command:

```
CREATE TABLE IF NOT EXISTS productdb.productSalesTable (  
productNumber Int,  
productName String,  
storeCity String,  
storeProvince String,  
revenue Int)  
STORED BY 'org.apache.carbondata.format'  
TBLPROPERTIES (  
'table_blocksize'='128',  
'DICTIONARY_EXCLUDE'='productName',  
'DICTIONARY_INCLUDE'='productNumber');
```

The following table describes parameters of preceding commands.

Table 2-2 Parameter description

Parameter	Description
productSalesTable	Table name. The table is used to load data for analysis. The table name consists of letters, digits, and underscores (_).
productdb	Database name. The database maintains logical connections with tables stored in it to identify and manage the tables. The database name consists of letters, digits, and underscores (_).
productNumber productName storeCity storeProvince revenue	Columns in the table. The columns are service entities for data analysis. The column name (field name) consists of letters, digits, and underscores (_). NOTE In CarbonData, you cannot configure a column's NOT NULL or default value, or the primary key of the table.

Parameter	Description
table_blocksize	<p>Block size of data files used by the CarbonData table. The value ranges from 1 MB to 2048 MB. The default is 1024 MB.</p> <ul style="list-style-type: none"> • If the value of table_blocksize is too small, a large number of small files will be generated when data is loaded. This may affect the performance in using HDFS. • If the value of table_blocksize is too large, a large volume of data must be read from a block and the read concurrency is low when data is queried. As a result, the query performance deteriorates. <p>You are advised to set the block size based on the data volume. For example, set the block size to 256 MB for GB-level data, 512 MB for TB-level data, and 1024 MB for PB-level data.</p>
DICTIONARY_EXCLUDE	<p>Specifies the columns that do not generate dictionaries. This function is optional and applicable to columns of high complexity. By default, the system generates dictionaries for columns of the String type. However, as the number of values in the dictionaries increases, conversion operations by the dictionaries increase and the system performance deteriorates.</p> <p>Generally, if a column has over 50,000 unique data records, it is considered as a highly complex column and dictionary generation must be disabled.</p> <p>NOTE Non-dictionary columns support only the String and Timestamp data types.</p>
DICTIONARY_INCLUDE	<p>Specifies the columns that generate dictionaries. This function is optional and applicable to columns of low complexity. It improves the performance of queries with the groupby condition. Generally, the complexity of a dictionary column cannot exceed 50,000.</p>

2.4 Deleting a CarbonData Table

Scenario

Unused CarbonData tables can be deleted. After a CarbonData table is deleted, its metadata and loaded data are deleted together.

Procedure

Step 1 Run the following command to delete a CarbonData table:

```
DROP TABLE [IF EXISTS] [db_name.]table_name;
```

db_name is optional. If **db_name** is not specified, the table named **table_name** in the current database is deleted.

For example, run the following command to delete the **productSalesTable** table in the **productdb** database:

```
DROP TABLE productdb.productSalesTable;
```

Step 2 Run the following command to confirm that the table is deleted:

```
SHOW TABLES;
```

```
----End
```

3 Using CarbonData (for MRS 3.x or Later)

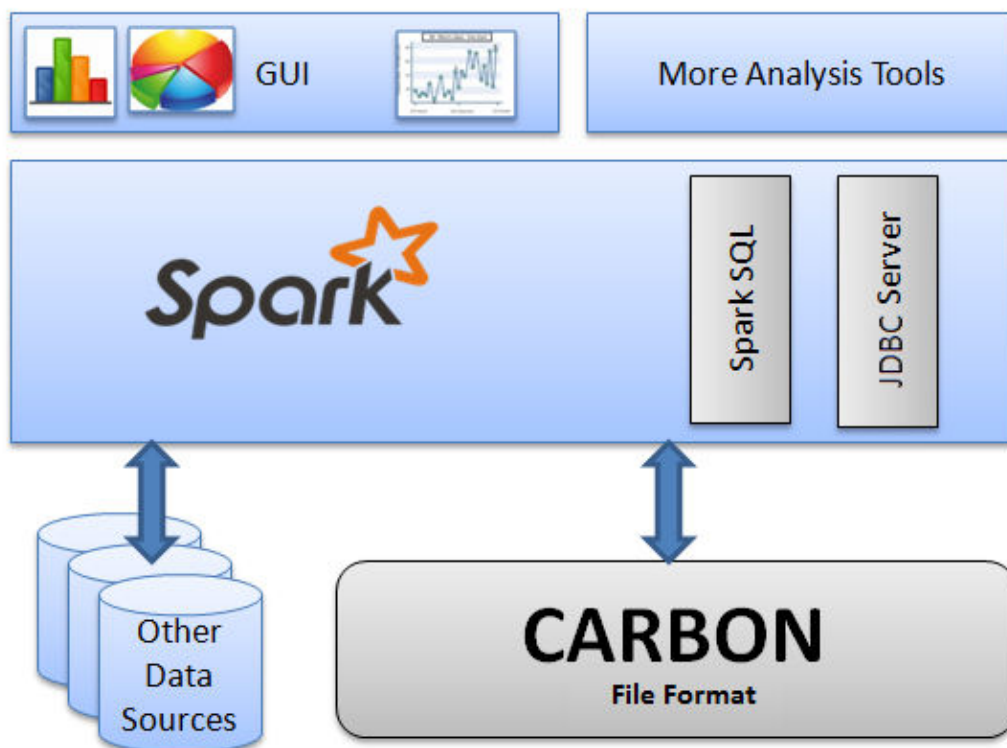
3.1 Overview

This section is for MRS 3.x or later. For MRS 3.x or earlier, see [Using CarbonData \(for Versions Earlier Than MRS 3.x\)](#).

3.1.1 CarbonData Overview

CarbonData is a new Apache Hadoop native data-store format. CarbonData allows faster interactive queries over PetaBytes of data using advanced columnar storage, index, compression, and encoding techniques to improve computing efficiency. In addition, CarbonData is also a high-performance analysis engine that integrates data sources with Spark.

Figure 3-1 Basic architecture of CarbonData



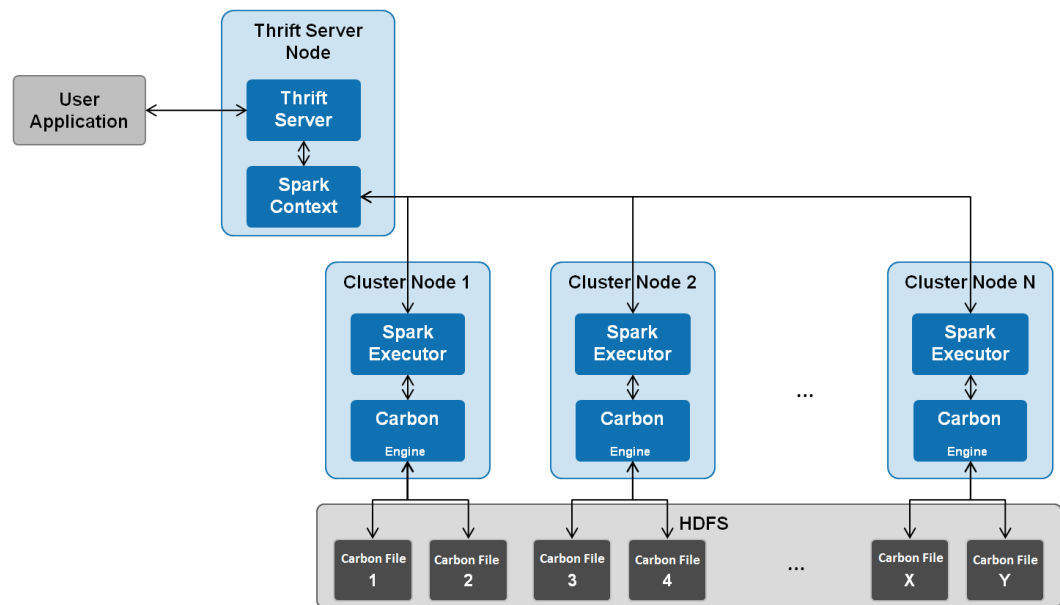
The purpose of using CarbonData is to provide quick response to ad hoc queries of big data. Essentially, CarbonData is an Online Analytical Processing (OLAP) engine, which stores data by using tables similar to those in Relational Database Management System (RDBMS). You can import more than 10 TB data to tables created in CarbonData format, and CarbonData automatically organizes and stores data using the compressed multi-dimensional indexes. After data is loaded to CarbonData, CarbonData responds to ad hoc queries in seconds.

CarbonData integrates data sources into the Spark ecosystem and you can query and analyze the data using Spark SQL. You can also use the third-party tool JDBCServer provided by Spark to connect to SparkSQL.

Topology of CarbonData

CarbonData runs as a data source inside Spark. Therefore, CarbonData does not start any additional processes on nodes in clusters. CarbonData engine runs inside the Spark executor.

Figure 3-2 Topology of CarbonData



Data stored in CarbonData Table is divided into several CarbonData data files. Each time when data is queried, CarbonData Engine reads and filters data sets. CarbonData Engine runs as a part of the Spark Executor process and is responsible for handling a subset of data file blocks.

Table data is stored in HDFS. Nodes in the same Spark cluster can be used as HDFS data nodes.

CarbonData Features

- SQL: CarbonData is compatible with Spark SQL and supports SQL query operations performed on Spark SQL.
- Simple Table dataset definition: CarbonData allows you to define and create datasets by using user-friendly Data Definition Language (DDL) statements. CarbonData DDL is flexible and easy to use, and can define complex tables.
- Easy data management: CarbonData provides various data management functions for data loading and maintenance. CarbonData supports bulk loading of historical data and incremental loading of new data. Loaded data can be deleted based on load time and a specific loading operation can be undone.
- CarbonData file format is a columnar store in HDFS. This format has many new column-based file storage features, such as table splitting and data compression. CarbonData has the following characteristics:
 - Stores data along with index: Significantly accelerates query performance and reduces the I/O scans and CPU resources, when there are filters in the query. CarbonData index consists of multiple levels of indices. A processing framework can leverage this index to reduce the task that needs to be scheduled and processed, and it can also perform skip scan in more finer grain unit (called blocklet) in task side scanning instead of scanning the whole file.
 - Operable encoded data: Through supporting efficient compression, CarbonData can query on compressed/encoded data. The data can be

converted just before returning the results to the users, which is called late materialized.

- Support for various use cases with one single data format: like interactive OLAP-style query, sequential access (big scan), and random access (narrow scan).

Key Technologies and Advantages of CarbonData

- Quick query response: CarbonData features high-performance query. The query speed of CarbonData is 10 times of that of Spark SQL. It uses dedicated data formats and applies multiple index technologies and multiple push-down optimizations, providing quick response to TB-level data queries.
- Efficient data compression: CarbonData compresses data by combining the lightweight and heavyweight compression algorithms. This significantly saves 60% to 80% data storage space and the hardware storage cost.

3.1.2 Main Specifications of CarbonData

Main Specifications of CarbonData

Table 3-1 Main Specifications of CarbonData

Entity	Tested Value	Test Environment
Number of tables	10000	3 nodes. 4 vCPUs and 20 GB memory for each executor. Driver memory: 5 GB, 3 executors. Total columns: 107 String: 75 Int: 13 BigInt: 7 Timestamp: 6 Double: 6
Number of table columns	2000	3 nodes. 4 vCPUs and 20 GB memory for each executor. Driver memory: 5 GB, 3 executors.
Maximum size of a raw CSV file	200 GB	17 cluster nodes. 150 GB memory and 25 vCPUs for each executor. Driver memory: 10 GB, 17 executors.

Entity	Tested Value	Test Environment
Number of CSV files in each folder	100 folders. Each folder has 10 files. The size of each file is 50 MB.	3 nodes. 4 vCPUs and 20 GB memory for each executor. Driver memory: 5 GB, 3 executors.
Number of load folders	10000	3 nodes. 4 vCPUs and 20 GB memory for each executor. Driver memory: 5 GB, 3 executors.

The memory required for data loading depends on the following factors:

- Number of columns
- Column values
- Concurrency (configured using **carbon.number.of.cores.while.loading**)
- Sort size in memory (configured using **carbon.sort.size**)
- Intermediate cache (configured using **carbon.graph.rowset.size**)

Data loading of an 8 GB CSV file that contains 10 million records and 300 columns with each row size being about 0.8 KB requires about 10 GB executor memory. That is, set **carbon.sort.size** to **100000** and retain the default values for other parameters.

Table Specifications

Table 3-2 Table specifications

Entity	Tested Value
Number of secondary index tables	10
Number of composite columns in a secondary index table	5
Length of column name in a secondary index table (unit: character)	120
Length of a secondary index table name (unit: character)	120
Cumulative length of all secondary index table names + column names in an index table* (unit: character)	3800**

 NOTE

- * Characters of column names in an index table refer to the upper limit allowed by Hive or the upper limit of available resources.
- ** Secondary index tables are registered using Hive and stored in HiveSERDEPROPERTIES in JSON format. The value of SERDEPROPERTIES supported by Hive can contain a maximum of 4,000 characters and cannot be changed.

3.2 Configuration Reference

This section provides the details of all the configurations required for the CarbonData System.

Table 3-3 System configurations in **carbon.properties**

Parameter	Default Value	Description
carbon.ddl.base.hdfs.url	hdfs://hacluster/opt/data	<p>HDFS relative path from the HDFS base path, which is configured in fs.defaultFS. The path configured in carbon.ddl.base.hdfs.url will be appended to the HDFS path configured in fs.defaultFS. If this path is configured, you do not need to pass the complete path while dataload.</p> <p>For example, if the absolute path of the CSV file is hdfs://10.18.101.155:54310/data/cnbc/2016/xyz.csv, the path hdfs://10.18.101.155:54310 will come from property fs.defaultFS and you can configure /data/cnbc/ as carbon.ddl.base.hdfs.url.</p> <p>During data loading, you can specify the CSV path as /2016/xyz.csv.</p>
carbon.badRecords.location	-	Storage path of bad records. This path is an HDFS path. The default value is Null . If bad records logging or bad records operation redirection is enabled, the path must be configured by the user.
carbon.badRecords.action	fail	<p>The following are four types of actions for bad records:</p> <p>FORCE: Data is automatically corrected by storing the bad records as NULL.</p> <p>REDIRECT: Bad records are written to the raw CSV instead of being loaded.</p> <p>IGNORE: Bad records are neither loaded nor written to the raw CSV.</p> <p>FAIL: Data loading fails if any bad records are found.</p>

Parameter	Default Value	Description
carbon.update.sync.folder	/tmp/ carbodata	Specifies the modifiedTime.mdt file path. You can set it to an existing path or a new path. NOTE If you set this parameter to an existing path, ensure that all users can access the path and the path has the 777 permission.

Table 3-4 Performance configurations in **carbon.properties**

Parameter	Default Value	Description
Data Loading Configuration		
carbon.sort.file.write.buffer.size	16384	CarbonData sorts data and writes it to a temporary file to limit memory usage. This parameter controls the size of the buffer used for reading and writing temporary files. The unit is bytes. The value ranges from 10240 to 10485760.
carbon.graph.rowset.size	100,000	Rowset size exchanged in data loading graph steps. The value ranges from 500 to 1,000,000.
carbon.number.of.cores.while.loading	6	Number of cores used during data loading. The greater the number of cores, the better the compaction performance. If the CPU resources are sufficient, you can increase the value of this parameter.
carbon.sort.size	500000	Number of records to be sorted
carbon.enableXXHash	true	Hashmap algorithm used for hashkey calculation
carbon.number.of.cores.block.sort	7	Number of cores used for sorting blocks during data loading
carbon.max.driver.lru.cache.size	-1	Maximum size of LRU caching for data loading at the driver side. The unit is MB. The default value is -1 , indicating that there is no memory limit for the caching. Only integer values greater than 0 are accepted.
carbon.max.executor.lru.cache.size	-1	Maximum size of LRU caching for data loading at the executor side. The unit is MB. The default value is -1 , indicating that there is no memory limit for the caching. Only integer values greater than 0 are accepted. If this parameter is not configured, the value of carbon.max.driver.lru.cache.size is used.

Parameter	Default Value	Description
carbon.merge.sort.prefetch	true	Whether to enable prefetch of data during merge sort while reading data from sorted temp files in the process of data loading
carbon.update.persist.enable	true	Configuration to enable the dataset of RDD/dataframe to persist data. Enabling this will reduce the execution time of UPDATE operation.
enable.unsafe.sort	true	Whether to use unsafe sort during data loading. Unsafe sort reduces the garbage collection during data load operation, resulting in better performance. The default value is true , indicating that unsafe sort is enabled.
enable.offheap.sort	true	Whether to use off-heap memory for sorting of data during data loading
offheap.sort.chunk.size.inmb	64	Size of data chunks to be sorted, in MB. The value ranges from 1 to 1024.
carbon.unsafe.working.memory.in.mb	512	<p>Size of the unsafe working memory. This will be used for sorting data and storing column pages. The unit is MB.</p> <p>Memory required for data loading: carbon.number.of.cores.while.loading [default value is 6] x Number of tables to load in parallel x offheap.sort.chunk.size.inmb [default value is 64 MB] + carbon.blockletgroup.size.in.mb [default value is 64 MB] + Current compaction ratio [64 MB/3.5]) = Around 900 MB per table</p> <p>Memory required for data query: (SPARK_EXECUTOR_INSTANCES. [default value is 2] x (carbon.blockletgroup.size.in.mb [default value: 64 MB] + carbon.blockletgroup.size.in.mb [default value = 64 MB x 3.5) x Number of cores per executor [default value: 1]) = ~ 600 MB</p>
carbon.sort.intermediate.memory.storage.size.in.mb	512	Size of the intermediate sort data to be kept in the memory. Once the specified value is reached, the system writes data to the disk. The unit is MB.

Parameter	Default Value	Description
sort.inmemory.size.inmb	1024	<p>Size of the intermediate sort data to be kept in the memory. Once the specified value is reached, the system writes data to the disk. The unit is MB.</p> <p>If carbon.unsafe.working.memory.in.mb and carbon.sort.inmemory.storage.size.in.mb are configured, you do not need to set this parameter. If this parameter has been configured, 20% of the memory is used for working memory carbon.unsafe.working.memory.in.mb, and 80% is used for sort storage memory carbon.sort.inmemory.storage.size.in.mb.</p> <p>NOTE The value of spark.yarn.executor.memoryOverhead configured for Spark must be greater than the value of sort.inmemory.size.inmb configured for CarbonData. Otherwise, Yarn might stop the executor if off-heap access exceeds the configured executor memory.</p>
carbon.blockletgroup.size.in.mb	64	<p>The data is read as a group of blocklets which are called blocklet groups. This parameter specifies the size of each blocklet group. Higher value results in better sequential I/O access.</p> <p>The minimum value is 16 MB. Any value less than 16 MB will be reset to the default value (64 MB).</p> <p>The unit is MB.</p>
enable.inmemory.merge.sort	false	Whether to enable inmemorymerge sort .
use.offheap.in.query.processing	true	Whether to enable offheap in query processing.
carbon.load.sort.scope	local_sort	Sort scope for the load operation. There are two types of sort: batch_sort and local_sort . If batch_sort is selected, the loading performance is improved but the query performance is reduced.
carbon.batch.sort.size.inmb	-	<p>Size of data to be considered for batch sorting during data loading. The recommended value is less than 45% of the total sort data. The unit is MB.</p> <p>NOTE If this parameter is not set, its value is about 45% of the value of sort.inmemory.size.inmb by default.</p>
enable.unsafe.columnpage	true	Whether to keep page data in heap memory during data loading or query to prevent garbage collection bottleneck.

Parameter	Default Value	Description
carbon.use.local.dir	false	Whether to use Yarn local directories for multi-disk data loading. If this parameter is set to true , Yarn local directories are used to load multi-disk data to improve data loading performance.
carbon.use.multiple.temp.dir	false	Whether to use multiple temporary directories for storing temporary files to improve data loading performance.
carbon.load.datamaps.parallel.db_name.table_name	N/A	The value can be true or false . You can set the database name and table name to improve the first query performance of the table.
Compaction Configuration		
carbon.number.of.cores.while.compacting	2	Number of cores to be used while compacting data. The greater the number of cores, the better the compaction performance. If the CPU resources are sufficient, you can increase the value of this parameter.
carbon.compaction.level.threshold	4,3	This configuration is for minor compaction which decides how many segments to be merged. For example, if this parameter is set to 2,3 , minor compaction is triggered every two segments. 3 is the number of level 1 compacted segments which is further compacted to new segment. The value ranges from 0 to 100.
carbon.major.compaction.size	1024	Major compaction size. Sum of the segments which is below this threshold will be merged. The unit is MB.
carbon.horizontal.compaction.enable	true	Whether to enable/disable horizontal compaction. After every DELETE and UPDATE statement, horizontal compaction may occur in case the incremental (DELETE/ UPDATE) files becomes more than specified threshold. By default, this parameter is set to true . You can set this parameter to false to disable horizontal compaction.
carbon.horizontal.update.compaction.threshold	1	Threshold limit on number of UPDATE delta files within a segment. In case the number of delta files goes beyond the threshold, the UPDATE delta files within the segment becomes eligible for horizontal compaction and are compacted into single UPDATE delta file. By default, this parameter is set to 1 . The value ranges from 1 to 10000 .

Parameter	Default Value	Description
carbon.horizontal.delete.compaction.threshold	1	Threshold limit on number of DELETE incremental files within a block of a segment. In case the number of incremental files goes beyond the threshold, the DELETE incremental files for the particular block of the segment becomes eligible for horizontal compaction and are compacted into single DELETE incremental file. By default, this parameter is set to 1 . The value ranges from 1 to 10000 .
Query Configuration		
carbon.number.of.cores	4	Number of cores to be used during query
carbon.limit.block.distribution.enable	false	Whether to enable the CarbonData distribution for limit query. The default value is false , indicating that block distribution is disabled for query statements that contain the keyword limit. For details about how to optimize this parameter, see Configurations for Performance Tuning .
carbon.custom.block.distribution	false	Whether to enable Spark or CarbonData block distribution. By default, the value is false , indicating that Spark block distribution is enabled. To enable CarbonData block distribution, change the value to true .
carbon.infilter.subquery.pushdown.enable	false	If this is set to true and a Select query is triggered in the filter with subquery, the subquery is executed and the output is broadcast as IN filter to the left table. Otherwise, SortMergeSemiJoin is executed. You are advised to set this to true when IN filter subquery does not return too many records. For example, when the IN sub-sentence query returns 10,000 or fewer records, enabling this parameter will give the query results faster. Example: <i>select * from flow_carbon_256b where cus_no in (select cus_no from flow_carbon_256b where dt>='20260101' and dt<='20260701' and txn_bk='tk_1' and txn_br='tr_1') limit 1000;</i>
carbon.scheduler.minRegisteredResourcesRatio	0.8	Minimum resource (executor) ratio needed for starting the block distribution. The default value is 0.8 , indicating that 80% of the requested resources are allocated for starting block distribution.
carbon.dynamicAllocation.schedulerTimeout	5	Maximum time that the scheduler waits for executors to be active. The default value is 5 seconds, and the maximum value is 15 seconds.

Parameter	Default Value	Description
enable.unsafe.in.query.processing	true	Whether to use unsafe sort during query. Unsafe sort reduces the garbage collection during query, resulting in better performance. The default value is true , indicating that unsafe sort is enabled.
carbon.enable.vector.reader	true	Whether to enable vector processing for result collection to improve query performance
carbon.query.show.datamaps	true	SHOW TABLES lists all tables including the primary table and datamaps. To filter out the datamaps, set this parameter to false .
Secondary Index Configuration		
carbon.secondary.index.creation.threads	1	Number of threads to concurrently process segments during secondary index creation. This property helps fine-tuning the system when there are a lot of segments in a table. The value ranges from 1 to 50.
carbon.secondary.index.lookup.partialstring	true	<ul style="list-style-type: none"> When the parameter value is true, it includes indexes started with, ended with, and contained. When the parameter value is false, it includes only secondary indexes started with.
carbon.secondary.index.segment.merge	true	<p>Enabling this property merges .carbondata files inside the secondary index segment. The merging will happen after the load operation. That is, at the end of the secondary index table load, small files are checked and merged.</p> <p>NOTE Table Block Size is used as the size threshold for merging small files.</p>

Table 3-5 Other configurations in **carbon.properties**

Parameter	Default Value	Description
Data Loading Configuration		

Parameter	Default Value	Description
carbon.lock.type	HDFSLOCK	Type of lock to be acquired during concurrent operations on a table. There are following types of lock implementation: <ul style="list-style-type: none"> • LOCALLOCK: Lock is created on local file system as a file. This lock is useful when only one Spark driver (or JDBCServer) runs on a machine. • HDFSLOCK: Lock is created on HDFS file system as a file. This lock is useful when multiple Spark applications are running and no ZooKeeper is running on a cluster.
carbon.sort.intermediate.files.limit	20	Minimum number of intermediate files. After intermediate files are generated, sort and merge the files. For details about how to optimize this parameter, see Configurations for Performance Tuning .
carbon.csv.read.buffer.size.byte	1048576	Size of CSV reading buffer
carbon.merge.sort.reader.thread	3	Maximum number of threads used for reading intermediate files for final merging.
carbon.concurrent.lock.retries	100	Maximum number of retries used to obtain the concurrent operation lock. This parameter is used for concurrent loading.
carbon.concurrent.lock.retry.timeout.sec	1	Interval between the retries to obtain the lock for concurrent operations.
carbon.lock.retries	3	Maximum number of retries to obtain the lock for any operations other than import.
carbon.lock.retry.timeout.sec	5	Interval between the retries to obtain the lock for any operation other than import.
carbon.tempstore.location	/opt/Carbon/TempStoreLoc	Temporary storage location. By default, the System.getProperty("java.io.tmpdir") method is used to obtain the value. For details about how to optimize this parameter, see the description of carbon.use.local.dir in Configurations for Performance Tuning .
carbon.load.log.counter	500000	Data loading records count in logs

Parameter	Default Value	Description
SERIALIZATION_NULL_FORMAT	\N	Value to be replaced with NULL
carbon.skip.empty.line	false	Setting this property will ignore the empty lines in the CSV file during data loading.
carbon.load.datamaps.parallel	false	Whether to enable parallel datamap loading for all tables in all sessions. This property will improve the time to load datamaps into memory by distributing the job among executors, thus improving query performance.
Merging Configuration		
carbon.numberof.preserve.segments	0	<p>If you want to preserve some number of segments from being compacted, then you can set this configuration.</p> <p>For example, if carbon.numberof.preserve.segments is set to 2, the latest two segments will always be excluded from the compaction.</p> <p>No segments will be preserved by default.</p>
carbon.allowed.compaction.days	0	<p>This configuration is used to control on the number of recent segments that needs to be merged.</p> <p>For example, if this parameter is set to 2, the segments which are loaded in the time frame of past 2 days only will get merged. Segments which are loaded earlier than 2 days will not be merged.</p> <p>This configuration is disabled by default.</p>
carbon.enable.auto.load.merge	false	Whether to enable compaction along with data loading.
carbon.merge.index.in.segment	true	This configuration enables to merge all the CarbonIndex files (.carbonindex) into a single MergeIndex file (.carbonindexmerge) upon data loading completion. This significantly reduces the delay in serving the first query.
Query Configuration		
max.query.execution.time	60	<p>Maximum time allowed for one query to be executed.</p> <p>The unit is minute.</p>

Parameter	Default Value	Description
carbon.enableMinMax	true	MinMax is used to improve query performance. You can set this to false to disable this function.
carbon.lease.recovery.retry.count	5	Maximum number of attempts that need to be made for recovering a lease on a file. Minimum value: 1 Maximum value: 50
carbon.lease.recovery.retry.interval	1000 (ms)	Interval or pause time after a lease recovery attempt is made on a file. Minimum value: 1000 (ms) Maximum value: 10000 (ms)

Table 3-6 Spark configuration reference in **spark-defaults.conf**

Parameter	Default Value	Description
spark.driver.memory	4G	Memory to be used for the driver process. SparkContext has been initialized. NOTE In client mode, do not use SparkConf to set this parameter in the application because the driver JVM has been started. To configure this parameter, configure it in the --driver-memory command-line option or in the default property file.
spark.executor.memory	4 GB	Memory to be used for each executor process.
spark.sql.crossJoin.enabled	true	If the query contains a cross join, enable this property so that no error is thrown. In this case, you can use a cross join instead of a join for better performance.

Configure the following parameters in the **spark-defaults.conf** file on the Spark driver.

- In spark-sql mode:

Table 3-7 Parameter description

Parameter	Value	Description
spark.driver.extraJavaOptions	-Dlog4j.configuration=file:/opt/client/Spark2x/spark/conf/log4j.properties - Djetty.version=x.y.z - Dzookeeper.server.principal=zookeeper/hadoop.<System domain name> - Djava.security.krb5.conf=/opt/client/KrbClient/kerberos/var/krb5kdc/krb5.conf - Djava.security.auth.login.config=/opt/client/Spark2x/spark/conf/jaas.conf - Dorg.xerial.snappy.tmpdir=/opt/client/Spark2x/tmp - Dcarbon.properties.filepath=/opt/client/Spark2x/spark/conf/carbon.properties - Djava.io.tmpdir=/opt/client/Spark2x/tmp	The default value /opt/client/Spark2x/spark indicates CLIENT_HOME of the client and is added to the end of the value of spark.driver.extraJavaOptions . This parameter is used to specify the path of the carbon.properties file in Driver. NOTE Spaces next to equal marks (=) are not allowed.
spark.sql.session.state.builder	org.apache.spark.sql.hive.HiveACLSessionStateBuilder	Session state constructor.
spark.carbon.sqlastbuilder.classname	org.apache.spark.sql.hive.CarbonInternalSqlAstBuilder	AST constructor.
spark.sql.catalog.class	org.apache.spark.sql.hive.HiveACLExternalCatalog	Hive External catalog to be used. This parameter is mandatory if Spark ACL is enabled.
spark.sql.hive.implementation	org.apache.spark.sql.hive.HiveACLClientImpl	How to call the Hive client. This parameter is mandatory if Spark ACL is enabled.
spark.sql.hiveClient.isolation.enabled	false	This parameter is mandatory if Spark ACL is enabled.

- In JDBCServer mode:

Table 3-8 Parameter description

Parameter	Value	Description
spark.driver.extraJavaOptions	-Xloggc:\${SPARK_LOG_DIR}/indexserver-omm-%p-gc.log -XX: +PrintGCDetails -XX:- OmitStackTracenFastThrow -XX: +PrintGCTimeStamps -XX: +PrintGCDateStamps - XX:MaxDirectMemorySize=512M - XX:MaxMetaspaceSize=512M -XX: +UseGCLogFileRotation - XX:NumberOfGCLogFiles=20 - XX:GCLogFileSize=10M - XX:OnOutOfMemoryError='kill -9 %p' - Djetty.version=x.y.z - Dorg.xerial.snappy.tmpdir=\${BIGDATA_HOME}/tmp/spark2x/JDBCServer/snappy_tmp - Djava.io.tmpdir=\${BIGDATA_HOME}/tmp/spark2x/JDBCServer/io_tmp - Dcarbon.properties.filepath=\${SPARK_CONF_DIR}/carbon.properties - Djdk.tls.ephemeralDHKeySize=20	The default value <code>\${SPARK_CONF_DIR}</code> depends on a specific cluster and is added to the end of the value of the <code>spark.driver.extraJavaOptions</code> parameter. This parameter is used to specify the path of the <code>carbon.properties</code> file in Driver. NOTE Spaces next to equal marks (=) are not allowed.

Parameter	Value	Description
	48 - Dspark.ssl.keyStore=\${SPARK_CONF_DIR}/child.keystore#{java_stack_prefer}	
spark.sql.session.state.builder	org.apache.spark.sql.hive.FIHiveACLSessionStateBuilder	Session state constructor.
spark.carbon.sqlastbuilder.classname	org.apache.spark.sql.hive.CarbonInternalSqlAstBuilder	AST constructor.
spark.sql.catalog.class	org.apache.spark.sql.hive.HiveACLExternalCatalog	Hive External catalog to be used. This parameter is mandatory if Spark ACL is enabled.
spark.sql.hive.implementation	org.apache.spark.sql.hive.HiveACLClientImpl	How to call the Hive client. This parameter is mandatory if Spark ACL is enabled.
spark.sql.hiveClient.isolation.enabled	false	This parameter is mandatory if Spark ACL is enabled.

3.3 CarbonData Operation Guide

3.3.1 CarbonData Quick Start

This section describes how to create CarbonData tables, load data, and query data. This quick start provides operations based on the Spark Beeline client. If you want to use Spark shell, wrap the queries with **spark.sql()**.

The following describes how to load data from a CSV file to a CarbonData table.

Table 3-9 CarbonData Quick Start

Operation	Description
Preparing a CSV File	Prepare the CSV file to be loaded to the CarbonData Table.

Operation	Description
Connecting to CarbonData	Connect to CarbonData before performing any operations on CarbonData.
Creating a CarbonData Table	Create a CarbonData table to load data and perform query operations.
Loading Data to a CarbonData Table	Load data from CSV to the created table.
Querying Data from a CarbonData Table	Perform query operations such as filters and groupby.

Preparing a CSV File

- Prepare a CSV file named **test.csv** on the local PC. An example is as follows:

```
13418592122,1001, MAC address, 2017-10-23 15:32:30,2017-10-24 15:32:30,62.50,74.56
13418592123 1002, MAC address, 2017-10-23 16:32:30,2017-10-24 16:32:30,17.80,76.28
13418592124,1003, MAC address, 2017-10-23 17:32:30,2017-10-24 17:32:30,20.40,92.94
13418592125 1004, MAC address, 2017-10-23 18:32:30,2017-10-24 18:32:30,73.84,8.58
13418592126,1005, MAC address, 2017-10-23 19:32:30,2017-10-24 19:32:30,80.50,88.02
13418592127 1006, MAC address, 2017-10-23 20:32:30,2017-10-24 20:32:30,65.77,71.24
13418592128,1007, MAC address, 2017-10-23 21:32:30,2017-10-24 21:32:30,75.21,76.04
13418592129,1008, MAC address, 2017-10-23 22:32:30,2017-10-24 22:32:30,63.30,94.40
13418592130, 1009, MAC address, 2017-10-23 23:32:30,2017-10-24 23:32:30,95.51,50.17
13418592131,1010, MAC address, 2017-10-24 00:32:30,2017-10-25 00:32:30,39.62,99.13
```
- Use WinSCP to import the CSV file to the directory of the node where the client is installed, for example, **/opt**.
- Log in to FusionInsight Manager and choose **System**. In the navigation pane on the left, choose **Permission > User**, click **Create** to create human-machine user **sparkuser**, and add the user to user groups hadoop (primary group) and hive.
- Run the following commands to go to the client installation directory, load environment variables, and authenticate the user.

```
cd /Client installation directory
source ./bigdata_env
source ./Spark2x/component_env
kinit sparkuser
```
- Run the following command to upload the CSV file to the **/data** directory of the HDFS.

```
hdfs dfs -put /opt/test.csv /data/
```

Connecting to CarbonData

- Use Spark SQL or Spark shell to connect to Spark and run Spark SQL commands.
- Run the following commands to start the JDBCServer and use a JDBC client (for example, Spark Beeline) to connect to the JDBCServer.

```
cd ./Spark2x/spark/bin
./spark-beeline
```

Creating a CarbonData Table

After connecting Spark Beeline with the JDBCServer, create a CarbonData table to load data and perform query operations. Run the following commands to create a simple table:

```
create table x1 (imei string, deviceInformationId int, mac string, productdate
timestamp, updatetime timestamp, gamePointId double, contractNumber
double) STORED AS carbondata TBLPROPERTIES
('SORT_COLUMNS'='imei,mac');
```

The command output is as follows:

```
+-----+
| Result |
+-----+
+-----+
No rows selected (1.093 seconds)
```

Loading Data to a CarbonData Table

After you have created a CarbonData table, you can load the data from CSV to the created table.

Run the following command with required parameters to load data from CSV. The column names of the CarbonData table must match the column names of the CSV file.

```
LOAD DATA inpath 'hdfs://hacluster/data/test.csv' into table x1
options('DELIMITER'=',', 'QUOTECHAR'='', 'FILEHEADER'='imei,
deviceinformationid,mac, productdate,updatetime,
gamepointid,contractnumber');
```

test.csv is the CSV file prepared in [Preparing a CSV File](#) and **x1** is the table name.

The CSV example file is as follows:

```
13418592122,1001, MAC address, 2017-10-23 15:32:30,2017-10-24 15:32:30,62.50,74.56
13418592123,1002, MAC address, 2017-10-23 16:32:30,2017-10-24 16:32:30,17.80,76.28
13418592124,1003, MAC address, 2017-10-23 17:32:30,2017-10-24 17:32:30,20.40,92.94
13418592125,1004, MAC address, 2017-10-23 18:32:30,2017-10-24 18:32:30,73.84,8.58
13418592126,1005, MAC address, 2017-10-23 19:32:30,2017-10-24 19:32:30,80.50,88.02
13418592127,1006, MAC address, 2017-10-23 20:32:30,2017-10-24 20:32:30,65.77,71.24
13418592128,1007, MAC address, 2017-10-23 21:32:30,2017-10-24 21:32:30,75.21,76.04
13418592129,1008, MAC address, 2017-10-23 22:32:30,2017-10-24 22:32:30,63.30,94.40
13418592130,1009, MAC address, 2017-10-23 23:32:30,2017-10-24 23:32:30,95.51,50.17
13418592131,1010, MAC address, 2017-10-24 00:32:30,2017-10-25 00:32:30,39.62,99.13
```

The command output is as follows:

```
+-----+
|Segment ID |
+-----+
|0          |
+-----+
No rows selected (3.039 seconds)
```

Querying Data from a CarbonData Table

After a CarbonData table is created and the data is loaded, you can perform query operations as required. Some query operations are provided as examples.

- **Obtaining the number of records**
Run the following command to obtain the number of records in the CarbonData table:
select count(*) from x1;
- **Querying with the groupby condition**
Run the following command to obtain the **deviceinformationid** records without repetition in the CarbonData table:
select deviceinformationid,count (distinct deviceinformationid) from x1 group by deviceinformationid;
- **Querying with Filter**
Run the following command to obtain specific **deviceinformationid** records:
select * from x1 where deviceinformationid='1010';

 **NOTE**

If the query result have other non-English characters, the columns in the query result may not be aligned. This is because characters of different languages occupy different widths.

Using CarbonData on Spark-shell

If you need to use CarbonData on a Spark-shell, you need to create a CarbonData table, load data to the CarbonData table, and query data in CarbonData as follows:

```
spark.sql("CREATE TABLE x2(imei string, deviceInformationId int, mac string, productdate timestamp, updatetime timestamp, gamePointId double, contractNumber double) STORED AS carbondata")
spark.sql("LOAD DATA inpath 'hdfs://hacluster/data/x1_without_header.csv' into table x2
options('DELIMITER=',', 'QUOTECHAR='\",'FILEHEADER'='imei, deviceinformationid,mac, productdate,updatetime, gamepointid,contractnumber')")
spark.sql("SELECT * FROM x2").show()
```

3.3.2 CarbonData Table Management

3.3.2.1 About CarbonData Table

Overview

In CarbonData, data is stored in entities called tables. CarbonData tables are similar to RDBMS tables. RDBMS data is stored in a table consisting of rows and columns. CarbonData tables store structured data, and have fixed columns and data types.

Supported Data Types

CarbonData tables support the following data types:

- Int
- String
- BigInt
- Smallint
- Char

- Varchar
- Boolean
- Decimal
- Double
- TimeStamp
- Date
- Array
- Struct
- Map

The following table describes supported data types and their respective values range.

Table 3-10 CarbonData data types

Data Type	Value Range
Int	4-byte signed integer ranging from -2,147,483,648 to 2,147,483,647. NOTE If a non-dictionary column is of the int data type, it is internally stored as the BigInt type.
String	100,000 characters NOTE If the CHAR or VARCHAR data type is used in CREATE TABLE , the two data types are automatically converted to the String data type. If a column contains more than 32,000 characters, add the column to the LONG_STRING_COLUMNS attribute of the tblproperties table during table creation.
BigInt	64-bit value ranging from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
SmallInt	-32,768 to 32,767
Char	A to Z and a to z
Varchar	A to Z, a to z, and 0 to 9
Boolean	true or false
Decimal	The default value is (10,0) and maximum value is (38,38). NOTE When query with filters, append BD to the number to achieve accurate results. For example, select * from carbon_table where num = 1234567890123456.22BD .
Double	64-bit value ranging from 4.9E-324 to 1.7976931348623157E308
TimeStamp	The default format is yyyy-MM-dd HH:mm:ss .

Data Type	Value Range
Date	The DATE data type is used to store calendar dates. The default format is yyyy-MM-DD .
Array<data_type>	N/A NOTE Currently, only two layers of complex types can be nested.
Struct<col_name: data_type COMMENT col_comment, ...>	
Map<primitive_type, data_type>	

3.3.2.2 Creating a CarbonData Table

Scenario

A CarbonData table must be created to load and query data. You can run the **Create Table** command to create a table. This command is used to create a table using custom columns.

Creating a Table with Self-Defined Columns

Users can create a table by specifying its columns and data types.

Sample command:

```
CREATE TABLE IF NOT EXISTS productdb.productSalesTable (  
productNumber Int,  
productName String,  
storeCity String,  
storeProvince String,  
productCategory String,  
productBatch String,  
saleQuantity Int,  
revenue Int)  
STORED AS carbondata  
TBLPROPERTIES (  
'table_blocksize'='128');  
)
```

The following table describes parameters of preceding commands.

Table 3-11 Parameter description

Parameter	Description
productSalesTable	Table name. The table is used to load data for analysis. The table name consists of letters, digits, and underscores (_).
productdb	Database name. The database maintains logical connections with tables stored in it to identify and manage the tables. The database name consists of letters, digits, and underscores (_).
productName storeCity storeProvince productCategory productBatch saleQuantity revenue	Columns in the table. The columns are service entities for data analysis. The column name (field name) consists of letters, digits, and underscores (_).
table_blocksize	Indicates the block size of data files used by the CarbonData table, in MB. The value ranges from 1 to 2048 . The default value is 1024 . If table_blocksize is too small, a large number of small files will be generated when data is loaded. This may affect the performance of HDFS. If table_blocksize is too large, during data query, the amount of block data that matches the index is large, and some blocks contain a large number of blocklets, affecting read concurrency and lowering query performance. You are advised to set the block size based on the data volume. For example, set the block size to 256 MB for GB-level data, 512 MB for TB-level data, and 1024 MB for PB-level data.

 **NOTE**

- Measurement of all Integer data is processed and displayed using the **BigInt** data type.
- CarbonData parses data strictly. Any data that cannot be parsed is saved as **null** in the table. For example, if the user loads the **double** value (3.14) to the **BigInt** column, the data is saved as **null**.
- The Short and Long data types used in the **Create Table** command are shown as **Smallint** and **BigInt** in the **DESCRIBE** command, respectively.
- You can run the **DESCRIBE** command to view the table data size and table index size.

Operation Result

Run the command to create a table.

3.3.2.3 Deleting a CarbonData Table

Scenario

You can run the **DROP TABLE** command to delete a table. After a CarbonData table is deleted, its metadata and loaded data are deleted together.

Procedure

Run the following command to delete a CarbonData table:

Run the following command:

```
DROP TABLE [IF EXISTS] [db_name.]table_name;
```

Once this command is executed, the table is deleted from the system. In the command, **db_name** is an optional parameter. If **db_name** is not specified, the table named **table_name** in the current database is deleted.

Example:

```
DROP TABLE productdb.productSalesTable;
```

Run the preceding command to delete the **productSalesTable** table from the **productdb** database.

Operation Result

Deletes the table specified in the command from the system. After the table is deleted, you can run the **SHOW TABLES** command to check whether the table is successfully deleted. For details, see [SHOW TABLES](#).

3.3.2.4 Modify the CarbonData Table

SET and UNSET

When the **SET** command is executed, the new properties overwrite the existing ones.

- SORT SCOPE

The following is an example of the **SET SORT SCOPE** command:

```
ALTER TABLE tablename SET TBLPROPERTIES('SORT_SCOPE'='no_sort')
```

After running the **UNSET SORT SCOPE** command, the default value **NO_SORT** is adopted.

The following is an example of the **UNSET SORT SCOPE** command:

```
ALTER TABLE tablename UNSET TBLPROPERTIES('SORT_SCOPE')
```

- SORT COLUMNS

The following is an example of the **SET SORT COLUMNS** command:

```
ALTER TABLE tablename SET TBLPROPERTIES('SORT_COLUMNS'='column1')
```

After this command is executed, the new value of **`SORT_COLUMNS`** is used. Users can adjust the **`SORT_COLUMNS`** based on the query results, but the original data is not affected. The operation does not affect the query performance of the original data segments which are not sorted by new **`SORT_COLUMNS`**.

The **`UNSET`** command is not supported, but the **`SORT_COLUMNS`** can be set to empty string instead of using the **`UNSET`** command.

`ALTER TABLE tablename SET TBLPROPERTIES('SORT_COLUMNS'='')`

 **NOTE**

- The later version will enhance custom compaction to resort the old segments.
- The value of **`SORT_COLUMNS`** cannot be modified in the streaming table.
- If the **`inverted index`** column is removed from **`SORT_COLUMNS`**, **`inverted index`** will not be created in this column. However, the old configuration of **`INVERTED_INDEX`** will be kept.

3.3.3 CarbonData Table Data Management

3.3.3.1 Loading Data

Scenario

After a CarbonData table is created, you can run the **`LOAD DATA`** command to load data to the table for query. Once data loading is triggered, data is encoded in CarbonData format and files in multi-dimensional and column-based format are compressed and copied to the HDFS path of CarbonData files for quick analysis and queries. The HDFS path can be configured in the **`carbon.properties`** file. For details, see [Configuration Reference](#).

3.3.3.2 Deleting Segments

Scenario

If you want to modify and reload the data because you have loaded wrong data into a table, or there are too many bad records, you can delete specific segments by segment ID or data loading time.

 **NOTE**

The segment deletion operation only deletes segments that are not compacted. You can run the **`CLEAN FILES`** command to clear compacted segments.

Deleting a Segment by Segment ID

Each segment has a unique ID. This segment ID can be used to delete the segment.

Step 1 Obtain the segment ID.

Command:

`SHOW SEGMENTS FOR Table dbname.tablename LIMIT number_of_loads;`

Example:

SHOW SEGMENTS FOR TABLE *carbonTable*;

Run the preceding command to show all the segments of the table named **carbonTable**.

SHOW SEGMENTS FOR TABLE *carbonTable LIMIT 2*;

Run the preceding command to show segments specified by *number_of_loads*.

The command output is as follows:

```
+-----+-----+-----+-----+-----+-----+-----+-----+
+
| ID | Status | Load Start Time | Load Time Taken | Partition | Data Size | Index Size | File Format |
+-----+-----+-----+-----+-----+-----+-----+-----+
+
| 3 | Success | 2020-09-28 22:53:26.336 | 3.726S | {} | 6.47KB | 3.30KB | columnar_v3 |
| 2 | Success | 2020-09-28 22:53:01.702 | 6.688S | {} | 6.47KB | 3.30KB | columnar_v3 |
+-----+-----+-----+-----+-----+-----+-----+-----+
+
```

 **NOTE**

The output of the **SHOW SEGMENTS** command includes ID, Status, Load Start Time, Load Time Taken, Partition, Data Size, Index Size, and File Format. The latest loading information is displayed in the first line of the command output.

Step 2 Run the following command to delete the segment after you have found the Segment ID:

Command:

DELETE FROM TABLE *tableName* **WHERE SEGMENT.ID IN (load_sequence_id1, load_sequence_id2, ...);**

Example:

DELETE FROM TABLE *carbonTable* **WHERE SEGMENT.ID IN (1,2,3);**

For details, see [DELETE SEGMENT by ID](#).

----End

Deleting a Segment by Data Loading Time

You can delete a segment based on the loading time.

Command:

DELETE FROM TABLE *db_name.table_name* **WHERE SEGMENT.STARTTIME BEFORE date_value;**

Example:

DELETE FROM TABLE *carbonTable* **WHERE SEGMENT.STARTTIME BEFORE '2017-07-01 12:07:20';**

The preceding command can be used to delete all segments before 2017-07-01 12:07:20.

For details, see [DELETE SEGMENT by DATE](#).

Result

Data of corresponding segments is deleted and is unavailable for query. You can run the **SHOW SEGMENTS** command to display the segment status and check whether the segment has been deleted.

NOTE

- Segments are not physically deleted after the execution of the **DELETE SEGMENT** command. Therefore, if you run the **SHOW SEGMENTS** command to check the status of a deleted segment, it will be marked as **Marked for Delete**. If you run the **SELECT * FROM tablename** command, the deleted segment will be excluded.
- The deleted segment will be deleted physically only when the next data loading reaches the maximum query execution duration, which is configured by the **max.query.execution.time** parameter. The default value of the parameter is 60 minutes.
- If you want to forcibly delete a physical segment file, run the **CLEAN FILES** command.

Example:

```
CLEAN FILES FOR TABLE table1;
```

This command will physically delete the segment file in the **Marked for delete** state.

If this command is executed before the time specified by **max.query.execution.time** arrives, the query may fail. **max.query.execution.time** indicates the maximum time allowed for a query, which is set in the **carbon.properties** file.

3.3.3.3 Combining Segments

Scenario

Frequent data access results in a large number of fragmented CarbonData files in the storage directory. In each data loading, data is sorted and indexing is performed. This means that an index is generated for each load. With the increase of data loading times, the number of indexes also increases. As each index works only on one loading, the performance of index is reduced. CarbonData provides loading and compression functions. In a compression process, data in each segment is combined and sorted, and multiple segments are combined into one large segment.

Prerequisites

Multiple data loadings have been performed.

Operation Description

There are three types of compaction: Minor, Major, and Custom.

- **Minor compaction:**
In minor compaction, you can specify the number of loads to be merged. If **carbon.enable.auto.load.merge** is set, minor compaction is triggered for every data load. If any segments are available to be merged, then compaction will run parallel with data load.

There are two levels in minor compaction:

- Level 1: Merging of the segments which are not yet compacted

- Level 2: Merging of the compacted segments again to form a larger segment
- Major compaction:
Multiple segments can be merged into one large segment. You can specify the compaction size so that all segments below the size will be merged. Major compaction is usually done during the off-peak time.
- Custom compaction:
In Custom compaction, you can specify the IDs of multiple segments to merge them into a large segment. The IDs of all the specified segments must exist and be valid. Otherwise, the compaction fails. Custom compaction is usually done during the off-peak time.

For details, see [ALTER TABLE COMPACTION](#).

Table 3-12 Compaction parameters

Parameter	Default Value	Application Type	Description
carbon.enable.automerge	false	Minor	Whether to enable compaction along with data loading. true: Compaction is automatically triggered when data is loaded. false: Compaction is not triggered when data is loaded.
carbon.compaction.level.threshold	4,3	Minor	This configuration is for minor compaction which decides how many segments to be merged. For example, if this parameter is set to 2,3 , minor compaction is triggered every two segments and segments form a single level 1 compacted segment. When the number of compacted level 1 segments reach 3, compaction is triggered again to merge them to form a single level 2 segment. The compaction policy depends on the actual data size and available resources. The value ranges from 0 to 100.

Parameter	Default Value	Application Type	Description
carbon.major.compaction.size	1024 MB	Major	<p>The major compaction size can be configured using this parameter. Sum of the segments which is below this threshold will be merged.</p> <p>For example, if this parameter is set to 1024 MB, and there are five segments whose sizes are 300 MB, 400 MB, 500 MB, 200 MB, and 100 MB used for major compaction, only segments whose total size is less than this threshold are compacted. In this example, only the segments whose sizes are 300 MB, 400 MB, 200 MB, and 100 MB are compacted.</p>
carbon.numberof.preserve.segments	0	Minor/Major	<p>If you want to preserve some number of segments from being compacted, then you can set this configuration.</p> <p>For example, if carbon.numberof.preserve.segments is set to 2, the latest two segments will always be excluded from the compaction.</p> <p>By default, no segments are reserved.</p>
carbon.allowed.compaction.days	0	Minor/Major	<p>This configuration is used to control on the number of recent segments that needs to be compacted.</p> <p>For example, if this parameter is set to 2, the segments which are loaded in the time frame of past 2 days only will get merged. Segments which are loaded earlier than 2 days will not be merged.</p> <p>This configuration is disabled by default.</p>
carbon.numberof.cores.while.compacting	2	Minor/Major	<p>Number of cores to be used while compacting data. The greater the number of cores, the better the compaction performance. If the CPU resources are sufficient, you can increase the value of this parameter.</p>

Parameter	Default Value	Application Type	Description
carbon.merge.index.in.segment	true	SEGMENT_INDEX	If this parameter is set to true , all the Carbon index (.carbonindex) files in a segment will be merged into a single Index (.carbonindexmerge) file. This enhances the first query performance.

Reference

You are advised not to perform minor compaction on historical data. For details, see [How to Avoid Minor Compaction for Historical Data?](#).

3.3.4 CarbonData Data Migration

Scenario

If you want to rapidly migrate CarbonData data from a cluster to another one, you can use the CarbonData backup and restoration commands. This method does not require data import in the target cluster, reducing required migration time.

Prerequisites

The Spark2x client has been installed in a directory, for example, **/opt/ficlient**, in two clusters. The source cluster is cluster A, and the target cluster is cluster B.

Procedure

Step 1 Log in to the node where the client is installed in cluster A as a client installation user.

Step 2 Run the following commands to configure environment variables:

```
source /opt/ficlient/bigdata_env
```

```
source /opt/ficlient/Spark2x/component_env
```

Step 3 If the cluster is in security mode, run the following command to authenticate the user. In normal mode, skip user authentication.

```
kinit carbondatauser
```

carbondatauser indicates the user of the original data. That is, the user has the read and write permissions for the tables.

NOTE

You must add the user to the **hadoop** (primary group) and **hive** groups, and associate it with the **System_administrator** role.

Step 4 Run the following command to connect to the database and check the location for storing table data on HDFS:

```
spark-beeline
```

```
desc formatted Name of the table containing the original data;
```

Location in the displayed information indicates the directory where the data file resides.

Step 5 Log in to the node where the client is installed in cluster B as a client installation user and configure the environment variables:

```
source /opt/ficlient/bigdata_env
```

```
source /opt/ficlient/Spark2x/component_env
```

Step 6 If the cluster is in security mode, run the following command to authenticate the user. In normal mode, skip user authentication.

```
kinit carbondatauser2
```

carbondatauser2 indicates the user that uploads data.

 **NOTE**

You must add the user to the **hadoop** (primary group) and **hive** groups, and associate it with the **System_administrator** role.

Step 7 Run the **spark-beeline** command to connect to the database.

Step 8 Does the database that maps to the original data exist?

- If yes, go to [Step 9](#).
- If no, run the **create database** *Database name* command to create a database with the same name as that maps to the original data and go to [Step 9](#).

Step 9 Copy the original data from the HDFS directory in cluster A to the HDFS directory in cluster B.

When uploading data in cluster B, ensure that the upload directory has the directories with the same names as the database and table in the original directory and the upload user has the permission to write data to the upload directory. After the data is uploaded, the user has the permission to read and write the data.

For example, if the original data is stored in **/user/carboncadauser/warehouse/db1/tb1**, the data can be stored in **/user/carbondatauser2/warehouse/db1/tb1** in the new cluster.

1. Run the following command to download the original data to the **/opt/backup** directory of cluster A:

```
hdfs dfs -get /user/carboncadauser/warehouse/db1/tb1 /opt/backup
```
2. Run the following command to copy the original data of cluster A to the **/opt/backup** directory on the client node of cluster B.

```
scp /opt/backup root@IP address of the client node of cluster B:/opt/backup
```
3. Run the following command to upload the data copied to cluster B to HDFS:

```
hdfs dfs -put /opt/backup /user/carbondatauser2/warehouse/db1/tb1
```

Step 10 In the client environment of cluster B, run the following command to generate the metadata associated with the table corresponding to the original data in Hive:

```
REFRESH TABLE $dbName.$tbName;
```

\$dbName indicates the database name, and *\$tbName* indicates the table name.

Step 11 If the original table contains an index table, perform [Step 9](#) and [Step 10](#) to migrate the index table directory from cluster A to cluster B.

Step 12 Run the following command to register an index table for the CarbonData table (skip this step if no index table is created for the original table):

```
REGISTER INDEX TABLE $tableName ON $maintable;
```

\$tableName indicates the index table name, and *\$maintable* indicates the table name.

----End

3.3.5 Migrating Data on CarbonData from Spark 1.5 to Spark2x

Migration Solution Overview

This migration guides you to migrate the CarbonData table data of Spark 1.5 to that of Spark2x.

NOTE

Before performing this operation, you need to stop the data import service of the CarbonData table in Spark 1.5 and migrate data to the CarbonData table of Spark2x at a time. After the migration is complete, use Spark2x to perform service operations.

Migration roadmap:

1. Use Spark 1.5 to migrate historical data to the intermediate table.
2. Use Spark2x to migrate data from the intermediate table to the target table and change the target table name to the original table name.
3. After the migration is complete, use Spark2x to operate data in the CarbonData table.

Migration Solution and Commands

Migrating Historical Data

Step 1 Stop the CarbonData data import service, use spark-beeline of Spark 1.5 to view the ID and time of the latest segment in the CarbonData table, and record the segment ID.

```
show segments for table dbname.tablename;
```

Step 2 Run spark-beeline of Spark 1.5 as the user who has created the original CarbonData table to create an intermediate table in ORC or Parquet format. Then import the data in the original CarbonData table to the intermediate table. After the import is complete, the services of the CarbonData table can be restored.

Create an ORC table.

```
CREATE TABLE dbname.mid_tablename_orc STORED AS ORC as select * from  
dbname.tablename;
```

Create a Parquet table.

```
CREATE TABLE dbname.mid_tablename_parq STORED AS PARQUET as select *  
from dbname.tablename;
```

In the preceding command, **dbname** indicates the database name and **tablename** indicates the name of the original CarbonData table.

- Step 3** Run spark-beeline of Spark2x as the user who has created the original CarbonData table. Run the table creation statement of the old table to create a CarbonData table.

 **NOTE**

In the statement for creating a new table, the field sequence and type must be the same as those of the old table. In this way, the index column structure of the old table can be retained, which helps avoid errors caused by the use of **select *** statement during data insertion.

Run the spark-beeline command of Spark 1.5 to view the table creation statement of the old table: **SHOW CREATE TABLE dbname.tablename;**

Create a CarbonData table named **dbname.new_tablename**.

- Step 4** Run spark-beeline of Spark2x as the user who has created the original CarbonData table to load the intermediate table data in ORC (or PARQUET) format created in [Step 2](#) to the new table created in [Step 3](#). This step may take a long time (about 2 hours for 200 GB data). The following uses the ORC intermediate table as an example to describe the command for loading data:

```
insert into dbname.new_tablename select *  
from dbname.mid_tablename_orc;
```

- Step 5** Run spark-beeline of Spark2x as the user who has created the original CarbonData table to query and verify the data in the new table. If the data is correct, change the name of the original CarbonData table and then change the name of the new CarbonData table to the name of the original one.

```
ALTER TABLE dbname.tablename RENAME TO dbname.old_tablename;
```

```
ALTER TABLE dbname.new_tablename RENAME TO dbname.tablename;
```

- Step 6** Complete the migration. In this case, you can use Spark2x to query the new table and rebuild the secondary index.

----End

3.4 CarbonData Performance Tuning

3.4.1 Tuning Guidelines

Query Performance Tuning

There are various parameters that can be tuned to improve the query performance in CarbonData. Most of the parameters focus on increasing the parallelism in processing and optimizing system resource usage.

- Spark executor count: Executors are basic entities of parallelism in Spark. Raising the number of executors can increase the amount of parallelism in the cluster. For details about how to configure the number of executors, see the Spark documentation.
- Executor core: The number of concurrent tasks that an executor can run are controlled in each executor. Increasing the number of executor cores will add more concurrent processing tasks to improve performance.
- HDFS block size: CarbonData assigns query tasks by allocating different blocks to different executors for processing. HDFS block is the partition unit. CarbonData maintains a global block level index in Spark driver, which helps to reduce the quantity of blocks that need to be scanned for a query. Higher block size means higher I/O efficiency and lower global index efficiency. Reversely, lower block size means lower I/O efficiency, higher global index efficiency, and greater memory consumption.
- Number of scanner threads: Scanner threads control the number of parallel data blocks that are processed by each task. By increasing the number of scanner threads, you can increase the number of data blocks that are processed in parallel to improve performance. The **carbon.number.of.cores** parameter in the **carbon.properties** file is used to configure the number of scanner threads. For example, **carbon.number.of.cores = 4**.
- B-Tree caching: The cache memory can be optimized using the B-Tree least recently used (LRU) caching. In the driver, the B-Tree LRU caching configuration helps free up the cache by releasing table segments which are not accessed or not used. Similarly, in the executor, the B-Tree LRU caching configuration will help release table blocks that are not accessed or used. For details, see the description of **carbon.max.driver.lru.cache.size** and **carbon.max.executor.lru.cache.size** in [Table 3-4](#).

CarbonData Query Process

When CarbonData receives a table query task, for example query for table A, the index data of table A will be loaded to the memory for the query process. When CarbonData receives a query task for table A again, the system does not need to load the index data of table A.

When a query is performed in CarbonData, the query task is divided into several scan tasks, namely, task splitting based on HDFS blocks. Scan tasks are executed by executors on the cluster. Tasks can run in parallel, partially parallel, or in sequence, depending on the number of executors and configured number of executor cores.

Some parts of a query task can be processed at the individual task level, such as **select** and **filter**. Some parts of a query task can be processed at the individual task level, such as **group-by**, **count**, and **distinct count**.

Some operations cannot be performed at the task level, such as **Having Clause** (filter after grouping) and **sort**. Operations which cannot be performed at the task level or can be only performed partially at the task level require data (partial results) transmission across executors on the cluster. The transmission operation is called shuffle.

The more the tasks are, the more data needs to be shuffled. This affects query performance.

The number of tasks is depending on the number of HDFS blocks and the number of blocks is depending on the size of each block. You are advised to configure proper HDFS block size to achieve a balance among increased parallelism, the amount of data to be shuffled, and the size of aggregate tables.

Relationship Between Splits and Executors

If the number of splits is less than or equal to the executor count multiplied by the executor core count, the tasks are run in parallel. Otherwise, some tasks can start only after other tasks are complete. Therefore, ensure that the executor count multiplied by executor cores is greater than or equal to the number of splits. In addition, make sure that there are sufficient splits so that a query task can be divided into sufficient subtasks to ensure concurrency.

Configuring Scanner Threads

The scanner threads property decides the number of data blocks to be processed. If there are too many data blocks, a large number of small data blocks will be generated, affecting performance. If there are few data blocks, the parallelism is poor and the performance is affected. Therefore, when determining the number of scanner threads, you are advised to consider the average data size within a partition and select a value that makes the data block not small. Based on experience, you are advised to divide a single block size (unit: MB) by 250 and use the result as the number of scanner threads.

The number of actual available vCPUs is an important factor to consider when you want to increase the parallelism. The number of vCPUs that conduct parallel computation must not exceed 75% to 80% of actual vCPUs.

The number of vCPUs is approximately equal to:

Number of parallel tasks x Number of scanner threads. Number of parallel tasks is the smaller value of number of splits or executor count x executor cores.

Data Loading Performance Tuning

Tuning of data loading performance is different from that of query performance. Similar to query performance, data loading performance depends on the amount of parallelism that can be achieved. In case of data loading, the number of worker threads decides the unit of parallelism. Therefore, more executors mean more executor cores and better data loading performance.

To achieve better performance, you can configure the following parameters in HDFS.

Table 3-13 HDFS configuration

Parameter	Recommended Value
dfs.datanode.drop.cache.behind.reads	false
dfs.datanode.drop.cache.behind.writes	false
dfs.datanode.sync.behind.writes	true

Compression Tuning

CarbonData uses a few lightweight compression and heavyweight compression algorithms to compress data. Although these algorithms can process any type of data, the compression performance is better if the data is ordered with similar values being together.

During data loading, data is sorted based on the order of columns in the table to achieve good compression performance.

Since CarbonData sorts data in the order of columns defined in the table, the order of columns plays an important role in the effectiveness of compression. If the low cardinality dimension is on the left, the range of data partitions after sorting is small and the compression efficiency is high. If a high cardinality dimension is on the left, a range of data partitions obtained after sorting is relatively large, and compression efficiency is relatively low.

Memory Tuning

CarbonData provides a mechanism for memory tuning where data loading depends on the columns needed in the query. Whenever a query command is received, columns required by the query are fetched and data is loaded for those columns in memory. During this operation, if the memory threshold is reached, the least used loaded files are deleted to release memory space for columns required by the query.

3.4.2 Suggestions for Creating CarbonData Tables

Scenario

This section provides suggestions based on more than 50 test cases to help you create CarbonData tables with higher query performance.

Table 3-14 Columns in the CarbonData table

Column name	Data type	Cardinality	Attribution
msisdn	String	30 million	dimension
BEGIN_TIME	bigint	10,000	dimension
host	String	1 million	dimension

Column name	Data type	Cardinality	Attribution
dime_1	String	1,000	dimension
dime_2	String	500	dimension
dime_3	String	800	dimension
counter_1	numeric(20,0)	NA	measure
...	...	NA	measure
counter_100	numeric(20,0)	NA	measure

Procedure

- If the to-be-created table contains a column that is frequently used for filtering, for example, this column is used in more than 80% of filtering scenarios,

implement optimization as follows:

Place this column in the first column of **sort_columns**.

For example, if **msisdn** is the most frequently used filter criterion in a query, it is placed in the first column. Run the following command to create a table. The query performance is good if **msisdn** is used as the filter condition.

```
create table carbondata_table(
  msisdn String,
  ...
)STORED AS carbondata TBLPROPERTIES ('SORT_COLUMNS'='msisdn');
```

- If the to-be-created table has multiple columns which are frequently used to filter the results,

implement optimization as follows:

Create an index for the columns.

For example, if **msisdn**, **host**, and **dime_1** are frequently used columns, the **sort_columns** column sequence is "dime_1-> host-> msisdn..." based on cardinality. Run the following command to create a table. The following command can improve the filtering performance of **dime_1**, **host**, and **msisdn**.

```
create table carbondata_table(
  dime_1 String,
  host String,
  msisdn String,
  dime_2 String,
  dime_3 String,
  ...
)STORED AS carbondata
TBLPROPERTIES ('SORT_COLUMNS'='dime_1,host,msisdn');
```

- If the frequency of each column used for filtering is similar,

implement optimization as follows:

sort_columns is sorted in ascending order of cardinality.

Run the following command to create a table:

```
create table carbondata_table(
  Dime_1 String,
```

```
BEGIN_TIME bigint,
HOST String,
MSISDN String,
...
)STORED AS carbondata
TBLPROPERTIES ('SORT_COLUMNS'='dime_2,dime_3,dime_1, BEGIN_TIME,host,msisdn');
```

- Create tables in ascending order of cardinalities. Then create secondary indexes for columns with more cardinalities. The statement for creating an index is as follows:

```
create index carbondata_table_index_msidn on tablecarbondata_table (
MSISDN String) as 'carbondata' PROPERTIES ('table_blocksize'='128');
create index carbondata_table_index_host on tablecarbondata_table (
host String) as 'carbondata' PROPERTIES ('table_blocksize'='128');
```

- For columns of measure type, not requiring high accuracy, the numeric (20,0) data type is not required. You are advised to use the double data type to replace the numeric (20,0) data type to enhance query performance.

The result of performance analysis of test-case shows reduction in query execution time from 15 to 3 seconds, thereby improving performance by nearly 5 times. The command for creating a table is as follows:

```
create table carbondata_table(
Dime_1 String,
BEGIN_TIME bigint,
HOST String,
MSISDN String,
counter_1 double,
counter_2 double,
...
counter_100 double,
)STORED AS carbondata
;
```

- If values (**start_time** for example) of a column are incremental:
For example, if data is loaded to CarbonData every day, **start_time** is incremental for each load. In this case, it is recommended that the **start_time** column be put at the end of **sort_columns**, because incremental values are efficient in using min/max index. The command for creating a table is as follows:

```
create table carbondata_table(
Dime_1 String,
HOST String,
MSISDN String,
counter_1 double,
counter_2 double,
BEGIN_TIME bigint,
...
counter_100 double,
)STORED AS carbondata
TBLPROPERTIES ( 'SORT_COLUMNS'='dime_2,dime_3,dime_1..BEGIN_TIME');
```

3.4.3 Configurations for Performance Tuning

Scenario

This section describes the configurations that can improve CarbonData performance.

Procedure

[Table 3-15](#) and [Table 3-16](#) describe the configurations about query of CarbonData.

Table 3-15 Number of tasks started for the shuffle process

Parameter	spark.sql.shuffle.partitions
Configuration File	spark-defaults.conf
Function	Data query
Scenario Description	Number of tasks started for the shuffle process in Spark
Tuning	You are advised to set this parameter to one to two times as much as the executor cores. In an aggregation scenario, reducing the number from 200 to 32 can reduce the query time by two folds.

Table 3-16 Number of executors and vCPUs, and memory size used for CarbonData data query

Parameter	spark.executor.cores spark.executor.instances spark.executor.memory
Configuration File	spark-defaults.conf
Function	Data query
Scenario Description	Number of executors and vCPUs, and memory size used for CarbonData data query
Tuning	In the bank scenario, configuring 4 vCPUs and 15 GB memory for each executor will achieve good performance. The two values do not mean the more the better. Configure the two values properly in case of limited resources. If each node has 32 vCPUs and 64 GB memory in the bank scenario, the memory is not sufficient. If each executor has 4 vCPUs and 12 GB memory, Garbage Collection may occur during query, time spent on query from increases from 3s to more than 15s. In this case, you need to increase the memory or reduce the number of vCPUs.

[Table 3-17](#), [Table 3-18](#), and [Table 3-19](#) describe the configurations for CarbonData data loading.

Table 3-17 Number of vCPUs used for data loading

Parameter	carbon.number.of.cores.while.loading
------------------	--------------------------------------

Configuration File	carbon.properties
Function	Data loading
Scenario Description	Number of vCPUs used for data processing during data loading in CarbonData
Tuning	If there are sufficient CPUs, you can increase the number of vCPUs to improve performance. For example, if the value of this parameter is changed from 2 to 4, the CSV reading performance can be doubled.

Table 3-18 Whether to use Yarn local directories for multi-disk data loading

Parameter	carbon.use.local.dir
Configuration File	carbon.properties
Function	Data loading
Scenario Description	Whether to use Yarn local directories for multi-disk data loading
Tuning	If this parameter is set to true , CarbonData uses local Yarn directories for multi-table load disk load balance, improving data loading performance.

Table 3-19 Whether to use multiple directories during loading

Parameter	carbon.use.multiple.temp.dir
Configuration File	carbon.properties
Function	Data loading
Scenario Description	Whether to use multiple temporary directories to store temporary sort files
Tuning	If this parameter is set to true , multiple temporary directories are used to store temporary sort files during data loading. This configuration improves data loading performance and prevents single points of failure (SPOFs) on disks.

[Table 3-20](#) describes the configurations for CarbonData data loading and query.

Table 3-20 Number of vCPUs used for data loading and query

Parameter	carbon.compaction.level.threshold
Configuration File	carbon.properties
Function	Data loading and query
Scenario Description	For minor compaction, specifies the number of segments to be merged in stage 1 and number of compacted segments to be merged in stage 2.
Tuning	Each CarbonData load will create one segment, if every load is small in size, it will generate many small files over a period of time impacting the query performance. Configuring this parameter will merge the small segments to one big segment which will sort the data and improve the performance. The compaction policy depends on the actual data size and available resources. For example, a bank loads data once a day and at night when no query is performed. If resources are sufficient, the compaction policy can be 6 or 5.

Table 3-21 Whether to enable data pre-loading when the index cache server is used

Parameter	carbon.indexserver.enable.prepriming
Configuration File	carbon.properties
Function	Data loading
Scenario Description	Enabling data pre-loading during the use of the index cache server can improve the performance of the first query.
Tuning	You can set this parameter to true to enable the pre-loading function. The default value is false .

3.5 CarbonData Access Control

The following table provides details about Hive ACL permissions required for performing operations on CarbonData tables.

Prerequisites

Parameters listed in [Table 3-7](#) or [Table 3-8](#) have been configured.

Hive ACL permissions

Table 3-22 Hive ACL permissions required for CarbonData table-level operations

Scenario	Required Permission
DESCRIBE TABLE	SELECT (of table)
SELECT	SELECT (of table)
EXPLAIN	SELECT (of table)
CREATE TABLE	CREATE (of database)
CREATE TABLE As SELECT	CREATE (on database), INSERT (on table), RW on data file, and SELECT (on table)
LOAD	INSERT (of table) RW on data file
DROP TABLE	OWNER (of table)
DELETE SEGMENTS	DELETE (of table)
SHOW SEGMENTS	SELECT (of table)
CLEAN FILES	DELETE (of table)
INSERT OVERWRITE / INSERT INTO	INSERT (of table) RW on data file and SELECT (of table)
CREATE INDEX	OWNER (of table)
DROP INDEX	OWNER (of table)
SHOW INDEXES	SELECT (of table)
ALTER TABLE ADD COLUMN	OWNER (of table)
ALTER TABLE DROP COLUMN	OWNER (of table)
ALTER TABLE CHANGE DATATYPE	OWNER (of table)
ALTER TABLE RENAME	OWNER (of table)
ALTER TABLE COMPACTION	INSERT (on table)
FINISH STREAMING	OWNER (of table)
ALTER TABLE SET STREAMING PROPERTIES	OWNER (of table)
ALTER TABLE SET TABLE PROPERTIES	OWNER (of table)
UPDATE CARBON TABLE	UPDATE (of table)
DELETE RECORDS	DELETE (of table)
REFRESH TABLE	OWNER (of main table)

Scenario	Required Permission
REGISTER INDEX TABLE	OWNER (of table)
SHOW PARTITIONS	SELECT (on table)
ALTER TABLE ADD PARTITION	OWNER (of table)
ALTER TABLE DROP PARTITION	OWNER (of table)

 **NOTE**

- If tables in the database are created by multiple users, the **Drop database** command fails to be executed even if the user who runs the command is the owner of the database.
- In a secondary index, when the parent table is triggered, **insert** and **compaction** are triggered on the index table. If you select a query that has a filter condition that matches index table columns, you should provide selection permissions for the parent table and index table.
- The LockFiles folder and lock files created in the LockFiles folder will have full permissions, as the LockFiles folder does not contain any sensitive data.
- If you are using ACL, ensure you do not configure any path for DDL or DML which is being used by other process. You are advised to create new paths.
Configure the path for the following configuration items:
 - 1) carbon.badRecords.location
 - 2) Db_Path and other items during database creation
- For Carbon ACL in a non-security cluster, **hive.server2.enable.doAs** in the **hive-site.xml** file must be set to **false**. Then the query will run as the user who runs the hiveserver2 process.

3.6 CarbonData Syntax Reference

3.6.1 DDL

3.6.1.1 CREATE TABLE

Function

This command is used to create a CarbonData table by specifying the list of fields along with the table properties.

Syntax

```
CREATE TABLE [IF NOT EXISTS] [db_name.]table_name
[(col_name data_type, ...)]
STORED AS carbodata
[TBLPROPERTIES (property_name=property_value, ...)];
```

Additional attributes of all tables are defined in **TBLPROPERTIES**.

Parameter Description

Table 3-23 CREATE TABLE parameters

Parameter	Description
db_name	Database name that contains letters, digits, and underscores (_).
col_name data_type	List with data types separated by commas (,). The column name contains letters, digits, and underscores (_). NOTE When creating a CarbonData table, do not use tupleId, PositionId, and PositionReference as column names because columns with these names are internally used by secondary index commands.
table_name	Table name of a database that contains letters, digits, and underscores (_).
STORED AS	The carbondata parameter defines and creates a CarbonData table.
TBLPROPERTIES	List of CarbonData table properties.

Precautions

Table attributes are used as follows:

- Block size

The block size of a data file can be defined for a single table using **TBLPROPERTIES**. The larger one between the actual size of the data file and the defined block size is selected as the actual block size of the data file in HDFS. The unit is MB. The default value is 1024 MB. The value ranges from 1 MB to 2048 MB. If the value is beyond the range, the system reports an error.

Once the block size reaches the configured value, the write program starts a new block of CarbonData data. Data is written in multiples of the page size (32,000 records). Therefore, the boundary is not strict at the byte level. If the new page crosses the boundary of the configured block, the page is written to the new block instead of the current block.

```
TBLPROPERTIES('table_blocksize'='128')
```

 NOTE

- If a small block size is configured in the CarbonData table while the size of the data file generated by the loaded data is large, the block size displayed in HDFS is different from the configured value. This is because when data is written to a local block file for the first time, even though the size of the to-be-written data is larger than the configured value of the block size, data will still be written into the block. Therefore, the actual value of block size in HDFS is the larger value between the size of the data to be written and the configured block size.
- If **block.num** is less than the parallelism, the blocks are split into new blocks so that new blocks.num is greater than parallelism and all cores can be used. This optimization is called block distribution.
- **SORT_SCOPE** specifies the sort scope during table creation. There are four types of sort scopes:
 - **GLOBAL_SORT**: It improves query performance, especially for point queries. *TBLPROPERTIES('SORT_SCOPE'='GLOBAL_SORT')*
 - **LOCAL_SORT**: Data is sorted locally (task-level sorting).
 - **NO_SORT**: The default sorting mode is used. Data is loaded in unsorted manner, which greatly improves loading performance.

- **SORT_COLUMNS**

This table property specifies the order of sort columns.

TBLPROPERTIES('SORT_COLUMNS'='column1, column3')

 NOTE

- If this attribute is not specified, no columns are sorted by default.
- If this property is specified but with empty argument, then the table will be loaded without sort. For example, *('SORT_COLUMNS='')*.
- **SORT_COLUMNS** supports the string, date, timestamp, short, int, long, byte, and boolean data types.
- **RANGE_COLUMN**

This property is used to specify a column to partition the input data by range. Only one column can be configured. During data import, you can use **global_sort_partitions** or **scale_factor** to avoid generating small files.

TBLPROPERTIES('RANGE_COLUMN'='column1')

- **LONG_STRING_COLUMNS**

The length of a common string cannot exceed 32,000 characters. To store a string of more than 32,000 characters, set **LONG_STRING_COLUMNS** to the target column.

TBLPROPERTIES('LONG_STRING_COLUMNS'='column1, column3')

 NOTE

LONG_STRING_COLUMNS can be set only for columns of the STRING, CHAR, or VARCHAR type.

Scenarios

Creating a Table by Specifying Columns

The **CREATE TABLE** command is the same as that of Hive DDL. The additional configurations of CarbonData are provided as table properties.

```
CREATE TABLE [IF NOT EXISTS] [db_name.]table_name  
[(col_name data_type , ...)]  
STORED AS carbodata  
[TBLPROPERTIES (property_name=property_value, ...)];
```

Examples

```
CREATE TABLE IF NOT EXISTS productdb.productSalesTable (  
productNumber Int,  
productName String,  
storeCity String,  
storeProvince String,  
productCategory String,  
productBatch String,  
saleQuantity Int,  
revenue Int)  
STORED AS carbodata  
TBLPROPERTIES (  
'table_blocksize'='128',  
'SORT_COLUMNS'='productBatch, productName')
```

System Response

A table will be created and the success message will be logged in system logs.

3.6.1.2 CREATE TABLE As SELECT

Function

This command is used to create a CarbonData table by specifying the list of fields along with the table properties.

Syntax

```
CREATE TABLE [IF NOT EXISTS] [db_name.]table_name STORED AS carbodata  
[TBLPROPERTIES (key1=val1, key2=val2, ...)] AS select_statement;
```

Parameter Description

Table 3-24 CREATE TABLE parameters

Parameter	Description
db_name	Database name that contains letters, digits, and underscores (_).
table_name	Table name of a database that contains letters, digits, and underscores (_).
STORED AS	Used to store data in CarbonData format.
TBLPROPERTIES	List of CarbonData table properties. For details, see Precautions .

Precautions

N/A

Examples

```
CREATE TABLE ctas_select_parquet STORED AS carbondata as select * from
parquet_ctas_test;
```

System Response

This example will create a Carbon table from any Parquet table and load all the records from the Parquet table.

3.6.1.3 DROP TABLE

Function

This command is used to delete an existing table.

Syntax

```
DROP TABLE [IF EXISTS] [db_name.]table_name;
```

Parameter Description

Table 3-25 DROP TABLE parameters

Parameter	Description
db_name	Database name. If this parameter is not specified, the current database is selected.
table_name	Name of the table to be deleted

Precautions

In this command, **IF EXISTS** and **db_name** are optional.

Example

```
DROP TABLE IF EXISTS productDatabase.productSalesTable;
```

System Response

The table will be deleted.

3.6.1.4 SHOW TABLES

Function

SHOW TABLES command is used to list all tables in the current or a specific database.

Syntax

```
SHOW TABLES [IN db_name];
```

Parameter Description

Table 3-26 SHOW TABLE parameters

Parameter	Description
IN db_name	Name of the database. This parameter is required only when tables of this specific database are to be listed.

Usage Guidelines

IN db_Name is optional.

Examples

```
SHOW TABLES IN ProductDatabase;
```

System Response

All tables are listed.

3.6.1.5 ALTER TABLE COMPACTION

Function

The **ALTER TABLE COMPACTION** command is used to merge a specified number of segments into a single segment. This improves the query performance of a table.

Syntax

```
ALTER TABLE [db_name.]table_name COMPACT 'MINOR/MAJOR/  
SEGMENT_INDEX';
```

```
ALTER TABLE [db_name.]table_name COMPACT 'CUSTOM' WHERE SEGMENT.ID IN  
(id1, id2, ...);
```

Parameter Description

Table 3-27 ALTER TABLE COMPACTION parameters

Parameter	Description
db_name	Database name. If this parameter is not specified, the current database is selected.
table_name	Table name.
MINOR	Minor compaction. For details, see Combining Segments .
MAJOR	Major compaction. For details, see Combining Segments .
SEGMENT_INDEX	This configuration enables you to merge all the CarbonData index files (.carbonindex) inside a segment to a single CarbonData index merge file (.carbonindexmerge). This enhances the first query performance. For more information, see Table 3-12 .
CUSTOM	Custom compaction. For details, see Combining Segments .

Precautions

N/A

Examples

```
ALTER TABLE ProductDatabase COMPACT 'MINOR';
```

```
ALTER TABLE ProductDatabase COMPACT 'MAJOR';
```

```
ALTER TABLE ProductDatabase COMPACT 'SEGMENT_INDEX';
```

```
ALTER TABLE ProductDatabase COMPACT 'CUSTOM' WHERE SEGMENT.ID IN  
(0, 1);
```


System Response

ALTER TABLE COMPACTION does not show the response of the compaction because it is run in the background.

If you want to view the response of minor and major compactions, you can check the logs or run the **SHOW SEGMENTS** command.

Example:

```

+-----+-----+-----+-----+-----+-----+-----+-----+
+--+
| ID | Status | Load Start Time | Load Time Taken | Partition | Data Size | Index Size | File
Format |
+-----+-----+-----+-----+-----+-----+-----+-----+
+--+
| 3 | Success | 2020-09-28 22:53:26.336 | 3.726S | {} | 6.47KB | 3.30KB | columnar_v3 |
| 2 | Success | 2020-09-28 22:53:01.702 | 6.688S | {} | 6.47KB | 3.30KB | columnar_v3 |
| 1 | Compacted | 2020-09-28 22:51:15.242 | 5.82S | {} | 6.50KB | 3.43KB |
columnar_v3 |
| 0.1 | Success | 2020-10-30 20:49:24.561 | 16.66S | {} | 12.87KB | 6.91KB | columnar_v3 |
| 0 | Compacted | 2020-09-28 22:51:02.6 | 6.819S | {} | 6.50KB | 3.43KB | columnar_v3 |
+-----+-----+-----+-----+-----+-----+-----+-----+
+--+

```

In the preceding information:

- **Compacted** indicates that data has been compacted.
- **0.1** indicates the compacting result of segment 0 and segment 1.

The compact operation does not incur any change to other operations.

Compacted segments, such as segment 0 and segment 1, become useless. To save space, before you perform other operations, run the **CLEAN FILES** command to delete compacted segments. For more information about the **CLEAN FILES** command, see [CLEAN FILES](#).

3.6.1.6 TABLE RENAME

Function

This command is used to rename an existing table.

Syntax

```
ALTER TABLE [db_name.]table_name RENAME TO new_table_name;
```

Parameter Description

Table 3-28 RENAME parameters

Parameter	Description
db_name	Database name. If this parameter is not specified, the current database is selected.

Parameter	Description
table_name	Current name of the existing table
new_table_name	New name of the existing table

Precautions

- Parallel queries (using table names to obtain paths for reading CarbonData storage files) may fail during this operation.
- The secondary index table cannot be renamed.

Example

```
ALTER TABLE carbon RENAME TO carbondata;
```

```
ALTER TABLE test_db.carbon RENAME TO test_db.carbondata;
```

System Response

The new table name will be displayed in the CarbonData folder. You can run **SHOW TABLES** to view the new table name.

3.6.1.7 ADD COLUMNS

Function

This command is used to add a column to an existing table.

Syntax

```
ALTER TABLE [db_name.]table_name ADD COLUMNS (col_name data_type,...)
TBLPROPERTIES ('COLUMNPROPERTIES.columnName.shared_column'='sharedFolder.sharedColumnName,...', 'DEFAULT.VALUE.COLUMN_NAME'='default_value');
```

Parameter Description

Table 3-29 ADD COLUMNS parameters

Parameter	Description
db_name	Database name. If this parameter is not specified, the current database is selected.
table_name	Table name.

Parameter	Description
col_name data_type	Name of a comma-separated column with a data type. It consists of letters, digits, and underscores (_). NOTE When creating a CarbonData table, do not name columns as tupleId, PositionId, and PositionReference because they will be used in UPDATE, DELETE, and secondary index commands.

Precautions

- Only **shared_column** and **default_value** are read. If any other property name is specified, no error will be thrown and the property will be ignored.
- If no default value is specified, the default value of the new column is considered null.
- If filter is applied to the column, new columns will not be added during sort. New columns may affect query performance.

Examples

- **ALTER TABLE** *carbon* **ADD COLUMNS** (*a1 INT, b1 STRING*);
- **ALTER TABLE** *carbon* **ADD COLUMNS** (*a1 INT, b1 STRING*)
TBLPROPERTIES ('COLUMNPROPERTIES.b1.shared_column'='sharedFolder.b1');
- **ALTER TABLE** *carbon* **ADD COLUMNS** (*a1 INT, b1 STRING*)
TBLPROPERTIES ('DEFAULT.VALUE.a1'='10');

System Response

The newly added column can be displayed by running the **DESCRIBE** command.

3.6.1.8 DROP COLUMNS

Function

This command is used to delete one or more columns from a table.

Syntax

```
ALTER TABLE [db_name.]table_name DROP COLUMNS (col_name, ...);
```

Parameter Description

Table 3-30 DROP COLUMNS parameters

Parameter	Description
db_name	Database name. If this parameter is not specified, the current database is selected.

Parameter	Description
table_name	Table name.
col_name	Name of a column in a table. Multiple columns are supported. It consists of letters, digits, and underscores (_).

Precautions

After a column is deleted, at least one key column must exist in the schema. Otherwise, an error message is displayed, and the column fails to be deleted.

Examples

Assume that the table contains four columns named a1, b1, c1, and d1.

- Delete a column:
ALTER TABLE *carbon* DROP COLUMNS (*b1*);
ALTER TABLE *test_db.carbon* DROP COLUMNS (*b1*);
- Delete multiple columns:
ALTER TABLE *carbon* DROP COLUMNS (*b1,c1*);
ALTER TABLE *test_db.carbon* DROP COLUMNS (*b1,c1*);

System Response

If you run the **DESCRIBE** command, the deleted columns will not be displayed.

3.6.1.9 CHANGE DATA TYPE

Function

This command is used to change the data type from INT to BIGINT or decimal precision from lower to higher.

Syntax

```
ALTER TABLE [db_name.]table_name CHANGE col_name col_name changed_column_type;
```

Parameter Description

Table 3-31 CHANGE DATA TYPE parameters

Parameter	Description
db_name	Name of the database. If this parameter is left unspecified, the current database is selected.
table_name	Name of the table.

Parameter	Description
col_name	Name of columns in a table. Column names contain letters, digits, and underscores (_).
changed_column_type	The change in the data type.

Usage Guidelines

- Change of decimal data type from lower precision to higher precision will only be supported for cases where there is no data loss.
Example:
 - **Invalid scenario** - Change of decimal precision from (10,2) to (10,5) is not valid as in this case only scale is increased but total number of digits remain the same.
 - **Valid scenario** - Change of decimal precision from (10,2) to (12,3) is valid as the total number of digits are increased by 2 but scale is increased only by 1 which will not lead to any data loss.
- The allowed range is 38,38 (precision, scale) and is a valid upper case scenario which is not resulting in data loss.

Examples

- Changing data type of column a1 from INT to BIGINT.
ALTER TABLE test_db.carbon CHANGE a1 a1 BIGINT;
- Changing decimal precision of column a1 from 10 to 18.
ALTER TABLE test_db.carbon CHANGE a1 a1 DECIMAL(18,2);

System Response

By running DESCRIBE command, the changed data type for the modified column is displayed.

3.6.1.10 REFRESH TABLE

Function

This command is used to register Carbon table to Hive meta store catalogue from existing Carbon table data.

Syntax

REFRESH TABLE db_name.table_name;

Parameter Description

Table 3-32 REFRESH TABLE parameters

Parameter	Description
db_name	Name of the database. If this parameter is left unspecified, the current database is selected.
table_name	Name of the table.

Usage Guidelines

- The new database name and the old database name should be same.
- Before executing this command the old table schema and data should be copied into the new database location.
- If the table is aggregate table, then all the aggregate tables should be copied to the new database location.
- For old store, the time zone of the source and destination cluster should be same.
- If old cluster used HIVE meta store to store schema, refresh will not work as schema file does not exist in file system.

Examples

```
REFRESH TABLE dbcarbon.productSalesTable;
```

System Response

By running this command, the Carbon table will be registered to Hive meta store catalogue from existing Carbon table data.

3.6.1.11 REGISTER INDEX TABLE

Function

This command is used to register an index table with the primary table.

Syntax

```
REGISTER INDEX TABLE indextable_name ON db_name.maintable_name;
```

Parameter Description

Table 3-33 REFRESH INDEX TABLE parameters

Parameter	Description
db_name	Database name. If this parameter is not specified, the current database is selected.
indextable_name	Index table name.
maintable_name	Primary table name.

Precautions

Before running this command, run **REFRESH TABLE** to register the primary table and secondary index table with the Hive metastore.

Examples

```
REGISTER INDEX TABLE productNameIndexTable ON
productdb.productSalesTable;
```

System Response

By running this command, the index table will be registered to the primary table.

3.6.1.12 REFRESH INDEX

Function

This command is used to merge all segments for data files in the secondary index table.

Syntax

- **REFRESH INDEX** *indextable_name* ON TABLE *maintable_name*
This command is used to merge all segments for which data files are to be merged.
- **REFRESH INDEX** *indextable_name* ON TABLE *maintable_name* WHERE SEGMENT.ID IN (0,1,2..N)
This command is used to merge a batch of specified segments.

Parameter Description

Table 3-34 REFRESH INDEX parameters

Parameter	Description
indextable_name	Name of an index table

Parameter	Description
maintable_name	Name of the primary table

Precautions

To clear the data file of compacted segments, run the **CLEAN FILES** command on the secondary index table.

Examples

```
REFRESH INDEX productNameIndexTable;
```

System Response

After this command is executed, the number of data files in the secondary index table will be reduced.

3.6.2 DML

3.6.2.1 LOAD DATA

Function

This command is used to load user data of a particular type, so that CarbonData can provide good query performance.

NOTE

Only the raw data on HDFS can be loaded.

Syntax

```
LOAD DATA INPATH 'folder_path' INTO TABLE [db_name.]table_name  
OPTIONS(property_name=property_value, ...);
```

Parameter Description

Table 3-35 LOAD DATA parameters

Parameter	Description
folder_path	Path of the file or folder used for storing the raw CSV data.
db_name	Database name. If this parameter is not specified, the current database is used.
table_name	Name of a table in a database.

Precautions

The following configuration items are involved during data loading:

- **DELIMITER:** Delimiters and quote characters provided in the load command. The default value is a comma (,).

OPTIONS('DELIMITER=',', 'QUOTECHAR='')

You can use '**DELIMITER**'='\t' to separate CSV data using tabs.

OPTIONS('DELIMITER='\t')

CarbonData also supports **\001** and **\017** as delimiters.

NOTE

When the delimiter of CSV data is a single quotation mark ('), the single quotation mark must be enclosed in double quotation marks (" "). For example, 'DELIMITER='''.

- **QUOTECHAR:** Delimiters and quote characters provided in the load command. The default value is double quotation marks (").
OPTIONS('DELIMITER=',', 'QUOTECHAR='')
- **COMMENTCHAR:** Comment characters provided in the load command. During data loading, if there is a comment character at the beginning of a line, the line is regarded as a comment line and data in the line will not be loaded. The default value is a pound key (#).
OPTIONS('COMMENTCHAR'='#')
- **FILEHEADER:** If the source file does not contain any header, add a header to the **LOAD DATA** command.
OPTIONS('FILEHEADER'='column1,column2')
- **ESCAPECHAR:** Is used to perform strict verification of the escape character on CSV files. The default value is backslash (\).
OPTIONS('ESCAPECHAR'='\')

NOTE

Enter **ESCAPECHAR** in the CSV data. **ESCAPECHAR** must be enclosed in double quotation marks (" "). For example, "a\b".

- Bad records handling:

In order for the data processing application to provide benefits, certain data integration is required. In most cases, data quality problems are caused by data sources.

Methods of handling bad records are as follows:

- Load all of the data before dealing with the errors.
- Clean or delete bad records before loading data or stop the loading when bad records are found.

There are many options for clearing source data during CarbonData data loading, as listed in [Table 3-36](#).

Table 3-36 Bad Records Logger

Configuration Item	Default Value	Description
BAD_RECORDS_LOGGER_ENABLE	false	Whether to create logs with details about bad records
BAD_RECORDS_ACTION	FAIL	<p>The four types of actions for bad records are as follows:</p> <ul style="list-style-type: none"> ● FORCE: Auto-corrects the data by storing the bad records as NULL. ● REDIRECT: Bad records are written to the raw CSV instead of being loaded. ● IGNORE: Bad records are neither loaded nor written to the raw CSV. ● FAIL: Data loading fails if any bad records are found. <p>NOTE In loaded data, if all records are bad records, BAD_RECORDS_ACTION is invalid and the load operation fails.</p>
IS_EMPTY_DATA_BAD_RECORD	false	Whether empty data of a column to be considered as bad record or not. If this parameter is set to false , empty data ("",', or,) is not considered as bad records. If this parameter is set to true , empty data is considered as bad records.
BAD_RECORD_PATH	-	HDFS path where bad records are stored. The default value is Null . If bad records logging or bad records operation redirection is enabled, the path must be configured by the user.

Example:

```
LOAD DATA INPATH 'filepath.csv' INTO TABLE tablename
OPTIONS('BAD_RECORDS_LOGGER_ENABLE'='true',
'BAD_RECORD_PATH'='hdfs://hacluster/tmp/carbon',
'BAD_RECORDS_ACTION'='REDIRECT',
'IS_EMPTY_DATA_BAD_RECORD'='false');
```

 **NOTE**

If **REDIRECT** is used, CarbonData will add all bad records into a separate CSV file. However, this file must not be used for subsequent data loading because the content may not exactly match the source record. You must clean up the source record for further data ingestion. This option is used to remind you which records are bad.

- **MAXCOLUMNS:** (Optional) Specifies the maximum number of columns parsed by a CSV parser in a line.

OPTIONS('MAXCOLUMNS'='400')

Table 3-37 MAXCOLUMNS

Name of the Optional Parameter	Default Value	Maximum Value
MAXCOLUMNS	2000	20000

Table 3-38 Behavior chart of MAXCOLUMNS

MAXCOLUMNS Value	Number of Columns in the File Header	Final Value Considered
Not specified in Load options	5	2000
Not specified in Load options	6000	6000
40	7	Max (column count of file header, MAXCOLUMNS value)
22000	40	20000
60	Not specified in Load options	Max (Number of columns in the first line of the CSV file, MAXCOLUMNS value)

 **NOTE**

There must be sufficient executor memory for setting the maximum value of **MAXCOLUMNS Option**. Otherwise, data loading will fail.

- If **SORT_SCOPE** is set to **GLOBAL_SORT** during table creation, you can specify the number of partitions to be used when sorting data. If this parameter is not set or is set to a value less than **1**, the number of map tasks is used as the number of reduce tasks. It is recommended that each reduce task process 512 MB to 1 GB data.

OPTIONS('GLOBAL_SORT_PARTITIONS'='2')

 **NOTE**

To increase the number of partitions, you may need to increase the value of **spark.driver.maxResultSize**, as the sampling data collected in the driver increases with the number of partitions.

- **DATEFORMAT**: Specifies the date format of the table.

```
OPTIONS('DATEFORMAT'='dateFormat')
```

 **NOTE**

Date formats are specified by date pattern strings. The date pattern letters in Carbon are same as in JAVA.

- **TIMESTAMPFORMAT**: Specifies the timestamp of a table.
- *OPTIONS('TIMESTAMPFORMAT'='timestampFormat')*
- **SKIP_EMPTY_LINE**: Ignores empty rows in the CSV file during data loading.

```
OPTIONS('SKIP_EMPTY_LINE'='TRUE/FALSE')
```

- **Optional: SCALE_FACTOR**: Used to control the number of partitions for **RANGE_COLUMN**, **SCALE_FACTOR**. The formula is as follows:
splitSize = max(blocklet_size, (block_size - blocklet_size)) * scale_factor
numPartitions = total size of input data / splitSize

The default value is **3**. The value ranges from **1** to **300**.

```
OPTIONS('SCALE_FACTOR'='10')
```

 **NOTE**

- If **GLOBAL_SORT_PARTITIONS** and **SCALE_FACTOR** are used at the same time, only **GLOBAL_SORT_PARTITIONS** is valid.
- The compaction on **RANGE_COLUMN** will use **LOCAL_SORT** by default.

Scenarios

To load a CSV file to a CarbonData table, run the following statement:

```
LOAD DATA INPATH 'folder path' INTO TABLE tablename  
OPTIONS(property_name=property_value, ...);
```

Examples

```
LOAD DATA inpath 'hdfs://hacluster/src/test/resources/data.csv' INTO table  
carbontable  
options('DELIMITER'=';',  
'QUOTECHAR'='"',  
'COMMENTCHAR'='#',  
'ESCAPECHAR'='\',  
'FILEHEADER'='empno,empname,designation,doj,  
workgroupcategory,workgroupcategoryname,  
deptno,deptname,projectcode,projectjoindate,  
projectenddate,attendance,utilization,salary',
```

```
'DATEFORMAT' = 'yyyy-MM-dd'
);
```

System Response

Success or failure will be recorded in the driver logs.

3.6.2.2 UPDATE CARBON TABLE

Function

This command is used to update the CarbonData table based on the column expression and optional filtering conditions.

Syntax

- Syntax 1:
UPDATE <CARBON TABLE> SET (column_name1, column_name2, ... column_name n) = (column1_expression , column2_expression , column3_expression ... column n_expression) [WHERE { <filter_condition> }];
- Syntax 2:
UPDATE <CARBON TABLE> SET (column_name1, column_name2,) = (select sourceColumn1, sourceColumn2 from sourceTable [WHERE { <filter_condition> }]) [WHERE { <filter_condition> }];

Parameter Description

Table 3-39 UPDATE parameters

Parameter	Description
CARBON TABLE	Name of the CarbonData table to be updated
column_name	Target column to be updated
sourceColumn	Column value of the source table that needs to be updated in the target table
sourceTable	Table from which the records are updated to the target table

Precautions

Note the following before running this command:

- The UPDATE command fails if multiple input rows in the source table are matched with a single row in the target table.
- If the source table generates empty records, the UPDATE operation completes without updating the table.

- If rows in the source table do not match any existing rows in the target table, the UPDATE operation completes without updating the table.
- UPDATE is not allowed in the table with secondary index.
- In a subquery, if the source table and target table are the same, the UPDATE operation fails.
- The UPDATE operation fails if the subquery used in the UPDATE command contains an aggregate function or a GROUP BY clause.

For example, **update t_carbn01 a set (a.item_type_code, a.profit) = (select b.item_type_cd, sum(b.profit) from t_carbn01b b where item_type_cd =2 group by item_type_code);**

In the preceding example, aggregate function **sum(b.profit)** and GROUP BY clause are used in the subquery. As a result, the UPDATE operation will fail.

- If the **carbon.input.segments** property has been set for the queried table, the UPDATE operation fails. To solve this problem, run the following statement before the query:

Syntax:

```
SET carbon.input.segments. <database_name>. <table_name>=*;
```

Examples

- Example 1:
update carbonTable1 d set (d.column3,d.column5) = (select s.c33 ,s.c55 from sourceTable1 s where d.column1 = s.c11) where d.column1 = 'country' exists(select * from table3 o where o.c2 > 1);
- Example 2:
update carbonTable1 d set (c3) = (select s.c33 from sourceTable1 s where d.column1 = s.c11) where exists(select * from iud.other o where o.c2 > 1);
- Example 3:
update carbonTable1 set (c2, c5) = (c2 + 1, concat(c5 , "y"));
- Example 4:
update carbonTable1 d set (c2, c5) = (c2 + 1, "yx") where d.column1 = 'india';
- Example 5:
update carbonTable1 d set (c2, c5) = (c2 + 1, "yx") where d.column1 = 'india' and exists(select * from table3 o where o.column2 > 1);

System Response

Success or failure will be recorded in the driver log and on the client.

3.6.2.3 DELETE RECORDS from CARBON TABLE

Function

This command is used to delete records from a CarbonData table.

Syntax

```
DELETE FROM CARBON_TABLE [WHERE expression];
```

Parameter Description

Table 3-40 DELETE RECORDS parameters

Parameter	Description
CARBON TABLE	Name of the CarbonData table in which the DELETE operation is performed

Precautions

- If a segment is deleted, all secondary indexes associated with the segment are deleted as well.
- If the **carbon.input.segments** property has been set for the queried table, the DELETE operation fails. To solve this problem, run the following statement before the query:

Syntax:

```
SET carbon.input.segments. <database_name>.<table_name>=*;
```

Examples

- Example 1:
delete from columncarbonTable1 d where d.column1 = 'country';
- Example 2:
delete from dest where column1 IN ('country1', 'country2');
- Example 3:
delete from columncarbonTable1 where column1 IN (select column11 from sourceTable2);
- Example 4:
delete from columncarbonTable1 where column1 IN (select column11 from sourceTable2 where column1 = 'USA');
- Example 5:
delete from columncarbonTable1 where column2 >= 4;

System Response

Success or failure will be recorded in the driver log and on the client.

3.6.2.4 INSERT INTO CARBON TABLE

Function

This command is used to add the output of the SELECT command to a Carbon table.

Syntax

```
INSERT INTO [CARBON TABLE] [select query];
```

Parameter Description

Table 3-41 INSERT INTO parameters

Parameter	Description
CARBON TABLE	Name of the CarbonData table to be inserted
select query	SELECT query on the source table (CarbonData, Hive, and Parquet tables are supported)

Precautions

- A table has been created.
- You must belong to the data loading group in order to perform data loading operations. By default, the data loading group is named **ficommon**.
- CarbonData tables cannot be overwritten.
- The data type of the source table and the target table must be the same. Otherwise, data in the source table will be regarded as bad records.
- The **INSERT INTO** command does not support partial success. If bad records exist, the command fails.
- When you insert data of the source table to the target table, you cannot upload or update data of the source table.

To enable data loading or updating during the INSERT operation, set the following parameter to **true**.

carbon.insert.persist.enable=true

By default, the preceding parameters are set to **false**.

NOTE

Enabling this property will reduce the performance of the INSERT operation.

Example

```
INSERT INTO CARBON select * from TABLENAME;
```

System Response

Success or failure will be recorded in the driver logs.

3.6.2.5 DELETE SEGMENT by ID

Function

This command is used to delete segments by the ID.

Syntax

```
DELETE FROM TABLE db_name.table_name WHERE SEGMENT.ID IN  
(segment_id1,segment_id2);
```

Parameter Description

Table 3-42 DELETE SEGMENT parameters

Parameter	Description
segment_id	ID of the segment to be deleted.
db_name	Database name. If the parameter is not specified, the current database is used.
table_name	The name of the table in a specific database.

Usage Guidelines

Segments cannot be deleted from the stream table.

Examples

```
DELETE FROM TABLE CarbonDatabase.CarbonTable WHERE SEGMENT.ID IN  
(0);
```

```
DELETE FROM TABLE CarbonDatabase.CarbonTable WHERE SEGMENT.ID IN  
(0,5,8);
```

System Response

Success or failure will be recorded in the CarbonData log.

3.6.2.6 DELETE SEGMENT by DATE

Function

This command is used to delete segments by loading date. Segments created before a specific date will be deleted.

Syntax

```
DELETE FROM TABLE db_name.table_name WHERE SEGMENT.STARTTIME  
BEFORE date_value;
```

Parameter Description

Table 3-43 DELETE SEGMENT by DATE parameters

Parameter	Description
db_name	Database name. If this parameter is not specified, the current database is used.
table_name	Name of a table in the specified database
date_value	Valid date when segments are started to be loaded. Segments before the date will be deleted.

Precautions

Segments cannot be deleted from the stream table.

Example

```
DELETE FROM TABLE db_name.table_name WHERE SEGMENT.STARTTIME
BEFORE '2017-07-01 12:07:20';
```

STARTTIME indicates the loading start time of different loads.

System Response

Success or failure will be recorded in CarbonData logs.

3.6.2.7 SHOW SEGMENTS

Function

This command is used to list the segments of a CarbonData table.

Syntax

```
SHOW SEGMENTS FOR TABLE [db_name.]table_name LIMIT number_of_loads;
```

Parameter Description

Table 3-44 SHOW SEGMENTS FOR TABLE parameters

Parameter	Description
db_name	Database name. If this parameter is not specified, the current database is used.
table_name	Name of a table in the specified database
number_of_loads	Threshold of records to be listed

Precautions

None

Examples

SHOW SEGMENTS FOR TABLE *CarbonDatabase.CarbonTable* **LIMIT 2;**

System Response

```

+-----+-----+-----+-----+-----+-----+-----+-----+
+
| ID | Status | Load Start Time | Load Time Taken | Partition | Data Size | Index Size | File Format |
+-----+-----+-----+-----+-----+-----+-----+-----+
+
| 3 | Success | 2020-09-28 22:53:26.336 | 3.726S | {} | 6.47KB | 3.30KB | columnar_v3 |
| 2 | Success | 2020-09-28 22:53:01.702 | 6.688S | {} | 6.47KB | 3.30KB | columnar_v3 |
+-----+-----+-----+-----+-----+-----+-----+-----+
+

```

3.6.2.8 CREATE SECONDARY INDEX

Function

This command is used to create secondary indexes in the CarbonData tables.

Syntax

```

CREATE INDEX index_name
ON TABLE [db_name].table_name (col_name1, col_name2)
AS 'carbondata'
PROPERTIES ('table_blocksize'='256');

```

Parameter Description

Table 3-45 CREATE SECONDARY INDEX parameters

Parameter	Description
index_name	Index table name. It consists of letters, digits, and special characters (_).
db_name	Database name. It consists of letters, digits, and special characters (_).
table_name	Name of the database table. It consists of letters, digits, and special characters (_).
col_name	Name of a column in a table. Multiple columns are supported. It consists of letters, digits, and special characters (_).
table_blocksize	Block size of a data file. For details, see Block Size .

Precautions

`db_name` is optional.

Examples

- **CREATE INDEX productNameIndexTable on table productdb.productSalesTable (productName,city) as 'carbodata';**
In this example, a secondary table named **productdb.productNameIndexTable** is created and index information of the provided column is loaded.
- **CREATE INDEX t1_index1 on table t1 (c7_Datatype_Desc) AS 'carbodata' PROPERTIES('table_blocksize'='256');**

System Response

A secondary index table will be created. Index information related to the provided column will be loaded into the secondary index table. The success message will be recorded in system logs.

3.6.2.9 SHOW SECONDARY INDEXES

Function

This command is used to list all secondary index tables in the CarbonData table.

Syntax

```
SHOW INDEXES ON db_name.table_name;
```

Parameter Description

Table 3-46 SHOW SECONDARY INDEXES parameters

Parameter	Description
<code>db_name</code>	Database name. It consists of letters, digits, and special characters (_).
<code>table_name</code>	Name of the database table. It consists of letters, digits, and special characters (_).

Precautions

`db_name` is optional.

Examples

```
SHOW INDEXES ON productsales.product;
```

System Response

All index tables and corresponding index columns in a given CarbonData table will be listed.

3.6.2.10 DROP SECONDARY INDEX

Function

This command is used to delete the existing secondary index table in a specific table.

Syntax

```
DROP INDEX [IF EXISTS] index_name ON [db_name.]table_name;
```

Parameter Description

Table 3-47 DROP SECONDARY INDEX parameters

Parameter	Description
<i>index_name</i>	Name of the index table. Table name contains letters, digits, and underscores (_).
<i>db_name</i>	Name of the database. If the parameter is not specified, the current database is used.
<i>table_name</i>	Name of the table to be deleted.

Usage Guidelines

In this command, **IF EXISTS** and **db_name** are optional.

Examples

```
DROP INDEX if exists productNameIndexTable ON productdb.productSalesTable;
```

System Response

Secondary Index Table will be deleted. Index information will be cleared in CarbonData table and the success message will be recorded in system logs.

3.6.2.11 CLEAN FILES

Function

After the **DELETE SEGMENT** command is executed, the deleted segments are marked as the **delete** state. After the segments are merged, the status of the original segments changes to **compacted**. The data files of these segments are

not physically deleted. If you want to forcibly delete these files, run the **CLEAN FILES** command.

However, running this command may result in a query command execution failure.

Syntax

CLEAN FILES FOR TABLE *[db_name.]table_name* ;

Parameter Description

Table 3-48 CLEAN FILES FOR TABLE parameters

Parameter	Description
db_name	Database name. It consists of letters, digits, and underscores (_).
table_name	Name of the database table. It consists of letters, digits, and underscores (_).

Precautions

None

Examples

CLEAN FILES FOR TABLE *CarbonDatabase.CarbonTable*;

In this example, all the segments marked as **deleted** and **compacted** are physically deleted.

System Response

Success or failure will be recorded in the driver logs.

3.6.2.12 SET/RESET

Function

This command is used to dynamically add, update, display, or reset the CarbonData properties without restarting the driver.

Syntax

- Add or Update parameter value:
SET *parameter_name=parameter_value*
This command is used to add or update the value of **parameter_name**.
- Display property value:
SET *parameter_name*

This command is used to display the value of **parameter_name**.

- Display session parameter:

SET

This command is used to display all supported session parameters.

- Display session parameters along with usage details:

SET -v

This command is used to display all supported session parameters and their usage details.

- Reset parameter value:

RESET

This command is used to clear all session parameters.

Parameter Description

Table 3-49 SET parameters

Parameter	Description
parameter_name	Name of the parameter whose value needs to be dynamically added, updated, or displayed
parameter_value	New value of parameter_name to be set

Precautions

The following table lists the properties which you can set or clear using the SET or RESET command.

Table 3-50 Properties

Property	Description
carbon.options.bad.records.logger.enable	Whether to enable bad record logger.
carbon.options.bad.records.action	Operations on bad records, for example, force, redirect, fail, or ignore. For more information, see Bad record handling .
carbon.options.is.empty.data.bad.record	Whether the empty data is considered as a bad record. For more information, see Bad record handling .
carbon.options.sort.scope	Scope of the sort during data loading.
carbon.options.bad.record.path	HDFS path where bad records are stored.

Property	Description
carbon.custom.block.distribution	Whether to enable Spark or CarbonData block distribution.
enable.unsafe.sort	Whether to use unsafe sort during data loading. Unsafe sort reduces the garbage collection during data loading, thereby achieving better performance.
carbon.si.lookup.partialstring	<p>If this is set to TRUE, the secondary index uses the starts-with, ends-with, contains, and LIKE partition condition strings.</p> <p>If this is set to FALSE, the secondary index uses only the starts-with partition condition string.</p>
carbon.input.segments	<p>Segment ID to be queried. This property allows you to query a specified segment of a specified table. CarbonScan reads data only from the specified segment ID.</p> <p>Syntax:</p> <p>carbon.input.segments. <database_name>. <table_name> = <list of segment ids ></p> <p>If you want to query a specified segment in multi-thread mode, you can use CarbonSession.threadSet instead of the SET statement.</p> <p>Syntax:</p> <p>CarbonSession.threadSet ("carbon.input.segments. <database_name>. <table_name>","<list of segment ids >");</p> <p>NOTE You are advised not to set this property in the carbon.properties file because all sessions contain the segment list unless session-level or thread-level overwriting occurs.</p>

Examples

- Add or Update:
SET enable.unsafe.sort=true
- Display property value:
SET enable.unsafe.sort

- Show the segment ID list, segment status, and other required details, and specify the segment list to be read:

SHOW SEGMENTS FOR *TABLE carbontable1;*

SET *carbon.input.segments.db.carbontable1 = 1, 3, 9;*

- Query a specified segment in multi-thread mode:

CarbonSession.threadSet

("carbon.input.segments.default.carbon_table_Multi_Thread", "1,3");

- Use **CarbonSession.threadSet** to query segments in a multi-thread environment (Scala code is used as an example):

```
def main(args: Array[String]) {
  Future
  {
    CarbonSession.threadSet("carbon.input.segments.default.carbon_table_Multi_Thread", "1")
    spark.sql("select count(empno) from carbon_table_Multi_Thread").show()
  }
}
```

- Reset:

RESET

System Response

- Success will be recorded in the driver log.
- Failure will be displayed on the UI.

3.6.3 Operation Concurrent Execution

Before performing **DDL** and **DML** operations, you need to obtain the corresponding locks. See **Table 3-51** for details about the locks that need to be obtained for each operation. The check mark (√) indicates that the lock is required. An operation can be performed only after all required locks are obtained.

You can check whether any two operations can be executed concurrently by using the following method: The first two lines in **Table 3-51** indicate two operations. If no column in the two lines is marked with the check mark (√), the two operations can be executed concurrently. That is, if the columns with check marks (√) in the two lines do not exist, the two operations can be executed concurrently.

Table 3-51 List of obtaining locks for operations

Operation	MET DATA_ LOCK	COM PARTI TION_ LOCK	DRO P_ TABLE_ LOCK	DELE TE_ SEG MENT_ LOCK	CLEA N_ FILES_ LOCK	ALTE R_ PARTI TION_ LOCK	UPD ATE_ LOCK	STRE AMI NG_ LOCK	CON CURRE NT_ LOCK	SEG MENT_ LOCK
CREATE TABLE	-	-	-	-	-	-	-	-	-	-

Operation	METADATA_LOCK	COMPACTION_LOCK	DROP_TABLE_LOCK	DELETE_SEGMENT_LOCK	CLEANFILES_LOCK	ALTER_PARTITION_LOCK	UPDATE_LOCK	STREAMING_LOCK	CURRENT_LOAD_LOCK	SEGMENT_LOCK
CREATE TABLE AS SELECT	-	-	-	-	-	-	-	-	-	-
DROP TABLE	√	-	√	-	-	-	-	√	-	-
ALTER TABLE COMPACTION	-	√	-	-	-	-	√	-	-	-
TABLE RENAME	-	-	-	-	-	-	-	-	-	-
ADD COLUMNS	√	√	-	-	-	-	-	-	-	-
DROP COLUMNS	√	√	-	-	-	-	-	-	-	-
CHANGE DATA TYPE	√	√	-	-	-	-	-	-	-	-
REFRESH TABLE	-	-	-	-	-	-	-	-	-	-

Operation	METADATA_LOCK	COMPACTION_LOCK	DROP_TABLE_LOCK	DELETE_SEGMENT_LOCK	CLEANFILES_LOCK	ALTER_PARTITION_LOCK	UPDATE_LOCK	STREAMING_LOCK	CURRENT_LOAD_LOCK	SEGMENT_LOCK
REGISTER INDEX TABLE	√	-	-	-	-	-	-	-	-	-
REFRESH INDEX	-	√	-	-	-	-	-	-	-	-
LOAD DATA/INSERT INTO	-	-	-	-	-	-	-	-	√	√
UPDATE CARBON TABLE	√	√	-	-	-	-	√	-	-	-
DELETE RECORDS from CARBON TABLE	√	√	-	-	-	-	√	-	-	-
DELETE SEGMENT by ID	-	-	-	√	√	-	-	-	-	-

Operation	METADATA_LOCK	COMPACTIO_N_LOCK	DROP_TABLE_LOCK	DELETE_SEGMENT_LOCK	CLEAN_FILES_LOCK	ALTER_PARTITION_LOCK	UPDATE_LOCK	STREAMING_LOCK	CURRENT_LOAD_LOCK	SEGMENT_LOCK
DELETE SEGMENT by DATE	-	-	-	√	√	-	-	-	-	-
SHOW SEGMENTS	-	-	-	-	-	-	-	-	-	-
CREATE SECONDARY INDEX	√	√	-	√	-	-	-	-	-	-
SHOW SECONDARY INDEXES	-	-	-	-	-	-	-	-	-	-
DROP SECONDARY INDEX	√	-	√	-	-	-	-	-	-	-
CLEAN FILES	-	-	-	-	-	-	-	-	-	-
SET/RESET	-	-	-	-	-	-	-	-	-	-

Operation	METADATA_LOCK	COMPARTION_LOCK	DROP_TABLE_LOCK	DELETE_SEGMENT_LOCK	CLEANFILES_LOCK	ALTER_PARTITION_LOCK	UPDATE_LOCK	STREAMING_LOCK	CURRENT_LOAD_LOCK	SEGMENT_LOCK
Add Hive Partition	-	-	-	-	-	-	-	-	-	-
Drop Hive Partition	√	√	√	√	√	√	-	-	-	-
Drop Partition	√	√	√	√	√	√	-	-	-	-
Alter table set	√	√	-	-	-	-	-	-	-	-

3.6.4 API

This section describes the APIs and usage methods of Segment. All methods are in the org.apache.spark.util.CarbonSegmentUtil class.

The following methods have been abandoned:

```
/**
 * Returns the valid segments for the query based on the filter condition
 * present in carbonScanRdd.
 *
 * @param carbonScanRdd
 * @return Array of valid segments
 */
@deprecated def getFilteredSegments(carbonScanRdd: CarbonScanRDD[InternalRow]): Array[String];
```

Usage Method

Use the following methods to obtain CarbonScanRDD from the query statement:

```
val df=carbon.sql("select * from table where age='12'")
val myscan=df.queryExecution.sparkPlan.collect {
case scan: CarbonDataSourceScan if scan.rdd.isInstanceOf[CarbonScanRDD[InternalRow]] => scan.rdd
case scan: RowDataSourceScanExec if scan.rdd.isInstanceOf[CarbonScanRDD[InternalRow]] => scan.rdd
}.head
val carbonrdd=myscan.asInstanceOf[CarbonScanRDD[InternalRow]]
```

Example:

```
CarbonSegmentUtil.getFilteredSegments(carbonrdd)
```

The filtered segment can be obtained by importing SQL statements.

```
/**
 * Returns an array of valid segment numbers based on the filter condition provided in the sql
```

```
* NOTE: This API is supported only for SELECT Sql (insert into,ctas,.. is not supported)
*
* @param sql
* @param sparkSession
* @return Array of valid segments
* @throws UnsupportedOperationException because Get Filter Segments API supports if and only
* if only one carbon main table is present in query.
*/
def getFilteredSegments(sql: String, sparkSession: SparkSession): Array[String];
```

Example:

```
CarbonSegmentUtil.getFilteredSegments("select * from table where age='12'", sparkSession)
```

Import the database name and table name to obtain the list of segments to be merged. The obtained segments can be used as parameters of the getMergedLoadName function.

```
/**
 * Identifies all segments which can be merged with MAJOR compaction type.
 * NOTE: This result can be passed to getMergedLoadName API to get the merged load name.
 *
 * @param sparkSession
 * @param tableName
 * @param dbName
 * @return list of LoadMetadataDetails
 */
def identifySegmentsToBeMerged(sparkSession: SparkSession,
tableName: String,
dbName: String) : util.List[LoadMetadataDetails];
```

Example:

```
CarbonSegmentUtil.identifySegmentsToBeMerged(sparkSession, "table_test","default")
```

Import the database name, table name, and obtain all segments which can be merged with CUSTOM compaction type. The obtained segments can be transferred as the parameter of the getMergedLoadName function.

```
/**
 * Identifies all segments which can be merged with CUSTOM compaction type.
 * NOTE: This result can be passed to getMergedLoadName API to get the merged load name.
 *
 * @param sparkSession
 * @param tableName
 * @param dbName
 * @param customSegments
 * @return list of LoadMetadataDetails
 * @throws UnsupportedOperationException if customSegments is null or empty.
 * @throws MalformedCarbonCommandException if segment does not exist or is not valid
 */
def identifySegmentsToBeMergedCustom(sparkSession: SparkSession,
tableName: String,
dbName: String,
customSegments: util.List[String]): util.List[LoadMetadataDetails];
```

Example:

```
val customSegments = new util.ArrayList[String]()
customSegments.add("1")
customSegments.add("2")
CarbonSegmentUtil.identifySegmentsToBeMergedCustom(sparkSession, "table_test","default",
customSegments)
```

If a segment list is specified, the merged load name is returned.

```
/**
 * Returns the Merged Load Name for given list of segments
 *
 */
```

```
* @param list of segments
* @return Merged Load Name
* @throws UnsupportedOperationException if list of segments is less than 1
*/
def getMergedLoadName(list: util.List[LoadMetadataDetails]): String;
```

Example:

```
val carbonTable = CarbonEnv.getCarbonTable(Option(databaseName), tableName)(sparkSession)
val loadMetadataDetails = SegmentStatusManager.readLoadMetadata(carbonTable.getMetadataPath)
CarbonSegmentUtil.getMergedLoadName(loadMetadataDetails.toList.asJava)
```

3.6.5 Spatial Indexes

Quick Example

```
create table IF NOT EXISTS carbonTable
(
  COLUMN1 BIGINT,
  LONGITUDE BIGINT,
  LATITUDE BIGINT,
  COLUMN2 BIGINT,
  COLUMN3 BIGINT
)
STORED AS carbondata
TBLPROPERTIES
('SPATIAL_INDEX.mygeohash.type='geohash','SPATIAL_INDEX.mygeohash.sourcecolumns='longitude,
latitude','SPATIAL_INDEX.mygeohash.originLatitude'='39.850713','SPATIAL_INDEX.mygeohash.gridSize'='50','S
PATIAL_INDEX.mygeohash.minLongitude'='115.828503','SPATIAL_INDEX.mygeohash.maxLongitude'='720.000
000','SPATIAL_INDEX.mygeohash.minLatitude'='39.850713','SPATIAL_INDEX.mygeohash.maxLatitude'='720.0
00000','SPATIAL_INDEX='mygeohash','SPATIAL_INDEX.mygeohash.conversionRatio'='1000000','SORT_COLU
MNS'='column1,column2,column3,latitude,longitude');
```

Introduction to Spatial Indexes

Spatial data includes multidimensional points, lines, rectangles, cubes, polygons, and other geometric objects. A spatial data object occupies a certain region of space, called spatial scope, characterized by its location and boundary. The spatial data can be either point data or region data.

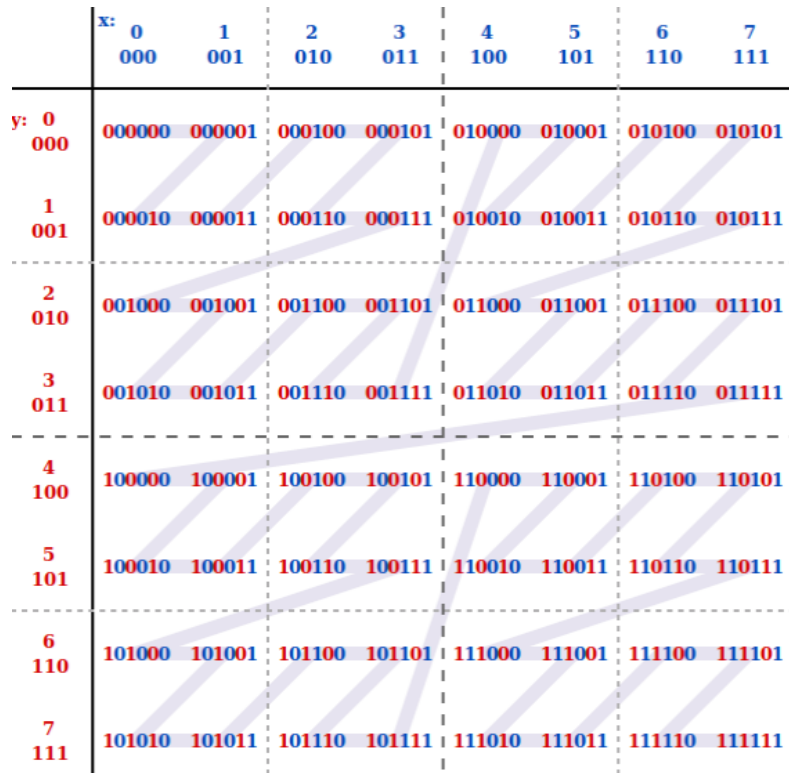
- Point data: A point has a spatial extent characterized completely by its location. It does not occupy space and has no associated boundary. Point data consists of a collection of points in a two-dimensional space. Points can be stored as a pair of longitude and latitude.
- Region data: A region has a spatial extent with a location, and boundary. The location can be considered as the position of a fixed point in the region, such as its centroid. In two dimensions, the boundary can be visualized as a line (for finite regions, a closed loop). Region data contains a collection of regions.

Currently, only point data is supported, and it can be stored.

Longitude and latitude can be encoded as a unique GeoID. Geohash is a public-domain geocoding system invented by Gustavo Niemeyer. It encodes geographical locations into a short string of letters and digits. It is a hierarchical spatial data structure which subdivides the space into buckets of grid shape, which is one of the many applications of what is known as the Z-order curve, and generally the space-filling curve.

The Z value of a point in multiple dimensions is calculated by interleaving the binary representation of its coordinate value, as shown in the following figure.

When Geohash is used to create a GeoID, data is sorted by GeoID instead of longitude and latitude. Data is stored by spatial proximity.



Creating a Table

GeoHash encoding:

```
create table IF NOT EXISTS carbonTable
(
...
`LONGITUDE` BIGINT,
`LATITUDE` BIGINT,
...
)
STORED AS carbondata
TBLPROPERTIES
('SPATIAL_INDEX.mygeohash.type='geohash','SPATIAL_INDEX.mygeohash.sourcecolumns='longitude,
latitude','SPATIAL_INDEX.mygeohash.originLatitude='xx.xxxxxx','SPATIAL_INDEX.mygeohash.gridSize='xx','SP
ATIAL_INDEX.mygeohash.minLongitude='xxx.xxxxxx','SPATIAL_INDEX.mygeohash.maxLongitude='xxx.xxxxxx',
'SPATIAL_INDEX.mygeohash.minLatitude='xx.xxxxxx','SPATIAL_INDEX.mygeohash.maxLatitude='xxx.xxxxxx',
'SPATIAL_INDEX='mygeohash','SPATIAL_INDEX.mygeohash.conversionRatio='1000000','SORT_COLUMNS'='co
lumn1,column2,column3,latitude,longitude');
```

SPATIAL_INDEX is a user-defined index handler. This handler allows users to create new columns from the table-structure column set. The new column name is the same as that of the handler name. The **type** and **sourcecolumns** properties of the handler are mandatory. Currently, the value of **type** supports only **geohash**. Carbon provides a default implementation class that can be easily used. You can extend the default implementation class to mount the customized implementation class of **geohash**. The default handler also needs to provide the following table properties:

- **SPATIAL_INDEX.xxx.originLatitude**: specifies the origin latitude. (**Double** type.)

- **SPATIAL_INDEX.xxx.gridSize**: specifies the grid length in meters. (**Int** type.)
- **SPATIAL_INDEX.xxx.minLongitude**: specifies the minimum longitude. (**Double** type.)
- **SPATIAL_INDEX.xxx.maxLongitude**: specifies the maximum longitude. (**Double** type.)
- **SPATIAL_INDEX.xxx.minLatitude**: specifies the minimum latitude. (**Double** type.)
- **SPATIAL_INDEX.xxx.maxLatitude**: specifies the maximum latitude. (**Double** type.)
- **SPATIAL_INDEX.xxx.conversionRatio**: used to convert the small value of the longitude and latitude to an integer. (**Int** type.)

You can add your own table properties to the handlers in the above format and access them in your custom implementation class. **originLatitude**, **gridSize**, and **conversionRatio** are mandatory. Other parameters are optional in Carbon. You can use the **SPATIAL_INDEX.xxx.class** property to specify their implementation classes.

The default implementation class can generate handler column values for **sourcecolumns** in each row and support query based on the **sourcecolumns** filter criteria. The generated handler column is invisible to users. Except the **SORT_COLUMNS** table properties, no DDL commands or properties are allowed to contain the handler column.

NOTE

- By default, the generated handler column is regarded as the sorting column. If **SORT_COLUMNS** does not contain any **sourcecolumns**, add the handler column to the end of the existing **SORT_COLUMNS**. If the handler column has been specified in **SORT_COLUMNS**, its order in **SORT_COLUMNS** remains unchanged.
- If **SORT_COLUMNS** contains any **sourcecolumns** but does not contain the handler column, the handler column is automatically inserted before **sourcecolumns** in **SORT_COLUMNS**.
- If **SORT_COLUMNS** needs to contain any **sourcecolumns**, ensure that the handler column is listed before the **sourcecolumns** so that the handler column can take effect during sorting.

GeoSOT encoding:

```
CREATE TABLE carbontable(  
...  
longitude DOUBLE,  
latitude DOUBLE,  
...)  
STORED AS carbondata  
TBLPROPERTIES ('SPATIAL_INDEX'='xxx',  
'SPATIAL_INDEX.xxx.type'='geosot',  
'SPATIAL_INDEX.xxx.sourcecolumns'='longitude, latitude',  
'SPATIAL_INDEX.xxx.level'='21',  
'SPATIAL_INDEX.xxx.class'='org.apache.carbondata.geo.GeoSOTIndex')
```

Table 3-52 Parameter description

Parameter	Description
SPATIAL_INDEX	Specifies the spatial index. Its value is the same as the column name.
SPATIAL_INDEX.xxx.type	(Mandatory) The value is set to geosot .
SPATIAL_INDEX.xxx.source columns	(Mandatory) Specifies the source columns for calculating the spatial index. The value must be two existing columns of the double type.
SPATIAL_INDEX.xxx.level	(Optional) Specifies the columns for calculating the spatial index. The default value is 17 , through which you can obtain an accurate result and improve the computing performance.
SPATIAL_INDEX.xxx.class	(Optional) Specifies the implementation class of GeoSOT. The default value is org.apache.carbondata.geo.GeoSOTIndex .

Example:

```
create table geosot(
timevalue bigint,
longitude double,
latitude double)
stored as carbondata
TBLPROPERTIES ('SPATIAL_INDEX'='mygeosot',
'SPATIAL_INDEX.mygeosot.type'='geosot',
'SPATIAL_INDEX.mygeosot.level'='21', 'SPATIAL_INDEX.mygeosot.sourcecolumns'='longitude, latitude');
```

Preparing Data

- Data file 1: **geosotdata.csv**
timevalue,longitude,latitude
1575428400000,116.285807,40.084087
1575428400000,116.372142,40.129503
1575428400000,116.187332,39.979316
1575428400000,116.337069,39.951887
1575428400000,116.359102,40.154684
1575428400000,116.736367,39.970323
1575428400000,116.720179,40.009893
1575428400000,116.346961,40.13355
1575428400000,116.302895,39.930753
1575428400000,116.288955,39.999101
1575428400000,116.17609,40.129953
1575428400000,116.725575,39.981115
1575428400000,116.266922,40.179415
1575428400000,116.353706,40.156483
1575428400000,116.362699,39.942444
1575428400000,116.325378,39.963129
- Data file 2: **geosotdata2.csv**
timevalue,longitude,latitude
1575428400000,120.17708,30.326882
1575428400000,120.180685,30.326327
1575428400000,120.184976,30.327105
1575428400000,120.189311,30.327549
1575428400000,120.19446,30.329698
1575428400000,120.186965,30.329133

```

1575428400000,120.177481,30.328911
1575428400000,120.169713,30.325614
1575428400000,120.164563,30.322243
1575428400000,120.171558,30.319613
1575428400000,120.176365,30.320687
1575428400000,120.179669,30.323688
1575428400000,120.181001,30.320761
1575428400000,120.187094,30.32354
1575428400000,120.193574,30.323651
1575428400000,120.186192,30.320132
1575428400000,120.190055,30.317464
1575428400000,120.195376,30.318094
1575428400000,120.160786,30.317094
1575428400000,120.168211,30.318057
1575428400000,120.173618,30.316612
1575428400000,120.181001,30.317316
1575428400000,120.185162,30.315908
1575428400000,120.192415,30.315871
1575428400000,120.161902,30.325614
1575428400000,120.164306,30.328096
1575428400000,120.197093,30.325985
1575428400000,120.19602,30.321651
1575428400000,120.198638,30.32354
1575428400000,120.165421,30.314834

```

Importing Data

The GeoHash default implementation class extends the customized index abstract class. If the handler property is not set to a customized implementation class, the default implementation class is used. You can extend the default implementation class to mount the customized implementation class of **geohash**. The methods of the customized index abstract class are as follows:

- **Init** method: Used to extract, verify, and store the handler property. If the operation fails, the system throws an exception and displays the error information.
- **Generate** method: Used to generate indexes. It generates an index for each row of data.
- **Query** method: Used to generate an index value range list for given input.

The commands for importing data are the same as those for importing common Carbon tables.

```
LOAD DATA inpath '/tmp/geosotdata.csv' INTO TABLE geosot OPTIONS
('DELIMITER'=',');
```

```
LOAD DATA inpath '/tmp/geosotdata2.csv' INTO TABLE geosot OPTIONS
('DELIMITER'=',');
```

NOTE

For details about **geosotdata.csv** and **geosotdata2.csv**, see [Preparing Data](#).

Aggregate Query of Irregular Spatial Sets

Query statements and filter UDFs

- Filtering data based on polygon
IN_POLYGON(pointList)
UDF input parameter

Parameter	Type	Description
pointList	String	Enter multiple points as a string. Each point is presented as longitude latitude . Longitude and latitude are separated by a space. Each pair of longitude and latitude is separated by a comma (,). The longitude and latitude values at the start and end of the string must be the same.

UDF output parameter

Parameter	Type	Description
inOrNot	Boolean	Checks whether data is in the specified polygon_list .

Example:

```
select longitude, latitude from geosot where IN_POLYGON('116.321011 40.123503, 116.137676 39.947911, 116.560993 39.935276, 116.321011 40.123503');
```

- Filtering data based on the polygon list

IN_POLYGON_LIST(polygonList, opType)

UDF input parameters

Parameter	Type	Description
polygonList	String	Inputs multiple polygons as a string. Each polygon is presented as POLYGON ((longitude1 latitude1, longitude2 latitude2, ...)) . Note that there is a space after POLYGON . Longitudes and latitudes are separated by spaces. Each pair of longitude and latitude is separated by a comma (,). The longitudes and latitudes at the start and end of a polygon must be the same. IN_POLYGON_LIST requires at least two polygons. Example: POLYGON ((116.137676 40.163503, 116.137676 39.935276, 116.560993 39.935276, 116.137676 40.163503))

Parameter	Type	Description
opType	String	Performs union, intersection, and subtraction on multiple polygons. Currently, the following operation types are supported: <ul style="list-style-type: none"> OR: A U B U C (Assume that three polygons A, B, and C are input.) AND: A ∩ B ∩ C

UDF output parameter

Parameter	Type	Description
inOrNot	Boolean	Checks whether data is in the specified polygon_list .

Example:

```
select longitude, latitude from geosot where IN_POLYGON_LIST('POLYGON ((120.176433
30.327431,120.171283 30.322245,120.181411 30.314540, 120.190509 30.321653,120.185188
30.329358,120.176433 30.327431)), POLYGON ((120.191603 30.328946,120.184179
30.327465,120.181819 30.321464, 120.190359 30.315388,120.199242 30.324464,120.191603
30.328946))', 'OR');
```

- Filtering data based on the polyline list

IN_POLYLINE_LIST(polylineList, bufferInMeter)

UDF input parameters

Parameter	Type	Description
polylineList	String	Inputs multiple polylines as a string. Each polyline is presented as LINSTRING (longitude1 latitude1, longitude2 latitude2, ...) . Note that there is a space after LINSTRING . Longitudes and latitudes are separated by spaces. Each pair of longitude and latitude is separated by a comma (,). A union will be output based on the data in multiple polylines. Example: LINSTRING (116.137676 40.163503, 116.137676 39.935276, 116.260993 39.935276)
bufferInMeter	Float	Polyline buffer distance, in meters. Right angles are used at the end to create a buffer.

UDF output parameter

Parameter	Type	Description
inOrNot	Boolean	Checks whether data is in the specified polyline_list .

Example:

```
select longitude, latitude from geosot where IN_POLYLINE_LIST('LINESTRING (120.184179 30.327465, 120.191603 30.328946, 120.199242 30.324464, 120.190359 30.315388)', 65);
```

- Filtering data based on the GeoID range list

IN_POLYGON_RANGE_LIST(polygonRangeList, opType)

UDF input parameters

Parameter	Type	Description
polygonRangeList	String	Inputs multiple rangeLists as a string. Each rangeList is presented as RANGELIST (startGeoid1 endGeoid1, startGeoid2 endGeoid2, ...) . Note that there is a space after RANGELIST . Start Geoids and end Geoids are separated by spaces. Each group of Geoid ranges is separated by a comma (,). Example: RANGELIST (855279368848 855279368850, 855280799610 855280799612, 855282156300 855282157400)
opType	String	Performs union, intersection, and subtraction on multiple rangeLists. Currently, the following operation types are supported: <ul style="list-style-type: none"> • OR: A U B U C (Assume that three rangeLists A, B, and C are input.) • AND: A ∩ B ∩ C

UDF output parameter

Parameter	Type	Description
inOrNot	Boolean	Checks whether data is in the specified polyRange_list .

Example:

```
select mygeosot, longitude, latitude from geosot where IN_POLYGON_RANGE_LIST('RANGELIST
(526549722865860608 526549722865860618, 532555655580483584 532555655580483594)', 'OR');
```

- Performing polygon query

IN_POLYGON_JOIN(GEO_HASH_INDEX_COLUMN, POLYGON_COLUMN)

Perform join query on two tables. One is a spatial data table containing the longitude, latitude, and GeoHashIndex columns, and the other is a dimension table that saves polygon data.

During query, **IN_POLYGON_JOIN UDF**, **GEO_HASH_INDEX_COLUMN**, and **POLYGON_COLUMN** of the polygon table are used. **Polygon_column** specifies the column containing multiple points (longitude and latitude pairs). The first and last points in each row of the Polygon table must be the same. All points in each row form a closed geometric shape.

UDF input parameters

Parameter	Type	Description
GEO_HASH_INDEX_COLUMN	Long	GeoHashIndex column of the spatial data table.
POLYGON_COLUMN	String	Polygon column of the polygon table, the value of which is represented by the string of polygon, for example, POLYGON ((longitude1 latitude1, longitude2 latitude2, ...)) .

Example:

```
CREATE TABLE polygonTable(
polygon string,
poiType string,
poiid String)
STORED AS carbondata;

insert into polygonTable select 'POLYGON ((120.176433 30.327431,120.171283 30.322245,
120.181411 30.314540,120.190509 30.321653,120.185188 30.329358,120.176433 30.327431))','abc','1';

insert into polygonTable select 'POLYGON ((120.191603 30.328946,120.184179 30.327465,
120.181819 30.321464,120.190359 30.315388,120.199242 30.324464,120.191603 30.328946))','abc','2';

select t1.longitude,t1.latitude from geosot t1
inner join
(select polygon,poiid from polygonTable where poiType='abc') t2
on in_polygon_join(t1.mygeosot,t2.polygon) group by t1.longitude,t1.latitude;
```

- Performing range_list query

IN_POLYGON_JOIN_RANGE_LIST(GEO_HASH_INDEX_COLUMN, POLYGON_COLUMN)

Use the **IN_POLYGON_JOIN_RANGE_LIST** UDF to associate the spatial data table with the polygon dimension table based on **Polygon_RangeList**. By using a range list, you can skip the conversion between a polygon and a range list.

UDF input parameters

Parameter	Type	Description
GEO_HASH_INDEX_COLUMN	Long	GeoHashIndex column of the spatial data table.
POLYGON_COLUMN	String	Rangelist column of the Polygon table, the value of which is represented by the string of rangeList, for example, RANGELIST (startGeoid1 endGeoid1, startGeoid2 endGeoid2, ...) .

Example:

```
CREATE TABLE polygonTable(
polygon string,
poiType string,
poild String)
STORED AS carbondata;

insert into polygonTable select 'RANGELIST (526546455897309184 526546455897309284,
526549831217315840 526549831217315850, 532555655580483534 532555655580483584)', 'xyz', '2';

select t1.*
from geosot t1
inner join
(select polygon, poild from polygonTable where poiType='xyz') t2
on in_polygon_join_range_list(t1.mygeosot, t2.polygon);
```

UDFs of spatial index tools

- Obtaining row number and column number of a grid converted from Geoid
GeoidToGridXy(geoid)

UDF input parameter

Parameter	Type	Description
geoid	Long	Calculates the row number and column number of the grid based on Geoid.

UDF output parameter

Parameter	Type	Description
gridArray	Array[Int]	Returns the grid row and column numbers contained in Geoid in array. The first digit indicates the row number, and the second digit indicates the column number.

Example:


```
select longitude, latitude, mygeohash, GeoldToGridXy(mygeohash) as GridXY from geoTable;
```

- Converting longitude and latitude to Geold

LatLngToGeold(latitude, longitude oriLatitude, gridSize)

UDF input parameters

Parameter	Type	Description
longitude	Long	Longitude. Note: The value is an integer after conversion.
latitude	Long	Latitude. Note: The value is an integer after conversion.
oriLatitude	Double	Origin latitude, required for calculating Geold.
gridSize	Int	Grid size, required for calculating Geold.

UDF output parameter

Parameter	Type	Description
geold	Long	Returns a number that indicates the longitude and latitude after coding.

Example:

```
select longitude, latitude, mygeohash, LatLngToGeold(latitude, longitude, 39.832277, 50) as geold from geoTable;
```

- Converting Geold to longitude and latitude

GeoldToLatLng(geold, oriLatitude, gridSize)

UDF input parameters

Parameter	Type	Description
geold	Long	Calculates the longitude and latitude based on Geold.
oriLatitude	Double	Origin latitude, required for calculating the longitude and latitude.
gridSize	Int	Grid size, required for calculating the longitude and latitude.

 **NOTE**

GeoID is generated based on the grid coordinates, which are the grid center. Therefore, the calculated longitude and latitude are the longitude and latitude of the grid center. There may be an error ranging from 0 degree to half of the grid size between the calculated longitude and latitude and the longitude and latitude of the generated GeoID.

UDF output parameter

Parameter	Type	Description
latitudeAndLongitude	Array[Double]	Returns the longitude and latitude coordinates of the grid center that represent the GeoID in array. The first digit indicates the latitude, and the second digit indicates the longitude.

Example:

```
select longitude, latitude, mygeohash, GeoidToLatLng(mygeohash, 39.832277, 50) as LatitudeAndLongitude from geoTable;
```

- Calculating the upper-layer GeoID of the pyramid model

ToUpperLayerGeoid(geoid)

UDF input parameter

Parameter	Type	Description
geoid	Long	Calculates the upper-layer GeoID of the pyramid model based on the input GeoID.

UDF output parameter

Parameter	Type	Description
geoid	Long	Returns the upper-layer GeoID of the pyramid model.

Example:

```
select longitude, latitude, mygeohash, ToUpperLayerGeoid(mygeohash) as upperLayerGeoid from geoTable;
```

- Obtaining the GeoID range list using the input polygon

ToRangeList(polygon, oriLatitude, gridSize)

UDF input parameters

Parameter	Type	Description
polygon	String	Input polygon string, which is a pair of longitude and latitude. Longitude and latitude are separated by a space. Each pair of longitude and latitude is separated by a comma (,). The longitude and latitude at the start and end must be the same.
oriLatitude	Double	Origin latitude, required for calculating GeoID.
gridSize	Int	Grid size, required for calculating GeoID.

UDF output parameter

Parameter	Type	Description
geoidList	Buffer[Array[Long]]	Converts polygons into GeoID range lists.

Example:

```
select ToRangeList('116.321011 40.123503, 116.137676 39.947911, 116.560993 39.935276, 116.321011 40.123503', 39.832277, 50) as rangeList from geoTable;
```

- Calculating the upper-layer longitude of the pyramid model

ToUpperLongitude (longitude, gridSize, oriLat)

UDF input parameters

Parameter	Type	Description
longitude	Long	Input longitude, which is a long integer.
gridSize	Int	Grid size, required for calculating longitude.
oriLatitude	Double	Origin latitude, required for calculating longitude.

UDF output parameter

Parameter	Type	Description
longitude	Long	Returns the upper-layer longitude.

Example:

```
select ToUpperLongitude (-23575161504L, 50, 39.832277) as upperLongitude from geoTable;
```

- Calculating the upper-layer latitude of the pyramid model

ToUpperLatitude(Latitude, gridSize, oriLat)

UDF input parameters

Parameter	Type	Description
latitude	Long	Input latitude, which is a long integer.
gridSize	Int	Grid size, required for calculating latitude.
oriLatitude	Double	Origin latitude, required for calculating latitude.

UDF output parameter

Parameter	Type	Description
Latitude	Long	Returns the upper-layer latitude.

Example:

```
select ToUpperLatitude (-23575161504L, 50, 39.832277) as upperLatitude from geoTable;
```

- Converting longitude and latitude to GeoSOT

LatLngToGridCode(latitude, longitude, level)

UDF input parameters

Parameter	Type	Description
latitude	Double	Latitude.
longitude	Double	Longitude.
level	Int	Level. The value range is [0, 32].

UDF output parameter

Parameter	Type	Description
geold	Long	A number that indicates the longitude and latitude after GeoSOT encoding.

Example:

```
select LatLngToGridCode(39.930753, 116.302895, 21) as geold;
```

3.7 CarbonData Troubleshooting

3.7.1 Filter Result Is not Consistent with Hive when a Big Double Type Value Is Used in Filter

Symptom

When double data type values with higher precision are used in filters, incorrect values are returned by filtering results.

Possible Causes

When double data type values with higher precision are used in filters, values are rounded off before comparison. Therefore, values of double data type with different fraction part are considered same.

Troubleshooting Method

NA.

Procedure

To avoid this problem, use decimal data type when high precision data comparisons are required, such as financial applications, equality and inequality checks, and rounding operations.

Reference Information

NA.

3.7.2 Query Performance Deterioration

Symptom

The query performance fluctuates when the query is executed in different query periods.

Possible Causes

During data loading, the memory configured for each executor program instance may be insufficient, resulting in more Java GCs. When GC occurs, the query performance deteriorates.

Troubleshooting Method

On the Spark UI, the GC time of some executors is obviously higher than that of other executors, or all executors have high GC time.

Procedure

Log in to Manager and choose **Cluster > Services > Spark2x**. On the displayed page, click the **Configurations** tab and then **All Configurations**, search for **spark.executor.memory** in the search box, and set its value to a larger value.



The screenshot shows a configuration interface with a light blue background. On the left, the text 'spark.executor.memory' is displayed. To its right is a white input field containing the value '4G'.

Reference

None

3.8 CarbonData FAQ

3.8.1 Why Is Incorrect Output Displayed When I Perform Query with Filter on Decimal Data Type Values?

Question

Why is incorrect output displayed when I perform query with filter on decimal data type values?

For example:

```
select * from carbon_table where num = 1234567890123456.22;
```

Output:

```
+-----+-----+-----+
| name |    num    |
+-----+-----+-----+
| IAA | 1234567890123456.22 |
| IAA | 1234567890123456.21 |
+-----+-----+-----+
```

Answer

To obtain accurate output, append BD to the number.

For example:

```
select * from carbon_table where num = 1234567890123456.22BD;
```

Output:

```
+-----+-----+-----+
| name |    num    |
+-----+-----+-----+
| IAA | 1234567890123456.22 |
+-----+-----+-----+
```

3.8.2 How to Avoid Minor Compaction for Historical Data?

Question

How to avoid minor compaction for historical data?

Answer

If you want to load historical data first and then the incremental data, perform following steps to avoid minor compaction of historical data:

1. Load all historical data.
2. Configure the major compaction size to a value smaller than the segment size of historical data.
3. Run the major compaction once on historical data so that these segments will not be considered later for minor compaction.
4. Load the incremental data.
5. You can configure the minor compaction threshold as required.

For example:

1. Assume that you have loaded all historical data to CarbonData and the size of each segment is 500 GB.
2. Set the threshold of major compaction property to **carbon.major.compaction.size = 491520** (480 GB x 1024).
3. Run major compaction. All segments will be compacted because the size of each segment is more than configured size.
4. Perform incremental loading.
5. Configure the minor compaction threshold to **carbon.compaction.level.threshold = 6,6**.
6. Run minor compaction. As a result, only incremental data is compacted.

3.8.3 How to Change the Default Group Name for CarbonData Data Loading?

Question

How to change the default group name for CarbonData data loading?

Answer

By default, the group name for CarbonData data loading is **ficommon**. You can perform the following operation to change the default group name:

1. Edit the **carbon.properties** file.
2. Change the value of the key **carbon.dataload.group.name** as required. The default value is **ficommon**.

3.8.4 Why Does INSERT INTO CARBON TABLE Command Fail?

Question

Why does the *INSERT INTO CARBON TABLE* command fail and the following error message is displayed?

```
Data load failed due to bad record
```

Answer

The *INSERT INTO CARBON TABLE* command fails in the following scenarios:

- If the data type of source and target table columns are not the same, the data from the source table will be treated as bad records and the *INSERT INTO* command fails.
- If the result of aggregation function on a source column exceeds the maximum range of the target column, then the *INSERT INTO* command fails.

Solution:

You can use the cast function on corresponding columns when inserting records.

For example:

- a. Run the *DESCRIBE* command to query the target and source table.

```
DESCRIBE newcarbontable;
```

Result:

```
col1 int  
col2 bigint
```

```
DESCRIBE sourcetable;
```

Result:

```
col1 int  
col2 int
```

- b. Add the cast function to convert bigint value to integer.

```
INSERT INTO newcarbontable select col1, cast(col2 as integer) from  
sourcetable;
```

3.8.5 Why Is the Data Logged in Bad Records Different from the Original Input Data with Escape Characters?

Question

Why is the data logged in bad records different from the original input data with escaped characters?

Answer

An escape character is a backslash (\) followed by one or more characters. If the input records contain escape characters such as \t, \b, \n, \r, \f, \', \", \\, java will process the escape character '\' and the following characters together to obtain the escaped meaning.

For example, if the CSV data type `2010\\10,test` is inserted to String,int type, the value is treated as bad records, because `test` cannot be converted to int. The value logged in the bad records is `2010\10` because java processes `\\` as `\`.

3.8.6 Why Data Load Performance Decreases due to Bad Records?

Question

Why data load performance decreases due to bad records?

Answer

If bad records are present in the data and `BAD_RECORDS_LOGGER_ENABLE` is `true` or `BAD_RECORDS_ACTION` is `redirect` then load performance will decrease due to extra I/O for writing failure reason in log file or redirecting the records to raw CSV.

3.8.7 Why INSERT INTO/LOAD DATA Task Distribution Is Incorrect and the Opened Tasks Are Less Than the Available Executors when the Number of Initial Executors Is Zero?

Question

Why `INSERT INTO` or `LOAD DATA` task distribution is incorrect, and the opened tasks are less than the available executors when the number of initial executors is zero?

Answer

In case of `INSERT INTO` or `LOAD DATA`, CarbonData distributes one task per node. If the executors are not allocated from the distinct nodes then CarbonData will launch fewer tasks.

Solution:

Configure higher value for the executor memory and core so that the yarn can launch only one executor per node.

1. Configure the number of the Executor cores.
 - Configure the `spark.executor.cores` in `spark-defaults.conf` or the `SPARK_EXECUTOR_CORES` in `spark-env.sh` appropriately.
 - Add `--executor-cores NUM` parameter to configure the cores during use the `spark-submit` command.
2. Configure the Executor memory.
 - Configure the `spark.executor.memory` in `spark-defaults.conf` or the `SPARK_EXECUTOR_MEMORY` in `spark-env.sh` appropriately.
 - Add `--executor-memory MEM` parameter to configure the memory during use the `spark-submit` command.

3.8.8 Why Does CarbonData Require Additional Executors Even Though the Parallelism Is Greater Than the Number of Blocks to Be Processed?

Question

Why does CarbonData require additional executors even though the parallelism is greater than the number of blocks to be processed?

Answer

CarbonData block distribution optimizes data processing as follows:

1. Optimize data processing parallelism.
2. Optimize parallel reading of block data.

To optimize parallel processing and parallel read, CarbonData requests executors based on the locality of blocks so that it can obtain executors on all nodes.

If you are using dynamic allocation, you need to configure the following properties:

1. Set **spark.dynamicAllocation.executorIdleTimeout** to 15 minutes (or the average query time).
2. Set **spark.dynamicAllocation.maxExecutors** correctly. The default value **2048** is not recommended. Otherwise, CarbonData will request the maximum number of executors.
3. For a bigger cluster, set **carbon.dynamicAllocation.schedulerTimeout** to a value ranging from 10 to 15 seconds. The default value is 5 seconds.
4. Set **carbon.scheduler.minRegisteredResourcesRatio** to a value ranging from 0.1 to 1.0. The default value is **0.8**. Block distribution can be started as long as the value of **carbon.scheduler.minRegisteredResourcesRatio** is within the range.

3.8.9 Why Data loading Fails During off heap?

Question

Why Data Loading fails during off heap?

Answer

YARN Resource Manager will consider (Java heap memory + **spark.yarn.am.memoryOverhead**) as memory limit, so during the off heap, the memory can exceed this limit. So you need to increase the memory by increasing the value of the parameter **spark.yarn.am.memoryOverhead**.

3.8.10 Why Do I Fail to Create a Hive Table?

Question

Why do I fail to create a hive table?

Answer

Creating a Hive table fails, when source table or sub query has more number of partitions. The implementation of the query requires a lot of tasks, then the number of files will be output a lot, resulting OOM in Driver.

It can be solved by using ***distribute by*** on suitable cardinality(distinct values) column in the statement of Hive table creation.

distribute by clause limits number of hive table partitions. It considers cardinality of given column or **`spark.sql.shuffle.partitions`** which ever is minimal. For example, if **`spark.sql.shuffle.partitions`** is 200, but cardinality of column is 100, out files is 200, but the other 100 files are empty. So using very low cardinality column like 1 will cause data skew and will effect later query distribution.

So we suggest using the column with cardinality greater than **`spark.sql.shuffle.partitions`**. It can be greater than 2 to 3 times.

Example:

```
create table hivetable1 as select * from sourcetable1 distribute by col_age;
```

3.8.11 Why CarbonData tables created in V100R002C50RC1 not reflecting the privileges provided in Hive Privileges for non-owner?

Question

Why CarbonData tables created in V100R002C50RC1 not reflecting the privileges provided in Hive Privileges for non-owner?

Answer

The Hive ACL is implemented after the version V100R002C50RC1, hence the Hive ACL Privileges are not reflecting.

To support HIVE ACL Privileges for CarbonData tables created in V100R002C50RC1, following two ALTER TABLE commands must be executed by owner of the table.

```
ALTER TABLE $dbname.$tablename SET LOCATION '$carbon.store/$dbname/$tablename';
```

```
ALTER TABLE $dbname.$tablename SET SERDEPROPERTIES ('path'='$carbon.store/$dbname/$tablename');
```

Example:

Assume database name is 'carbondb', table name is 'carbontable', and CarbonData store location is 'hdfs://hacluster/user/hive/warehouse/carbon.store', then the commands should be executed is as follows:

```
ALTER TABLE carbondb.carbontable SET LOCATION 'hdfs://hacluster/user/hive/warehouse/carbon.store/carbondb/carbontable';
```

```
ALTER TABLE carbondb.carbontable SET SERDEPROPERTIES ('path'='hdfs://hacluster/user/hive/warehouse/carbon.store/carbondb/carbontable');
```

3.8.12 How Do I Logically Split Data Across Different Namespaces?

Question

How do I logically split data across different namespaces?

Answer

- Configuration:
To logically split data across different namespaces, you must update the following configuration in the **core-site.xml** file of HDFS, Hive, and Spark.

NOTE

Changing the Hive component will change the locations of carbonstore and warehouse.

- Configuration in HDFS
 - **fs.defaultFS**: Name of the default file system. The URI mode must be set to **viewfs**. When **viewfs** is used, the permission part must be **ClusterX**.
 - **fs.viewfs.mountable.ClusterX.homedir**: Home directory base path. You can use the `getHomeDirectory()` method defined in **FileSystem/FileContext** to access the home directory.
 - **fs.viewfs.mountable.default.link.<dir_name>**: ViewFS mount table.

Example:

```
<property>
<name>fs.defaultFS</name>
<value>viewfs://ClusterX</value>
</property>
<property>
<name>fs.viewfs.mounttable.ClusterX.link./folder1</name>
<value>hdfs://NS1/folder1</value>
</property>
<property>
<name>fs.viewfs.mounttable.ClusterX.link./folder2</name>
<value>hdfs://NS2/folder2</value>
</property>
```

- Configurations in Hive and Spark
 - fs.defaultFS**: Name of the default file system. The URI mode must be set to **viewfs**. When **viewfs** is used, the permission part must be **ClusterX**.

- Syntax:

```
LOAD DATA INPATH 'path to data' INTO TABLE table_name OPTIONS ('...');
```

 NOTE

When Spark is configured with the viewFS file system and attempts to load data from HDFS, users must specify a path such as **viewfs://** or a relative path as the file path in the **LOAD** statement.

- Example:

- Sample viewFS path:

```
LOAD DATA INPATH 'viewfs://ClusterX/dir/data.csv' INTO TABLE  
table_name OPTIONS ('...');
```

- Sample relative path:

```
LOAD DATA INPATH '/apps/input_data1.txt' INTO TABLE table_name;
```

3.8.13 Why Missing Privileges Exception is Reported When I Perform Drop Operation on Databases?

Question

Why drop database cascade is throwing the following exception?

```
Error: org.apache.spark.sql.AnalysisException: Missing Privileges;(State=,code=0)
```

Answer

This error is thrown when the owner of the database performs **drop database <database_name> cascade** which contains tables created by other users.

3.8.14 Why the UPDATE Command Cannot Be Executed in Spark Shell?

Question

Why the UPDATE command cannot be executed in Spark Shell?

Answer

The syntax and examples provided in this document are about Beeline commands instead of Spark Shell commands.

To run the UPDATE command in Spark Shell, use the following syntax:

- Syntax 1

```
<carbon_context>.sql("UPDATE <CARBON TABLE> SET (column_name1,  
column_name2, ... column_name n) = (column1_expression ,  
column2_expression , column3_expression ... column n_expression)  
[ WHERE { <filter_condition> } ]");.show
```

- Syntax 2

```
<carbon_context>.sql("UPDATE <CARBON TABLE> SET (column_name1,  
column_name2,) = (select sourceColumn1, sourceColumn2 from  
sourceTable [ WHERE { <filter_condition> } ] ) [ WHERE  
{ <filter_condition> } ]");.show
```

Example:

If the context of CarbonData is **carbon**, run the following command:

```
carbon.sql("update carbonTable1 d set (d.column3,d.column5) = (select s.c33 ,s.c55 from sourceTable1 s where d.column1 = s.c11) where d.column1 = 'country' exists( select * from table3 o where o.c2 > 1);").show
```

3.8.15 How Do I Configure Unsafe Memory in CarbonData?

Question

How do I configure unsafe memory in CarbonData?

Answer

In the Spark configuration, the value of **spark.yarn.executor.memoryOverhead** must be greater than the sum of (**sort.inmemory.size.inmb + Netty offheapmemory required**), or the sum of (**carbon.unsafe.working.memory.in.mb + carbon.sort.inmemory.storage.size.in.mb + Netty offheapmemory required**). Otherwise, if off-heap access exceeds the configured executor memory, Yarn may stop the executor.

If **spark.shuffle.io.preferDirectBufs** is set to **true**, the netty transfer service in Spark takes off some heap memory (around 384 MB or 0.1 x executor memory) from **spark.yarn.executor.memoryOverhead**.

For details, see [Configuring Executor Off-Heap Memory](#).

3.8.16 Why Exception Occurs in CarbonData When Disk Space Quota is Set for Storage Directory in HDFS?

Question

Why exception occurs in CarbonData when Disk Space Quota is set for the storage directory in HDFS?

Answer

The data will be written to HDFS when you during create table, load table, update table, and so on. If the configured HDFS directory does not have sufficient disk space quota, then the operation will fail and throw following exception.

```
org.apache.hadoop.hdfs.protocol.DSQuotaExceededException:  
The DiskSpace quota of /user/tenant is exceeded:  
quota = 314572800 B = 300 MB but disk space consumed = 402653184 B = 384 MB at  
org.apache.hadoop.hdfs.server.namenode.DirectoryWithQuotaFeature.verifyStoragespaceQuota(DirectoryWith  
hQuotaFeature.java:211) at  
org.apache.hadoop.hdfs.server.namenode.DirectoryWithQuotaFeature.verifyQuota(DirectoryWithQuotaFeatu  
re.java:239) at  
org.apache.hadoop.hdfs.server.namenode.FSDirectory.verifyQuota(FSDirectory.java:941) at  
org.apache.hadoop.hdfs.server.namenode.FSDirectory.updateCount(FSDirectory.java:745)
```

If such exception occurs, configure a sufficient disk space quota for the tenant.

For example:

If the HDFS replication factor is 3 and HDFS default block size is 128 MB, then at least 384 MB (no. of block x block_size x replication_factor of the schema file = 1 x 128 x 3 = 384 MB) disk space quota is required to write a table schema file to HDFS.

 NOTE

In case of fact files, as the default block size is 1024 MB, the minimum space required is 3072 MB per fact file for data load.

3.8.17 Why Does Data Query or Loading Fail and "org.apache.carbondata.core.memory.MemoryException: Not enough memory" Is Displayed?

Question

Why does data query or loading fail and "org.apache.carbondata.core.memory.MemoryException: Not enough memory" is displayed?

Answer

This exception is thrown when the out-of-heap memory required for data query and loading in the executor is insufficient.

In this case, increase the values of **carbon.unsafe.working.memory.in.mb** and **spark.yarn.executor.memoryOverhead**.

For details, see [How Do I Configure Unsafe Memory in CarbonData?](#)

The memory is shared by data query and loading. Therefore, if the loading and query operations need to be performed at the same time, you are advised to set **carbon.unsafe.working.memory.in.mb** and **spark.yarn.executor.memoryOverhead** to a value greater than 2,048 MB.

The following formula can be used for estimation:

Memory required for data loading:

$\text{carbon.number.of.cores.while.loading [default value is 6]} \times \text{Number of tables to load in parallel} \times \text{offheap.sort.chunk.size.inmb [default value is 64 MB]} + \text{carbon.blockletgroup.size.in.mb [default value is 64 MB]} + \text{Current compaction ratio [64 MB/3.5]}$

= Around 900 MB per table

Memory required for data query:

$(\text{SPARK_EXECUTOR_INSTANCES. [default value is 2]} \times (\text{carbon.blockletgroup.size.in.mb [default value: 64 MB]} + \text{carbon.blockletgroup.size.in.mb [default value = 64 MB} \times 3.5]) \times \text{Number of cores per executor [default value: 1]})$

= ~ 600 MB

4 Using ClickHouse

4.1 Using ClickHouse from Scratch

ClickHouse is a column-based database oriented to online analysis and processing. It supports SQL query and provides good query performance. The aggregation analysis and query performance based on large and wide tables is excellent, which is one order of magnitude faster than other analytical databases.

Prerequisites

The cluster client has been installed. For example, the client has been installed in the `/opt/Bigdata/client` directory. The client directory in the following operations is only an example. Change it based on the actual installation directory onsite. Before using the client, download and update the client configuration file, and ensure that the active management node of Manager is available.

Procedure

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client installation directory.

```
cd /opt/Bigdata/client
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 Run the client command of the ClickHouse component.

Run the **clickhouse -h** command to view the command help of ClickHouse.

The command output is as follows:

```
Use one of the following commands:  
clickhouse local [args]  
clickhouse client [args]  
clickhouse benchmark [args]  
clickhouse server [args]  
clickhouse performance-test [args]  
clickhouse extract-from-config [args]
```



```
clickhouse compressor [args]
clickhouse format [args]
clickhouse copier [args]
clickhouse obfuscator [args]
...
```

For details about how to use the command, see <https://clickhouse.tech/docs/en/operations/>.

The following table describes the parameters when the **clickhouse client** command is used to connect to the ClickHouse server.

Table 4-1 Parameters of the clickhouse client command

Parameter	Description
--host	Host name of the server. The default value is localhost . You can use the host name or IP address of the node where the ClickHouse instance is located.
--port	Port for connection. The default value is 9000 .
--user	Username. The default username is default .
--password	Password. The default password is an empty string.
--query	Query to process when using non-interactive mode.
--database	Current default database. The default value is default , which is the default configuration on the server.
--multiline	If this parameter is specified, multiline queries are allowed. (Enter only indicates line feed and does not indicate that the query statement is complete.)
--multiquery	If this parameter is specified, multiple queries separated with semicolons (;) can be processed. This parameter is valid only in non-interactive mode.
--format	Specified default format used to output the result.
--vertical	If this parameter is specified, the result is output in vertical format by default. In this format, each value is printed on a separate line, which helps to display a wide table.
--time	If this parameter is specified, the query execution time is printed to stderr in non-interactive mode.
--stacktrace	If this parameter is specified, stack trace information will be printed when an exception occurs.
--config-file	Name of the configuration file.
--secure	If this parameter is specified, the server will be connected in SSL mode.
--history_file	Path of files that record command history.

Parameter	Description
-- param_<name>	Query with parameters. Pass values from the client to the server. For details, see https://clickhouse.tech/docs/en/interfaces/cli/#cli-queries-with-parameters .

Run the client command of the ClickHouse component.

clickhouse client --host *IP address of the ClickHouse instance* --user *Username* --password *Password* --port *ClickHouse port number*

 NOTE

Username: The default username is **default**. If the --user parameter is not carried, user **default** is used for login.

Password: By default, this parameter is left blank. If the default value is used, the --password parameter is optional.

ClickHouse port number: The default port number is **9000**.

IP address of the ClickHouse instance: choose **Components** > **ClickHouse** > **Instances** on the cluster details page to obtain it.

----End

ClickHouse Database Operations

Creating a database

- Basic syntax

CREATE DATABASE [IF NOT EXISTS] *database_name* [ON CLUSTER *ClickHouse cluster name*]

 NOTE

The syntax **ON CLUSTER *ClickHouse cluster name*** enables the DDL statement to be executed on all instances in the cluster at a time. You can run the following statement to obtain the cluster name from the **cluster** field:

select cluster,shard_num,replica_num,host_name from system.clusters;

- Example

```
-- Create a database named test.
CREATE DATABASE test ON CLUSTER default_cluster;
-- After the creation is successful, run the query command for verification.
show databases;
```

```
name
default
system
test
```

Creating a table

- Basic syntax

- Method 1: Creating a table named **table_name** in the specified **database_name** database.

If the table creation statement does not contain **database_name**, the name of the database selected during client login is used by default.

```
CREATE TABLE [IF NOT EXISTS] [database_name.]table_name [ON CLUSTER ClickHouse cluster name]
```

```
(
  name1 [type1] [DEFAULT|MATERIALIZED|ALIAS expr1],
  name2 [type2] [DEFAULT|MATERIALIZED|ALIAS expr2],
  ...
) ENGINE= engine_name()
[PARTITION BY expr_list]
[ORDER BY expr_list]
```

- Method 2: Creating a table with the same structure as **database_name2.table_name2** and specifying a different table engine for the table

If no table engine is specified, the created table uses the same table engine as **database_name2.table_name2**.

```
CREATE TABLE [IF NOT EXISTS] [database_name.]table_name AS [database_name2.]table_name2 [ENGINE = engine_name]
```

- Method 3: Using the specified engine to create a table with the same structure as the result of the SELECT clause and filling it with the result of the SELECT clause

```
CREATE TABLE [IF NOT EXISTS] [database_name.]table_name ENGINE = engine_name AS SELECT ...
```

- Example

```
-- Create a table named test in the default database and default_cluster cluster.
CREATE TABLE default.test ON CLUSTER default_cluster
(
  `EventDate` DateTime,
  `id` UInt64
)
ENGINE = ReplicatedMergeTree('/clickhouse/tables/{shard}/default/test', '{replica}')
PARTITION BY toYYYYMM(EventDate)
ORDER BY id
```

Inserting data into a table

- Basic syntax

- Method 1: Inserting data in standard format

```
INSERT INTO [database_name.]table [(c1, c2, c3)] VALUES (v11, v12, v13), (v21, v22, v23), ...
```

- Method 2: Using the SELECT result to insert data

```
INSERT INTO [database_name.]table [(c1, c2, c3)] SELECT ...
```

- Example

```
-- Insert data into the test2 table.
insert into test2 (id, name) values (1, 'abc'), (2, 'bbbb');
-- Query data in the test2 table.
select * from test2;
```

id	name
1	abc
2	bbbb

Querying data in the table

- Basic syntax

```

SELECT [DISTINCT] expr_list
[FROM [database_name.]table | (subquery) | table_function] [FINAL]
[SAMPLE sample_coeff]
[ARRAY JOIN ...]
[GLOBAL] [ANY|ALL|ASOF] [INNER|LEFT|RIGHT|FULL|CROSS] [OUTER|SEMI|
ANTI] JOIN (subquery)|table (ON <expr_list>)|(USING <column_list>)
[PREWHERE expr]
[WHERE expr]
[GROUP BY expr_list] [WITH TOTALS]
[HAVING expr]
[ORDER BY expr_list] [WITH FILL] [FROM expr] [TO expr] [STEP expr]
[LIMIT [offset_value, ]n BY columns]
[LIMIT [n, ]m] [WITH TIES]
[UNION ALL ...]
[INTO OUTFILE filename]
[FORMAT format]

```

- Example

```

-- View ClickHouse cluster information.
select * from system.clusters;
-- View the macros set for the current node.
select * from system.macros;
-- Check the database capacity.
select
sum(rows) as "Total number of rows",
formatReadableSize(sum(data_uncompressed_bytes)) as "Original size",
formatReadableSize(sum(data_compressed_bytes)) as "Compression size",
round(sum(data_compressed_bytes) / sum(data_uncompressed_bytes) * 100,
0) "Compression rate"
from system.parts;
-- Query the capacity of the test table. Add or modify the where clause based on the site
requirements.
select
sum(rows) as "Total number of rows",
formatReadableSize(sum(data_uncompressed_bytes)) as "Original size",
formatReadableSize(sum(data_compressed_bytes)) as "Compression size",
round(sum(data_compressed_bytes) / sum(data_uncompressed_bytes) * 100,
0) "Compression rate"
from system.parts
where table in ('test')
and partition like '2020-11-%'
group by table;

```

Modifying table structure

- Basic syntax

```

ALTER TABLE [database_name].name [ON CLUSTER cluster] ADD|DROP|
CLEAR|COMMENT|MODIFY COLUMN ...

```

 **NOTE**

ALTER supports only *MergeTree, Merge, and Distributed engine tables.

- Example

```

-- Add the test01 column to the t1 table.
ALTER TABLE t1 ADD COLUMN test01 String DEFAULT 'defaultvalue';
-- Query the modified table t1.
desc t1

```


Deleting a table

- Basic syntax

```
DROP [TEMPORARY] TABLE [IF EXISTS] [database_name.]name [ON CLUSTER cluster]
```

- Example

```
-- Delete the t1 table.  
drop table t1;
```

4.2 ClickHouse Cluster Configuration

Background

ClickHouse uses the multi-shard and multi-replica deployment architecture to implement the cluster high availability. Multiple shards are defined in each cluster, and each shard has two or more replicas. If a node is faulty, replicas on other nodes in the shard can take over services from the faulty node, ensuring service continuity and improving cluster stability.

Cluster Configuration

After a ClickHouse cluster is created, the cluster sharding and replication configurations are written into the **metrika.xml** file.

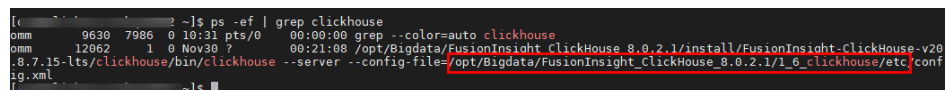
The **metrika.xml** file is in the **`\${BIGDATA_HOME}/FusionInsight_ClickHouse_Version number/x_x_ClickHouse instance name/etc** directory on the ClickHouse instance node.

To obtain the directory where the configuration file is located, perform the following steps:

1. Log in to the MRS console and click the target cluster to go to its details page.
2. Click the **Components** tab and then **ClickHouse**. On the displayed page, click the **Instances** tab to obtain the IP addresses of ClickHouse instances.
3. Log in to the node where a ClickHouse instance is located using SSH. Then, run the following command to check the ClickHouse process and obtain the directory where the ClickHouse configuration file is stored:

```
ps -ef | grep clickhouse
```

Figure 4-1 Directory of the ClickHouse configuration file



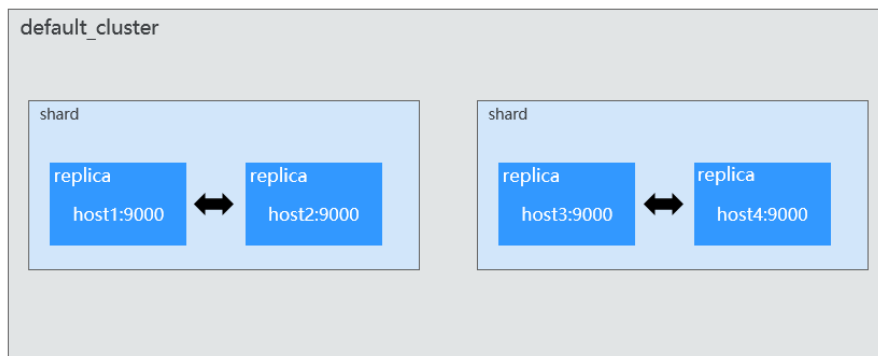
```
[root@ip-10.20.20.10 ~]# ps -ef | grep clickhouse  
omm      9630  7986      0 10:31 pts/0    00:00:00 grep --color=auto clickhouse  
omm      12062  1      0 Nov30   ?        00:21:08 /opt/Bigdata/FusionInsight_ClickHouse_8.0.2.1/install/FusionInsight_ClickHouse-v20  
8.7.15-lts-clickhouse/bin/clickhouse --server --config-file=/opt/Bigdata/FusionInsight_ClickHouse_8.0.2.1/1_6_clickhouse/etc/conf  
ig.xml  
[root@ip-10.20.20.10 ~]#
```

The **metrika.xml** file is as follows provided that the ClickHouse cluster has four instance nodes:

```
<yandex>  
<clickhouse_remote_servers>  
  <default_cluster>  
    <shard>  
      <replica>  
        <host>host1.9bf17e66-e7ed-4f21-9dfc-34575f955ae6.com</host>
```

```
<port>9000</port>
</replica>
<replica>
  <host>host2.9bf17e66-e7ed-4f21-9dfc-34575f955ae6.com</host>
  <port>9000</port>
</replica>
<internal_replication>>true</internal_replication>
</shard>
<shard>
  <replica>
    <port>9000</port>
    <host>host3.9bf17e66-e7ed-4f21-9dfc-34575f955ae6.com</host>
  </replica>
  <replica>
    <host>host4.9bf17e66-e7ed-4f21-9dfc-34575f955ae6.com</host>
    <port>9000</port>
  </replica>
<internal_replication>>true</internal_replication>
</shard>
</default_cluster>
</clickhouse_remote_servers>
<macros>
  <id>33</id>
  <replica>1</replica>
  <shard>2</shard>
</macros>
</yandex>
```

The following figure shows the cluster architecture:



The following describes the parameters:

- **default_cluster**
 - **default_cluster** indicates the name of the current cluster.
 - The current cluster has two shards. Each shard has two replicas, and each replica corresponds to a ClickHouse instance node.
 - **internal_replication** indicates whether internal replication is performed between replicas. It takes effect when data is inserted into shards through the cluster.

The default value is **true**, indicating that data is written to only one replica. (Data is synchronized between replica through replicated tables to ensure data consistency.)

If this parameter is set to **false (not recommended)**, same data is written to all replicas of the shard. (Data between replicas is not strongly consistent, and full synchronization cannot be ensured.)

- **macros**

macros indicates IDs the shard and replica where the current instance node resides. It can be used to distinguish different replicas.

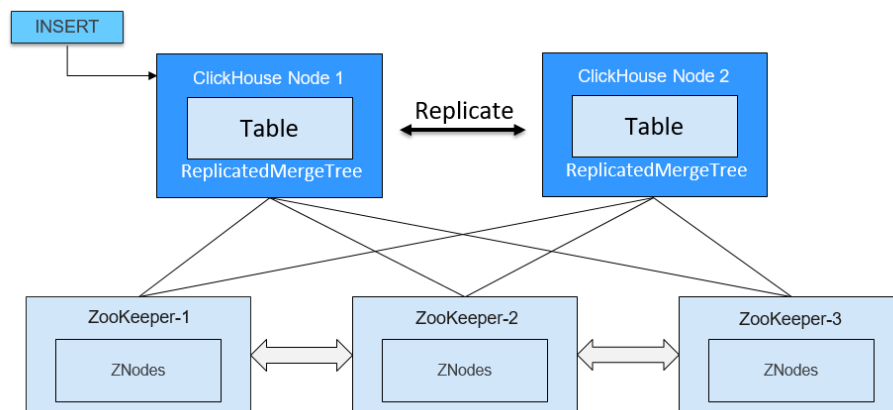
For example, the preceding example shows the configuration of the **host3** instance. The shard ID of the instance is **2** and the replica ID is **1**.

This section describes how to configure sharding and replication. For details about how to synchronize data between replicas in the ClickHouse cluster, see [Replication](#).

Replication

ClickHouse uses ZooKeeper and the ReplicatedMergeTree engine (of Replicated series) to implement replication. Replication uses a multi-master scheme. The INSERT statement can be sent to any replica, and data is replicated to other replicas in the shard asynchronously.

In the following figure, Node 1 and Node 2 correspond to **host1** and **host2** in [Cluster Configuration](#).



After the ClickHouse cluster is successfully created, three ZooKeeper nodes are created by default. ZooKeeper stores the metadata of the ClickHouse table during replication.

For details about the ZooKeeper node information, see the **config.xml** file in the **\$ {BIGDATA_HOME}/FusionInsight_ClickHouse_Version number/x_x_ClickHouse instance name/etc** directory.

```
<yandex>
...
<zookeeper>
<node index="1">
<host>node-master1lrgj.9bf17e66-e7ed-4f21-9dfc-34575f955ae6.com</host>
<port>2181</port>
</node>
<node index="2">
<port>2181</port>
<host>node-master2vocd.9bf17e66-e7ed-4f21-9dfc-34575f955ae6.com</host>
</node>
<node index="3">
<host>node-master3xwmu.9bf17e66-e7ed-4f21-9dfc-34575f955ae6.com</host>
<port>2181</port>
</node>
```



```
</zookeeper>  
...
```

For details about how to use the cluster after configuration, see [Creating and Using ClickHouse Replicated Tables and Distributed Tables](#).

4.3 Setting the ClickHouse Username and Password

After a ClickHouse cluster is created, you can use the ClickHouse client to connect to the ClickHouse server. The default username is **default**, and the password is empty.

This section describes how to set ClickHouse username and password after a ClickHouse cluster is successfully created.

Prerequisites

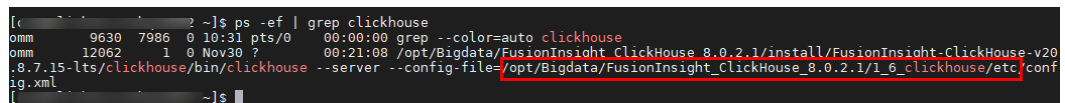
The cluster client has been installed. For example, the client has been installed in the `/opt/Bigdata/client` directory. The client directory in the following operations is only an example. Change it based on the actual installation directory onsite.

Setting the ClickHouse Username and Password

- Step 1** Log in to the MRS console and click the target cluster to go to its details page.
- Step 2** On the MRS cluster details page, click the **Components** tab and then **ClickHouse**. On the displayed page, click the **Instances** tab to obtain the IP addresses of ClickHouse instances.
- Step 3** Log in to the node where a ClickHouse instance is located using SSH. Then, run the following command to check the ClickHouse process and obtain the directory where the ClickHouse configuration file is stored:

```
ps -ef | grep clickhouse
```

Figure 4-2 Directory of the ClickHouse configuration file



```
[c ~]# ps -ef | grep clickhouse  
omm      9630  7986  0 10:31 pts/0    00:00:00 grep --color=auto clickhouse  
omm      12062  1  0 Nov30 ?        00:21:08 /opt/Bigdata/FusionInsight_ClickHouse_8.0.2.1/install/FusionInsight_ClickHouse-v2018.7.15-lts/clickhouse/bin/clickhouse --server --config-file=/opt/Bigdata/FusionInsight_ClickHouse_8.0.2.1/1_6_clickhouse/etc/config.xml  
[c ~]#
```

- Step 4** Run the following command to access the directory for storing the configuration file.

```
cd ${BIGDATA_HOME}/FusionInsight_ClickHouse_*/x_x_ClickHouse instance name/etc
```

Example: `cd ${BIGDATA_HOME}/FusionInsight_ClickHouse_8.0.2.1/1_6_clickhouse/etc`

- Step 5** Run the following commands to obtain the content of the **users.xml** and **metrika.xml** configuration files:

Run the following command to obtain the content of the **users.xml** configuration file:

```
cat users.xml
```

Run the following command to obtain the content of the **metrika.xml** configuration file:

```
cat metrika.xml
```

- Step 6** Log in to FusionInsight Manager of the ClickHouse cluster. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
- Step 7** Choose **Cluster > Services > ClickHouse**. On the displayed page, click the **Configurations** tab and then **All Configurations**. Select the **_user-xml-content** parameter under **ClickHouse**.
- Step 8** Copy the content of the **users.xml** configuration file obtained in [Step 5](#) to the value of the **_user-xml-content** parameter and change the value of **<password>**. Click **Save**.

In the following example, the password of user **default** is changed to **Password_123**. **Note: Change the password based on the site requirements.**

```
<yandex>
  <users>
    <default>
      <profile>default</profile>
      <networks>
        <ip>::/0</ip>
      </networks>
      <quota>default</quota>
      <password>Password_123</password>
    </default>
  </users>
</yandex>
```

 **NOTE**

When changing the **password**, you only need to change the value of password. Retain the values of other parameters.

- Step 9** On FusionInsight Manager, choose **Cluster > Services > ClickHouse**. On the displayed page, click the **Configurations** tab and then **All Configurations**. Select the **_metrika-xml-content** parameter under **ClickHouse**.
- Step 10** Copy the content of the **metrika.xml** configuration file obtained in [Step 5](#) to the value of the **_metrika-xml-content** parameter. Then, add the username and password to the parameter configurations by referring to the following example. Click **Save**.

Note: The following configurations are for reference only.

Before the change, user **default** does not have a password. Therefore, the configuration file does not contain the username and password.

```
<yandex>
  <clickhouse_remote_servers>
    <default_cluster>
      <shard>
        <replica>
          <host>ClickHouseIiKz0001</host>
          <port>9000</port>
        </replica>
        <replica>
          <host>ClickHouseIiKz0002</host>
          <port>9000</port>
        </replica>
      <internal_replication>true</internal_replication>
    </default_cluster>
  </clickhouse_remote_servers>
</yandex>
```

```
</shard>  
</default_cluster>  
...  
</yandex>
```

After the change, the **<user>** and **<password>** parameters are added under **<replica>**. The value of **<password>** is the new password.

```
<yandex>  
<clickhouse_remote_servers>  
<default_cluster>  
<shard>  
<replica>  
<host>ClickHouselikz0001</host>  
<port>9000</port>  
<user>default</user>  
<password>Password_123</password>  
</replica>  
<replica>  
<host>ClickHouselikz0002</host>  
<port>9000</port>  
<user>default</user>  
<password>Password_123</password>  
</replica>  
<internal_replication>true</internal_replication>  
</shard>  
</default_cluster>  
...  
</yandex>
```

Step 11 Log in to the node where the client is installed and run the following command to switch to the client installation directory.

```
cd /opt/Bigdata/client
```

Step 12 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 13 Log in to ClickHouse using the new password.

```
clickhouse client --host IP address of the ClickHouse instance --user default --password Password_123
```

 **NOTE**

To obtain the IP address of the ClickHouse instance, choose **Components > ClickHouse > Instances** on the cluster details page.

----End

4.4 ClickHouse Table Engine Overview

Background

Table engines play a key role in ClickHouse to determine:

- Where to write and read data
- Supported query modes
- Whether concurrent data access is supported
- Whether indexes can be used

- Whether multi-thread requests can be executed
- Parameters used for data replication

This section describes MergeTree and Distributed engines, which are the most important and frequently used ClickHouse table engines.

For details about other table engines, visit <https://clickhouse.tech/docs/en/engines/table-engines>.

MergeTree Family

Engines of the MergeTree family are the most universal and functional table engines for high-load tasks. They have the following key features:

- Data is stored by partition and block based on partitioning keys.
- Data index is sorted based on primary keys and the **ORDER BY** sorting keys.
- Data replication is supported by table engines prefixed with Replicated.
- Data sampling is supported.

When data is written, a table with this type of engine divides data into different folders based on the partitioning key. Each column of data in the folder is an independent file. A file that records serialized index sorting is created. This structure reduces the volume of data to be retrieved during data reading, greatly improving query efficiency.

- MergeTree

Syntax for creating a table:

```
CREATE TABLE [IF NOT EXISTS] [db.]table_name [ON CLUSTER cluster]
(
  name1 [type1] [DEFAULT|MATERIALIZED|ALIAS expr1] [TTL expr1],
  name2 [type2] [DEFAULT|MATERIALIZED|ALIAS expr2] [TTL expr2],
  ...
  INDEX index_name1 expr1 TYPE type1(...) GRANULARITY value1,
  INDEX index_name2 expr2 TYPE type2(...) GRANULARITY value2
) ENGINE = MergeTree()
ORDER BY expr
[PARTITION BY expr]
[PRIMARY KEY expr]
[SAMPLE BY expr]
[TTL expr [DELETE|TO DISK 'xxx'|TO VOLUME 'xxx'], ...]
[SETTINGS name=value, ...]
```

Example:

```
CREATE TABLE default.test (
  name1 DateTime,
  name2 String,
  name3 String,
  name4 String,
  name5 Date,
  ...
) ENGINE = MergeTree()
PARTITION BY toYYYYMM(name5)
ORDER BY (name1, name2)
SETTINGS index_granularity = 8192
```

Parameters in the example are described as follows:

- **ENGINE = MergeTree()**: specifies the MergeTree engine.
- **PARTITION BY toYYYYMM(name4)**: specifies the partition. The sample data is partitioned by month, and a folder is created for each month.

- **ORDER BY:** specifies the sorting field. A multi-field index can be sorted. If the first field is the same, the second field is used for sorting, and so on.
- **index_granularity = 8192:** specifies the index granularity. One index value is recorded for every 8,192 data records.

If the data to be queried exists in a partition or sorting field, the data query time can be greatly reduced.

- **ReplacingMergeTree**

Different from MergeTree, ReplacingMergeTree deletes duplicate entries with the same sorting key. ReplacingMergeTree is suitable for clearing duplicate data to save space, but it does not guarantee the absence of duplicate data. Generally, it is not recommended.

Syntax for creating a table:

```
CREATE TABLE [IF NOT EXISTS] [db.]table_name [ON CLUSTER cluster]
(
    name1 [type1] [DEFAULT|MATERIALIZED|ALIAS expr1],
    name2 [type2] [DEFAULT|MATERIALIZED|ALIAS expr2],
    ...
) ENGINE = ReplacingMergeTree([ver])
[PARTITION BY expr]
[ORDER BY expr]
[SAMPLE BY expr]
[SETTINGS name=value, ...]
```

- **SummingMergeTree**

When merging data parts in SummingMergeTree tables, ClickHouse merges all rows with the same primary key into one row that contains summed values for the columns with the numeric data type. If the primary key is composed in a way that a single key value corresponds to large number of rows, storage volume can be significantly reduced and the data query speed can be accelerated.

Syntax for creating a table:

```
CREATE TABLE [IF NOT EXISTS] [db.]table_name [ON CLUSTER cluster]
(
    name1 [type1] [DEFAULT|MATERIALIZED|ALIAS expr1],
    name2 [type2] [DEFAULT|MATERIALIZED|ALIAS expr2],
    ...
) ENGINE = SummingMergeTree([columns])
[PARTITION BY expr]
[ORDER BY expr]
[SAMPLE BY expr]
[SETTINGS name=value, ...]
```

Example:

Create a SummingMergeTree table named **testTable**.

```
CREATE TABLE testTable
(
    id UInt32,
    value UInt32
)
ENGINE = SummingMergeTree()
ORDER BY id
```

Insert data into the table.

```
INSERT INTO testTable Values(5,9),(5,3),(4,6),(1,2),(2,5),(1,4),(3,8);
INSERT INTO testTable Values(88,5),(5,5),(3,7),(3,5),(1,6),(2,6),(4,7),(4,6),(43,5),(5,9),(3,6);
```

Query all data in unmerged parts.

```
SELECT * FROM testTable
```

id	value
1	6

2	5
3	8
4	6
5	12

id	value
1	6
2	6
3	18
4	13
5	14
43	5
88	5

If ClickHouse has not summed up all rows and you need to aggregate data by ID, use the **sum** function and **GROUP BY** statement.

```
SELECT id, sum(value) FROM testTable GROUP BY id
```

id	sum(value)
4	19
3	26
88	5
2	11
5	26
1	12
43	5

Merge rows manually.

```
OPTIMIZE TABLE testTable
```

Query data in the **testTable** table again.

```
SELECT * FROM testTable
```

id	value
1	12
2	11
3	26
4	19
5	26
43	5
88	5

SummingMergeTree uses the **ORDER BY** sorting keys as the condition keys to aggregate data. That is, if sorting keys are the same, data records are merged into one and the specified merged fields are aggregated.

Data is pre-aggregated only when merging is executed in the background, and the merging execution time cannot be predicted. Therefore, it is possible that some data has been pre-aggregated and some data has not been aggregated. Therefore, the **GROUP BY** statement must be used during aggregation.

- AggregatingMergeTree

AggregatingMergeTree is a pre-aggregation engine used to improve aggregation performance. When merging partitions, the AggregatingMergeTree engine aggregates data based on predefined conditions, calculates data based on predefined aggregate functions, and saves the data in binary format to tables.

Syntax for creating a table:

```
CREATE TABLE [IF NOT EXISTS] [db.]table_name [ON CLUSTER cluster]
(
  name1 [type1] [DEFAULT|MATERIALIZED|ALIAS expr1],
  name2 [type2] [DEFAULT|MATERIALIZED|ALIAS expr2],
  ...
)
```

```
) ENGINE = AggregatingMergeTree()
[PARTITION BY expr]
[ORDER BY expr]
[SAMPLE BY expr]
[TTL expr]
[SETTINGS name=value, ...]
```

Example:

You do not need to set the `AggregatingMergeTree` parameter separately. When partitions are merged, data in each partition is aggregated based on the **ORDER BY** sorting key. You can set the aggregate functions to be used and column fields to be calculated by defining the `AggregateFunction` type, as shown in the following example:

```
create table test_table (
name1 String,
name2 String,
name3 AggregateFunction(uniq,String),
name4 AggregateFunction(sum,Int),
name5 DateTime
) ENGINE = AggregatingMergeTree()
PARTITION BY toYYYYMM(name5)
ORDER BY (name1,name2)
PRIMARY KEY name1;
```

When data of the `AggregateFunction` type is written or queried, the ***state** and ***merge** functions need to be called. The asterisk (*) indicates the aggregate functions used for defining the field type. For example, the **uniq** and **sum** functions are specified for the **name3** and **name4** fields defined in the **test_table**, respectively. Therefore, you need to call the **uniqState** and **sumState** functions and run the **INSERT** and **SELECT** statements when writing data into the table.

```
insert into test_table select '8','test1',uniqState('name1'),sumState(toInt32(100)),'2021-04-30
17:18:00';
insert into test_table select '8','test1',uniqState('name1'),sumState(toInt32(200)),'2021-04-30
17:18:00';
```

When querying data, you need to call the corresponding functions **uniqMerge** and **sumMerge**.

```
select name1,name2,uniqMerge(name3),sumMerge(name4) from test_table group by name1,name2;
```

name1	name2	uniqMerge(name3)	sumMerge(name4)
8	test1	1	300

`AggregatingMergeTree` is more commonly used with materialized views, which are query views of other data tables at the upper layer. For details, visit <https://clickhouse.tech/docs/en/engines/table-engines/mergetree-family/aggregatingmergetree/>

- CollapsingMergeTree

`CollapsingMergeTree` defines a **Sign** field to record status of data rows. If **Sign** is **1**, the data in this row is valid. If **Sign** is **-1**, the data in this row needs to be deleted.

Syntax for creating a table:

```
CREATE TABLE [IF NOT EXISTS] [db.]table_name [ON CLUSTER cluster]
(
name1 [type1] [DEFAULT|MATERIALIZED]ALIAS expr1],
name2 [type2] [DEFAULT|MATERIALIZED]ALIAS expr2],
...
) ENGINE = CollapsingMergeTree(sign)
[PARTITION BY expr]
[ORDER BY expr]
[SAMPLE BY expr]
[SETTINGS name=value, ...]
```

Example:

For details about the example, visit <https://clickhouse.tech/docs/en/engines/table-engines/mergetree-family/collapsingmergetree/>.

- VersionedCollapsingMergeTree

The VersionedCollapsingMergeTree engine adds **Version** to the table creation statement to record the mapping between a **state** row and a **cancel** row in case that rows are out of order. The rows with the same primary key, same **Version**, and opposite **Sign** will be deleted during compaction.

Syntax for creating a table:

```
CREATE TABLE [IF NOT EXISTS] [db.]table_name [ON CLUSTER cluster]
(
    name1 [type1] [DEFAULT|MATERIALIZED|ALIAS expr1],
    name2 [type2] [DEFAULT|MATERIALIZED|ALIAS expr2],
    ...
) ENGINE = VersionedCollapsingMergeTree(sign, version)
[PARTITION BY expr]
[ORDER BY expr]
[SAMPLE BY expr]
[SETTINGS name=value, ...]
```

Example:

For details about the example, visit <https://clickhouse.tech/docs/en/engines/table-engines/mergetree-family/versionedcollapsingmergetree/>.

- GraphiteMergeTree

The GraphiteMergeTree engine is used to store data in the time series database Graphite.

Syntax for creating a table:

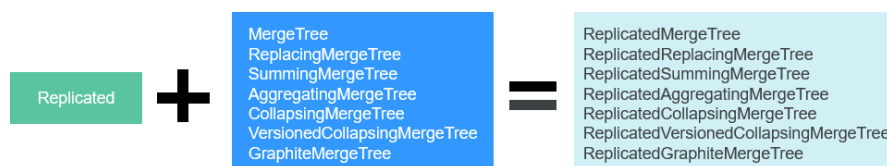
```
CREATE TABLE [IF NOT EXISTS] [db.]table_name [ON CLUSTER cluster]
(
    Path String,
    Time DateTime,
    Value <Numeric_type>,
    Version <Numeric_type>
    ...
) ENGINE = GraphiteMergeTree(config_section)
[PARTITION BY expr]
[ORDER BY expr]
[SAMPLE BY expr]
[SETTINGS name=value, ...]
```

Example:

For details about the example, visit <https://clickhouse.tech/docs/en/engines/table-engines/mergetree-family/graphitemergetree/>.

Replicated*MergeTree Engines

All engines of the MergeTree family in ClickHouse prefixed with Replicated become MergeTree engines that support replicas.



Replicated series engines use ZooKeeper to synchronize data. When a replicated table is created, all replicas of the same shard are synchronized based on the information registered with ZooKeeper.

Template for creating a Replicated engine:

```
ENGINE = ReplicatedMergeTree('Storage path in ZooKeeper', 'Replica name', ...)
```

Two parameters need to be specified for a Replicated engine:

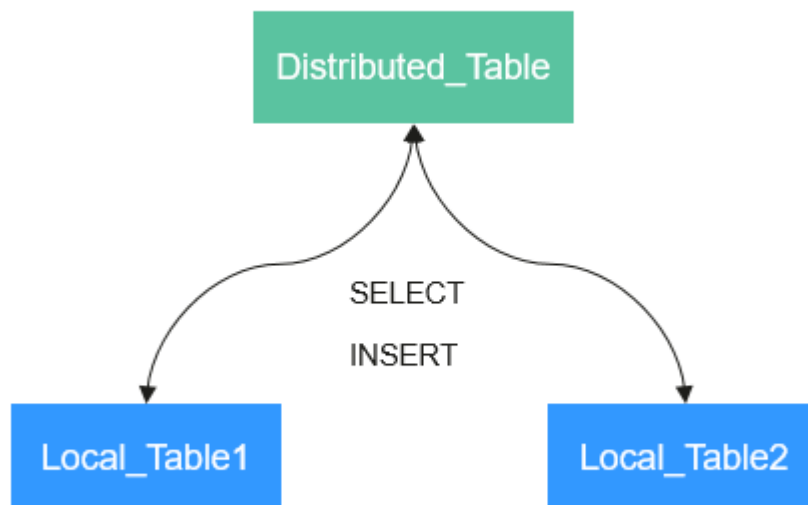
- *Storage path in ZooKeeper*: specifies the path for storing table data in ZooKeeper. The path format is */clickhouse/tables/{shard}/Database name/ Table name*.
- *Replica name*: Generally, **{replica}** is used.

For details about the example, see [Creating and Using ClickHouse Replicated Tables and Distributed Tables](#).

Distributed Engine

The Distributed engine does not store any data. It serves as a transparent proxy for data shards and can automatically transmit data to each node in the cluster. Distributed tables need to work with other local data tables. Distributed tables distribute received read and write tasks to each local table where data is stored.

Figure 4-3 Working principle of the Distributed engine



Template for creating a Distributed engine:

```
ENGINE = Distributed(cluster_name, database_name, table_name, [sharding_key])
```

Parameters of a distributed table are described as follows:

- **cluster_name**: specifies the cluster name. When a distributed table is read or written, the cluster configuration information is used to search for the corresponding ClickHouse instance node.

- **database_name**: specifies the database name.
- **table_name**: specifies the name of a local table in the database. It is used to map a distributed table to a local table.
- **sharding_key** (optional): specifies the sharding key, based on which a distributed table distributes data to each local table.

Example:

```
-- Create a ReplicatedMergeTree local table named test.
CREATE TABLE default.test ON CLUSTER default_cluster_1
(
  `EventDate` DateTime,
  `id` UInt64
)
ENGINE = ReplicatedMergeTree('/clickhouse/tables/{shard}/default/test', '{replica}')
PARTITION BY toYYYYMM(EventDate)
ORDER BY id

-- Create a distributed table named test_all based on the local table test.
CREATE TABLE default.test_all ON CLUSTER default_cluster_1
(
  `EventDate` DateTime,
  `id` UInt64
)
ENGINE = Distributed(default_cluster_1, default, test, rand())
```

Rules for creating a distributed table:

- When creating a distributed table, add **ON CLUSTER** *cluster_name* to the table creation statement so that the statement can be executed once on a ClickHouse instance and then distributed to all instances in the cluster for execution.
- Generally, a distributed table is named in the following format: *Local table name_all*. It forms a one-to-many mapping with local tables. Then, multiple local tables can be operated using the distributed table proxy.
- Ensure that the structure of a distributed table is the same as that of local tables. If they are inconsistent, no error is reported during table creation, but an exception may be reported during data query or insertion.

4.5 Creating and Using ClickHouse Replicated Tables and Distributed Tables

ClickHouse implements the replicated table mechanism based on the ReplicatedMergeTree engine and ZooKeeper. When creating a table, you can specify an engine to determine whether the table is highly available. Shards and replicas of each table are independent of each other.

ClickHouse also implements the distributed table mechanism based on the Distributed engine. Views are created on all shards (local tables) for distributed query, which is easy to use. ClickHouse has the concept of data sharding, which is one of the features of distributed storage. That is, parallel read and write are used to improve efficiency.

The ClickHouse cluster table engine that uses Kunpeng as the CPU architecture does not support HDFS and Kafka.

Viewing cluster and Other Environment Parameters of ClickHouse

Step 1 Use the ClickHouse client to connect to the ClickHouse server by referring to [Using ClickHouse from Scratch](#).

Step 2 Query the cluster identifier and other information about the environment parameters.

```
select cluster,shard_num,replica_num,host_name from system.clusters;
```

```
SELECT  
  cluster,  
  shard_num,  
  replica_num,  
  host_name  
FROM system.clusters
```

cluster	shard_num	replica_num	host_name
default_cluster_1	1	1	node-master1dOnG
default_cluster_1	1	2	node-group-1tXED0001
default_cluster_1	2	1	node-master2OXQS
default_cluster_1	2	2	node-group-1tXED0002
default_cluster_1	3	1	node-master3QsRI
default_cluster_1	3	2	node-group-1tXED0003

6 rows in set. Elapsed: 0.001 sec.

Step 3 Query the shard and replica identifiers.

```
select * from system.macros;
```

```
SELECT *  
FROM system.macros
```

macro	substitution
id	76
replica	node-master3QsRI
shard	3

3 rows in set. Elapsed: 0.001 sec.

----End

Creating a Local Replicated Table and a distributed Table

Step 1 Log in to the ClickHouse node on the client, for example, `clickhouse client --host node-master3QsRI --multiline ;`.

 **NOTE**

`node-master3QsRI` is the value of `host_name` obtained in [Step 2](#) in [Viewing cluster and Other Environment Parameters of ClickHouse](#).

Step 2 Create a replicated table using the ReplicatedMergeTree engine.

For details about the syntax, see <https://clickhouse.tech/docs/en/engines/table-engines/mergetree-family/replication/#creating-replicated-tables>.

For example, run the following commands to create a ReplicatedMergeTree table named `test` on the `default_cluster_1` node and in the `default` database:

```
CREATE TABLE default.test ON CLUSTER default_cluster_1
```

```
(
```

```

`EventDate` DateTime,
`id` UInt64
)
ENGINE = ReplicatedMergeTree('/clickhouse/tables/{shard}/default/test',
'{replica}')
PARTITION BY toYYYYMM(EventDate)
ORDER BY id;

```

The parameters are described as follows:

- The **ON CLUSTER** syntax indicates the distributed DDL, that is, the same local table can be created on all instances in the cluster after the statement is executed once.
- **default_cluster_1** is the cluster identifier obtained in [Step 2 in Viewing cluster and Other Environment Parameters of ClickHouse](#).

 **CAUTION**

ReplicatedMergeTree engine receives the following two parameters:

- Storage path of the table data in ZooKeeper

The path must be in the **/clickhouse** directory. Otherwise, data insertion may fail due to insufficient ZooKeeper quota.

To avoid data conflict between different tables in ZooKeeper, the directory must be in the following format:

/clickhouse/tables/{shard} default/test, in which **/clickhouse/tables/{shard}** is fixed, *default* indicates the database name, and *test* indicates the name of the created table.

- Replica name: Generally, **{replica}** is used.

```

CREATE TABLE default.test ON CLUSTER default_cluster_1
(
  `EventDate` DateTime,
  `id` UInt64
)
ENGINE = ReplicatedMergeTree('/clickhouse/tables/{shard}/default/test', '{replica}')
PARTITION BY toYYYYMM(EventDate)
ORDER BY id

```

host	port	status	error	num_hosts_remaining
num_hosts_active				
node-group-1tXED0002	9000	0		5
node-group-1tXED0003	9000	0		4
node-master1dOnG	9000	0		3
num_hosts_active				
node-master3QsRI	9000	0		2
node-group-1tXED0001	9000	0		1
node-master2OXQS	9000	0		0

6 rows in set. Elapsed: 0.189 sec.

Step 3 Create a distributed table using the Distributed engine.

For example, run the following commands to create a distributed table named **test_all** on the **default_cluster_1** node and in the **default** database:

```
CREATE TABLE default.test_all ON CLUSTER default_cluster_1
(
  `EventDate` DateTime,
  `id` UInt64
)
ENGINE = Distributed(default_cluster_1, default, test, rand());
```

```
CREATE TABLE default.test_all ON CLUSTER default_cluster_1
(
  `EventDate` DateTime,
  `id` UInt64
)
ENGINE = Distributed(default_cluster_1, default, test, rand())
```

host	port	status	error	num_hosts_remaining
node-group-1tXED0002	9000	0		5
node-master3QsRI	9000	0		4
node-group-1tXED0003	9000	0		3
node-group-1tXED0001	9000	0		2
node-master1dOnG	9000	0		1
node-master2OXQS	9000	0		0

6 rows in set. Elapsed: 0.115 sec.

NOTE

Distributed requires the following parameters:

- **default_cluster_1** is the cluster identifier obtained in [Step 2 in Viewing cluster and Other Environment Parameters of ClickHouse](#).
- **default** indicates the name of the database where the local table is located.
- **test** indicates the name of the local table. In this example, it is the name of the table created in [Step 2](#).
- (Optional) Sharding key

This key and the weight configured in the **config.xml** file determine the route for writing data to the distributed table, that is, the physical table to which the data is written. It can be the original data (for example, **site_id**) of a column in the table or the result of the function call, for example, **rand()** is used in the preceding SQL statement. Note that data must be evenly distributed in this key. Another common operation is to use the hash value of a column with a large difference, for example, **intHash64(user_id)**.

----End

ClickHouse Table Data Operations

Step 1 Log in to the ClickHouse node on the client. For example,

```
clickhouse client --host node-master3QsRI --multiline ;
```

 NOTE

node-master3QsRI is the value of **host_name** obtained in [Step 2](#) in [Viewing cluster and Other Environment Parameters of ClickHouse](#).

Step 2 After creating a table by referring to [Creating a Local Replicated Table and a distributed Table](#), you can insert data to the local table.

For example, run the following command to insert data to the local table **test**:

insert into test values(toDateTime(now()), rand());

Step 3 Query the local table information.

For example, run the following command to query data information of the table **test** in [Step 2](#):

select * from test;

```
SELECT *  
FROM test
```

EventDate	id
2020-11-05 21:10:42	1596238076

1 rows in set. Elapsed: 0.002 sec.

Step 4 Query the distributed table.

For example, the distributed table **test_all** is created based on table **test** in [Step 3](#). Therefore, the same data in table **test** can also be queried in table **test_all**.

select * from test_all;

```
SELECT *  
FROM test_all
```

EventDate	id
2020-11-05 21:10:42	1596238076

1 rows in set. Elapsed: 0.004 sec.

Step 5 Switch to the shard node with the same **shard_num** and query the information about the current table. The same table data can be queried.

For example, run the **exit;** command to exit the original node.

Run the following command to switch to the **node-group-1tXED0003** node:

clickhouse client --host node-group-1tXED0003 --multiline ;

 NOTE

The **shard_num** values of **node-group-1tXED0003** and **node-master3QsRI** are the same by performing [Step 2](#).

show tables;

```
SHOW TABLES
```

name
test

```
test_all
```

Step 6 Query the local table data. For example, run the following command to query data in table **test** on the **node-group-1tXED0003** node:

```
select * from test;  
SELECT *  
FROM test
```

EventDate	id
2020-11-05 21:10:42	1596238076

```
1 rows in set. Elapsed: 0.005 sec.
```

Step 7 Switch to the shard node with different **shard_num** value and query the data of the created table.

For example, run the following command to exit the **node-group-1tXED0003** node:

```
exit;
```

Switch to the **node-group-1tXED0001** node. The **shard_num** values of **node-group-1tXED0001** and **node-master3QsRI** are different by performing [Step 2](#).

```
clickhouse client --host node-group-1tXED0001 --multiline ;
```

Query the local table **test**. Data cannot be queried on the different shard node because table **test** is a local table.

```
select * from test;  
SELECT *  
FROM test  
Ok.
```

Query data in the distributed table **test_all**. The data can be queried properly.

```
select * from test_all;  
SELECT *  
FROM test
```

EventDate	id
2020-11-05 21:12:19	3686805070

```
1 rows in set. Elapsed: 0.002 sec.
```

```
----End
```

4.6 ClickHouse Log Overview

Log Description

Log path: The default storage path of ClickHouse log files is as follows: **\$ {BIGDATA_LOG_HOME}/clickhouse**

Log archive rule: The automatic ClickHouse log compression function is enabled. By default, when the size of logs exceeds 100 MB, logs are automatically compressed into a log file named in the following format: *<Original log name>.[ID].gz*. A maximum of 10 latest compressed files are reserved by default. The number of compressed files can be configured on Manager.

Table 4-2 ClickHouse log list

Log Type	Log File Name	Description
Run logs	/clickhouse/clickhouse-server.err.log	Path of the ClickHouse running error log files.
	/clickhouse/checkService.log	Path of the ClickHouse key run log files.

Log Level

Table 4-3 describes the log levels supported by ClickHouse.

Levels of run logs are error, warning, trace, information, and debug from the highest to the lowest priority. Run logs of equal or higher levels are recorded. The higher the specified log level, the fewer the logs recorded.

Table 4-3 Log levels

Log Type	Level	Description
Run log	error	Logs of this level record error information about system running.
	warning	Logs of this level record exception information about the current event processing.
	trace	Logs of this level record trace information about the current event processing.
	information	Logs of this level record normal running status information about the system and events.
	debug	Logs of this level record system running and debugging information.

To modify log levels, perform the following operations:

- Step 1** Log in to FusionInsight Manager.
- Step 2** Choose **Cluster > Services > ClickHouse > Configurations**.
- Step 3** Select **All Configurations**.
- Step 4** On the menu bar on the left, select the log menu of the target role.
- Step 5** Select a desired log level.
- Step 6** Click **Save**. Then, click **OK**.

----End

 **NOTE**

The configurations take effect immediately without the need to restart the service.

Log Format

The following table lists the ClickHouse log format:

Table 4-4 Log formats

Log Type	Format	Example
Run log	<i><yyyy-MM-dd HH:mm:ss,SSS> <Log level> <Name of the thread that generates the log> <Message in the log> <Location where the log event occurs></i>	2021.02.23 15:26:30.691301 [6085] { } <Error> DynamicQueryHandler: Code: 516, e.displayText() = DB::Exception: default: Authentication failed: password is incorrect or there is no user with such name, Stack trace (when copying this message, always include the lines below): 0. Poco::Exception::Exceptio n(std::_1::basic_string<c har, std::_1::char_traits<char >, std::_1::allocator<char> > const&, int) @ 0x1250e59c

5 Using DBService

5.1 DBService Log Overview

Log Description

Log path: The default storage path of DBService log files is `/var/log/Bigdata/dbservice`.

- GaussDB: `/var/log/Bigdata/dbservice/DB` (GaussDB run log directory), `/var/log/Bigdata/dbservice/scriptlog/gaussdbinstall.log` (GaussDB installation log), and `/var/log/gaussdbuninstall.log` (GaussDB uninstallation log).
- HA: `/var/log/Bigdata/dbservice/ha/runlog` (HA run log directory) and `/var/log/Bigdata/dbservice/ha/scriptlog` (HA script log directory)
- DBServer: `/var/log/Bigdata/dbservice/healthCheck` (Directory of service and process health check logs)
`/var/log/Bigdata/dbservice/scriptlog` (run log directory), `/var/log/Bigdata/audit/dbservice/` (audit log directory)

Log archive rule: The automatic DBService log compression function is enabled. By default, when the size of logs exceeds 1 MB, logs are automatically compressed into a log file named in the following format: `<Original log file name>-[No.].gz`. A maximum of 20 latest compressed files are reserved.

NOTE

Log archive rules cannot be modified.

Table 5-1 DBService log list

Type	Log File Name	Description
DBServer run log	dbservice_serviceCheck.log	Run log file of the service check script

Type	Log File Name	Description
	dbservice_processCheck.log	Run log file of the process check script
	backup.log	Run logs of backup and restoration operations (The DBService backup and restoration operations need to be performed.)
	checkHaStatus.log	Log file of HA check records
	cleanupDBService.log	Uninstallation log file (You need to uninstall DBService logs.)
	componentUserManager.log	Log file that records the adding and deleting operations on the database by users (Services that depend on DBService need to be added.)
	install.log	Installation log file
	preStartDBService.log	Pre-startup log file
	start_dbserver.log	DBServer startup operation log file (DBService needs to be started.)
	stop_dbserver.log	DBServer stop operation log file (DBService needs to be stopped.)
	status_dbserver.log	Log file of the DBServer status check (You need to execute the \$DBSERVICE_HOME/sbin/status-dbserver.sh script.)
	modifyPassword.log	Run log file of changing the DBService password script. (You need to execute the \$DBSERVICE_HOME/sbin/modifyDBPwd.sh script.)

Type	Log File Name	Description
	modifyDBPwd_YYYY-MM-DD.log	Run log file that records the DBService password change tool (You need to execute the \$DBSERVICE_HOME/sbin/modifyDBPwd.sh script.)
	dbserver_switchover.log	Log for DBServer to execute the active/standby switchover script (the active/standby switchover needs to be performed)
GaussDB run log	gaussdb.log	Log file that records database running information
	gs_ctl-current.log	Log file that records operations performed by using the gs_ctl tool
	gs_guc-current.log	Log file that records operations, mainly parameter modification performed by using the gs_guc tool
	gaussdbinstall.log	GaussDB installation log file
	gaussdbuninstall.log	GaussDB uninstallation log file
HA script run log	floatip_ha.log	Log file that records the script of floating IP addresses
	gaussDB_ha.log	Log file that records the script of GaussDB resources
	ha_monitor.log	Log file that records the HA process monitoring information
	send_alarm.log	Alarm sending log file
	ha.log	HA run log file

Type	Log File Name	Description
DBService audit log	dbservice_audit.log	Audit log file that records DBService operations, such as backup and restoration operations

Log Format

The following table lists the DBService log formats.

Table 5-2 Log format

Type	Format	Example
Run log	[<yyyy-MM-dd HH:mm:ss> <Log level>: [< Name of the script that generates the log. Line number >]: < Message in the log>	[2020-12-19 15:56:42] INFO [postinstall.sh:653] Is cloud flag is false. (main)
Audit log	[<yyyy-MM-dd HH:mm:ss,SSS> UserName:<Username> UserIP:<User IP address> Operation:<Operation content> Result:<Operation results> Detail:<Detailed information>	[2020-05-26 22:00:23] UserName:omm UserIP: 192.168.10.21 Operation:DBService data backup Result: SUCCESS Detail: DBService data backup is successful.

6 Using Flink

6.1 Using Flink from Scratch

This section describes how to use Flink to run wordcount jobs.

Prerequisites

- Flink has been installed in an MRS cluster.
- The cluster runs properly and the client has been correctly installed, for example, in the **/opt/hadoopclient** directory. The client directory in the following operations is only an example. Change it to the actual installation directory.

Using the Flink Client (Versions Earlier Than MRS 3.x)

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

Step 3 Run the following command to initialize environment variables:

```
source /opt/hadoopclient/bigdata_env
```

Step 4 If Kerberos authentication is enabled for the cluster, perform the following steps. If not, skip this whole step.

1. Prepare a user for submitting Flink jobs..
2. Log in to Manager and download the authentication credential.
Log in to Manager of the cluster. For details, see [Accessing MRS Manager \(Versions Earlier Than MRS 3.x\)](#). Choose **System Settings > User Management**. In the **Operation** column of the row that contains the added user, choose **More > Download Authentication Credential**.
3. Decompress the downloaded authentication credential package and copy the **user.keytab** file to the client node, for example, to the **/opt/hadoopclient/Flink/flink/conf** directory on the client node. If the client is installed on a

node outside the cluster, copy the **krb5.conf** file to the **/etc/** directory on this node.

4. Configure security authentication by adding the **keytab** path and username in the **/opt/hadoopclient/Flink/flink/conf/flink-conf.yaml** configuration file.

security.kerberos.login.keytab: *<user.keytab file path>*

security.kerberos.login.principal: *<Username>*

Example:

security.kerberos.login.keytab: */opt/hadoopclient/Flink/flink/conf/user.keytab*

security.kerberos.login.principal: *test*

5. In the **bin** directory of the Flink client, run the following command to perform security hardening and set password to a new password for submitting jobs. For details, see [Authentication and Encryption](#).

sh generate_keystore.sh <password>

The script automatically replaces the SSL value in the **/opt/hadoopclient/Flink/flink/conf/flink-conf.yaml** file. For MRS 2.x or earlier, external SSL is disabled by default in a security cluster. To enable external SSL, configure related parameters and run the script again. For details, see [Security Hardening](#).

 **NOTE**

- You do not need to manually generate the **generate_keystore.sh** script.
 - **flink.keystore**, **flink.truststore**, and **flink.truststore** will be automatically filled in the corresponding configuration items in the **flink-conf.yaml** file after authentication and encryption. For details, see [Authentication and Encryption](#).
6. Configure paths for the client to access the **flink.keystore** and **flink.truststore** files.
 - Absolute path: After the script is executed, the file path of **flink.keystore** and **flink.truststore** is automatically set to the absolute path **/opt/hadoopclient/Flink/flink/conf/** in the **flink-conf.yaml** file. In this case, you need to move the **flink.keystore** and **flink.truststore** files from the **conf** directory to this absolute path on the Flink client and Yarn nodes.
 - Relative path: Perform the following steps to set the file path of **flink.keystore** and **flink.truststore** to the relative path and ensure that the directory where the Flink client command is executed can directly access the relative paths.
 - i. Create a directory, for example, **ssl**, in **/opt/hadoopclient/Flink/flink/conf/**.

```
cd /opt/hadoopclient/Flink/flink/conf/  
mkdir ssl
```
 - ii. Move the **flink.keystore** and **flink.truststore** files to the **/opt/hadoopclient/Flink/flink/conf/ssl/** directory.

```
mv flink.keystore ssl/  
mv flink.truststore ssl/
```
 - iii. Change the values of the following parameters to relative paths in the **flink-conf.yaml** file:

```
security.ssl.internal.keystore: ssl/flink.keystore  
security.ssl.internal.truststore: ssl/flink.truststore
```

Step 5 Run a wordcount job.

NOTICE

To submit or run jobs on Flink, the user must have the following permissions:

- If Ranger authentication is enabled, the current user must belong to the **hadoop** group or the user has been granted the **/flink** read and write permissions in Ranger.
- If Ranger authentication is disabled, the current user must belong to the **hadoop** group.

-
- Normal cluster (Kerberos authentication disabled)
 - Run the following commands to start a session and submit a job in the session:
yarn-session.sh -nm "session-name"
flink run /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
 - Run the following command to submit a single job on Yarn:
flink run -m yarn-cluster /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
 - Security cluster (Kerberos authentication enabled)
 - If the **flink.keystore** and **flink.truststore** file are stored in the absolute path:
 - Run the following commands to start a session and submit a job in the session:
yarn-session.sh -nm "session-name"
flink run /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
 - Run the following command to submit a single job on Yarn:
flink run -m yarn-cluster /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
 - If the **flink.keystore** and **flink.truststore** files are stored in the relative path:
 - In the same directory of SSL, run the following commands to start a session and submit jobs in the session. The SSL directory is a relative path. For example, if the SSL directory is **opt/hadoopclient/Flink/flink/conf/**, then run the following commands in this directory:
yarn-session.sh -t ssl/ -nm "session-name"
flink run /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
 - Run the following command to submit a single job on Yarn:
flink run -m yarn-cluster -yt ssl/ /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar

Step 6 After the job has been successfully submitted, the following information is displayed on the client:

Figure 6-1 Job submitted successfully on Yarn

```
[root@node-master1ks2p ~]# flink run -a yarn -c /opt/client/flink/flink/examples/streaming/WordCount.jar
2019-07-10 16:30:11,090 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory (DomainSocketFactory.java:118)
2019-07-10 16:30:11,098 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory (DomainSocketFactory.java:118)
Starting execution of program
Executing WordCount example with default input data set.
Use --input to specify file input.
Printing result to stdout. Use --output to specify output path.
Program execution finished.
Job with JobID c043b1921e89afe2bba24b51a5beid has finished.
Job Runtime: 7953 ms
```

Figure 6-2 Session started successfully

```
[root@node-master1ks2p ~]# yarn-session.sh -m "test4doc" -d
2019-07-26 09:17:08,919 | WARN | [main] | Unable to load native-hadoop library for your platform... using builtin-java classes where applicable | org.apache.hadoop.util.NativeCodeLoader (NativeCodeLoader.java:62)
2019-07-26 09:17:08,989 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory (DomainSocketFactory.java:118)
Flink JobManager is now running on node-ema-corehdp-32506 with leader id b9b5ab0-1983-435f-bb00-ad12fd1d46b.
JobManager Web Interface: http://192.168.2.61:47897
[root@node-master1ks2p ~]#
```

Figure 6-3 Job submitted successfully in the session

```
[root@node-master1ks2p ~]# flink run /opt/client/flink/flink/examples/streaming/WordCount.jar
YARN properties set default parallelism to 3
2019-07-26 09:19:20,548 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory (DomainSocketFactory.java:118)
2019-07-26 09:19:20,588 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory (DomainSocketFactory.java:118)
Starting execution of program
Executing WordCount example with default input data set.
Use --input to specify file input.
Printing result to stdout. Use --output to specify output path.
Program execution finished.
Job with JobID 9ab6c186563fd792a19163c2e7c3c3 has finished.
Job Runtime: 5906 ms
[root@node-master1ks2p ~]#
```

Step 7 Go to the native Yarn service page, find the application of the job, and click the application name to go to the job details page. For details, see [Viewing Flink Job Information](#).

- If the job is not completed, click **Tracking URL** to go to the native Flink page and view the job running information.
- If the job submitted in a session has been completed, you can click **Tracking URL** to log in to the native Flink service page to view job information.

----End

Using the Flink Client (MRS 3.x or Later)

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

Step 3 Run the following command to initialize environment variables:

```
source /opt/hadoopclient/bigdata_env
```

Step 4 If Kerberos authentication is enabled for the cluster, perform the following steps. If not, skip this whole step.

1. Prepare a user for submitting Flink jobs.
2. Log in to Manager and download the authentication credential.
Log in to Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **System > Permission > Manage User**. On the displayed page, locate the row that contains the added user, click **More** in the **Operation** column, and select **Download authentication credential**.
3. Decompress the downloaded authentication credential package and copy the **user.keytab** file to the client node, for example, to the **/opt/hadoopclient/**

Flink/flink/conf directory on the client node. If the client is installed on a node outside the cluster, copy the **krb5.conf** file to the **/etc/** directory on this node.

4. Append the service IP address of the node where the client is installed, floating IP address of Manager, and IP address of the master node to the **jobmanager.web.access-control-allow-origin** and **jobmanager.web.allow-access-address** configuration item in the **/opt/hadoopclient/Flink/flink/conf/flink-conf.yaml** file. Use commas (,) to separate IP addresses.

```
jobmanager.web.access-control-allow-origin: xx.xx.xxx.xxx,xx.xx.xxx.xxx,xx.xx.xxx.xxx
jobmanager.web.allow-access-address: xx.xx.xxx.xxx,xx.xx.xxx.xxx,xx.xx.xxx.xxx
```

NOTE

- To obtain the service IP address of the node where the client is installed, perform the following operations:
 - Node inside the cluster:

In the navigation tree of the MRS management console, choose **Clusters > Active Clusters**, select a cluster, and click its name to switch to the cluster details page.

On the **Nodes** tab page, view the IP address of the node where the client is installed.
 - Node outside the cluster: IP address of the ECS where the client is installed.
 - To obtain the floating IP address of Manager, perform the following operations:
 - In the navigation tree of the MRS management console, choose **Clusters > Active Clusters**, select a cluster, and click its name to switch to the cluster details page.
 - On the **Nodes** tab page, view the **Name**. The node that contains **master1** in its name is the Master1 node. The node that contains **master2** in its name is the Master2 node.
 - Log in to the Master2 node remotely, and run the **ifconfig** command. In the command output, **eth0:wsom** indicates the floating IP address of MRS Manager. Record the value of **inet**. If the floating IP address of MRS Manager cannot be queried on the Master2 node, switch to the Master1 node to query and record the floating IP address. If there is only one Master node, query and record the cluster manager IP address of the Master node.
5. Configure security authentication by adding the **keytab** path and username in the **/opt/hadoopclient/Flink/flink/conf/flink-conf.yaml** configuration file.

security.kerberos.login.keytab: *<user.keytab file path>*

security.kerberos.login.principal: *<Username>*

Example:

security.kerberos.login.keytab: /opt/hadoopclient/Flink/flink/conf/user.keytab

security.kerberos.login.principal: test

6. In the **bin** directory of the Flink client, run the following command to perform security hardening and set password to a new password for submitting jobs. For details, see [Authentication and Encryption](#).

sh generate_keystore.sh <password>

The script automatically replaces the SSL value in the **/opt/hadoopclient/Flink/flink/conf/flink-conf.yaml** file.

 NOTE

After operations in [Authentication and Encryption](#) are performed, the **flink.keystore** and **flink.truststore** files are generated in the **conf** directory of the Flink client and the following configuration items in the **flink-conf.yaml** client configuration file are set by default:

- Set **security.ssl.keystore** to the absolute path of the **flink.keystore** file.
- Set **security.ssl.truststore** to the absolute path of the **flink.truststore** file.
- Set **security.cookie** to a random password automatically generated by the **generate_keystore.sh** script.
- By default, **security.ssl.encrypt.enabled** is set to **false** in the **flink-conf.yaml** file by default. The **generate_keystore.sh** script sets **security.ssl.key-password**, **security.ssl.keystore-password**, and **security.ssl.truststore-password** to the password entered when the **generate_keystore.sh** script is called.

7. Configure paths for the client to access the **flink.keystore** and **flink.truststore** files.

- Absolute path: After the script is executed, the file path of **flink.keystore** and **flink.truststore** is automatically set to the absolute path **/opt/hadoopclient/Flink/flink/conf/** in the **flink-conf.yaml** file. In this case, you need to move the **flink.keystore** and **flink.truststore** files from the **conf** directory to this absolute path on the Flink client and Yarn nodes.
- Relative path: Perform the following steps to set the file path of **flink.keystore** and **flink.truststore** to the relative path and ensure that the directory where the Flink client command is executed can directly access the relative paths.

- i. Create a directory, for example, **ssl**, in **/opt/hadoopclient/Flink/flink/conf/**.

```
cd /opt/hadoopclient/Flink/flink/conf/  
mkdir ssl
```

- ii. Move the **flink.keystore** and **flink.truststore** files to the **/opt/hadoopclient/Flink/flink/conf/ssl/** directory.

```
mv flink.keystore ssl/  
mv flink.truststore ssl/
```

- iii. Change the values of the following parameters to relative paths in the **flink-conf.yaml** file:

```
security.ssl.keystore: ssl/flink.keystore  
security.ssl.truststore: ssl/flink.truststore
```

Step 5 Run a wordcount job.

NOTICE

To submit or run jobs on Flink, the user must have the following permissions:

- If Ranger authentication is enabled, the current user must belong to the **hadoop** group or the user has been granted the **/flink** read and write permissions in Ranger.
 - If Ranger authentication is disabled, the current user must belong to the **hadoop** group.
-

- Normal cluster (Kerberos authentication disabled)
 - Run the following commands to start a session and submit a job in the session:


```
yarn-session.sh -nm "session-name"
flink run /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
```
 - Run the following command to submit a single job on Yarn:


```
flink run -m yarn-cluster /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
```
- Security cluster (Kerberos authentication enabled)
 - If the **flink.keystore** and **flink.truststore** files are stored in the absolute path:
 - Run the following commands to start a session and submit a job in the session:


```
yarn-session.sh -nm "session-name"
flink run /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
```
 - Run the following command to submit a single job on Yarn:


```
flink run -m yarn-cluster /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
```
 - If the **flink.keystore** and **flink.truststore** file are stored in the relative path:
 - In the same directory of SSL, run the following commands to start a session and submit jobs in the session. The SSL directory is a relative path. For example, if the SSL directory is **opt/hadoopclient/Flink/flink/conf/**, then run the following commands in this directory:


```
yarn-session.sh -t ssl/ -nm "session-name"
flink run /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
```
 - Run the following command to submit a single job on Yarn:


```
flink run -m yarn-cluster -yt ssl/ /opt/hadoopclient/Flink/flink/examples/streaming/WordCount.jar
```

Step 6 After the job has been successfully submitted, the following information is displayed on the client:

Figure 6-4 Job submitted successfully on Yarn

```
[root@node-master1kz2P ~]# flink run -m yarn-cluster /opt/client/Flink/flink/examples/streaming/WordCount.jar
2019-07-10 16:30:11,090 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory (DomainSocketFactory.java:118)
2019-07-10 16:30:11,090 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory (DomainSocketFactory.java:118)
Starting execution of program.
Executing WordCount example with default input data set.
Use --input to specify file input.
Printing result to stdout. Use --output to specify output path.
Program execution finished
Job with JobID c043b192e8eafe2bbaz4b51a8be1d has finished.
Job Runtime: 7253 ms
```

Figure 6-5 Session started successfully

```
[root@node-master1kz2P Hive]# yarn-session.sh -nm "test4doc" -d
2019-07-26 09:17:08,619 | WARN | [main] | Unable to load native-hadoop library for your platform... using builtin-java classes where applicable | org.apache.hadoop.util.NativeCodeLoader (NativeCodeLoader.java:92)
2019-07-26 09:17:08,886 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory (DomainSocketFactory.java:118)
Flink JobManager is now running on node-ana-corehdxp:32586 with leader id b9b5ab0-1903-435f-bb90-ad28fd1d46b.
JobManager Web Interface: http://192.168.2.61:47897
[root@node-master1kz2P Hive]#
```

Figure 6-6 Job submitted successfully in the session

```
[root@node-master1kzP Hive]# flink run /opt/client/flink/flink/examples/streaming/wordcount.jar
[WARN] properties set default parallelism to 2
2019-07-26 09:19:20,548 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory
(DomainSocketFactory.java:118)
2019-07-26 09:19:20,548 | WARN | [main] | The short-circuit local reads feature cannot be used because libhadoop cannot be loaded. | org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory
(DomainSocketFactory.java:118)
Starting execution of program
Executing WordCount example with default input data set.
Use --input to specify file input.
Printing result to stdout. Use --output to specify output path.
Program execution finished
Job with JobID 5bdbc18d6583f3d792a19163c2e7c3c3 has finished.
Job Runtime: 5908 ms
[root@node-master1kzP Hive]#
```

Step 7 Go to the native Yarn service page, find the application of the job, and click the application name to go to the job details page. For details, see [Viewing Flink Job Information](#).

- If the job is not completed, click **Tracking URL** to go to the native Flink page and view the job running information.
- If the job submitted in a session has been completed, you can click **Tracking URL** to log in to the native Flink service page to view job information.

----End

6.2 Viewing Flink Job Information

You can view Flink job information on the Yarn web UI.

Prerequisites

The Flink service has been installed in a cluster.

Accessing the Flink Web UI

Step 1 Go to the Yarn service page.

- For versions earlier than MRS 3.x, click the cluster name to go to the cluster details page and choose **Components > Yarn > Yarn Summary**.

NOTE

- If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)
- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster > Name of the desired cluster > Services > Yarn > Instance > Dashboard**.

Step 2 Click the link next to **ResourceManager WebUI** to go to the Yarn web UI page.

----End

6.3 Flink Configuration Management

6.3.1 Configuring Parameter Paths

All parameters of Flink must be set on a client. The path of a configuration file is as follows: **Client installation path/Flink/flink/conf/flink-conf.yaml**.

 NOTE

- You are advised to modify the **flink-conf.yaml** configuration file on the client. The configuration format of the YAML file is *key: value*.
Example: **taskmanager.heap.size: 1024mb**
Note that a space is required between *key:* and *value*.
- If parameters are modified in the Flink service configuration, you need to download and install the client again after the configuration is complete.

6.3.2 JobManager & TaskManager

Scenarios

JobManager and TaskManager are main components of Flink. You can configure the parameters for different security and performance scenarios on the client.

Configuration Description

Main configuration items include communication port, memory management, connection retry, and so on.

For versions earlier than MRS 3.x, see [Table 6-1](#).

Table 6-1 Parameters

Parameter	Mandatory	Default Value	Description
taskmanager.rpc.port	No	32326-32390	IPC port range of TaskManager
taskmanager.data.port	No	32391-32455	Data exchange port range of TaskManager
taskmanager.data.ssl.enabled	No	false	Whether to enable secure sockets layer (SSL) encryption for data transfer between TaskManagers. This parameter is valid only when the global switch security.ssl is enabled.
taskmanager.numberOfTaskSlots	No	3	Number of slots occupied by TaskManager. Generally, the value is configured as the number of cores of the physical machine. In yarn-session mode, the value can be transmitted by only the -s parameter. In yarn-cluster mode, the value can be transmitted by only the -ys parameter.
parallelism.default	No	1	Number of concurrent job operators.

Parameter	Mandatory	Default Value	Description
taskmanager.memory.size	No	0	Amount of heap memory of the Java virtual machine (JVM) that TaskManager reserves for sorting, hash tables, and caching of intermediate results. If unspecified, the memory manager will take a fixed ratio with respect to the size of JVM as specified by taskmanager.memory.fraction . The unit is MB.
taskmanager.memory.fraction	No	0.7	Ratio of JVM heap memory that TaskManager reserves for sorting, hash tables, and caching of intermediate results.
taskmanager.memory.off-heap	Yes	false	Whether TaskManager uses off-heap memory for sorting, hash tables and intermediate status. You are advised to enable this item for large memory needs to improve memory operation efficiency.
taskmanager.memory.segment-size	No	32768	Size of memory segment on TaskManager. Memory segment is the basic unit of the reserved memory space and is used to configure network buffer stacks. The unit is bytes.
taskmanager.memory.preallocate	No	false	Whether TaskManager allocates reserved memory space upon startup. You are advised to enable this item when off-heap memory is used.
taskmanager.registration.initial-backoff	No	500 ms	Initial interval between two consecutive registration attempts. The unit is ms/s/m/h/d. NOTE The time value and unit are separated by half-width spaces. ms/s/m/h/d indicates millisecond, second, minute, hour, and day, respectively.
taskmanager.registration.refused-backoff	No	5 min	Retry interval when a registration connection is rejected by JobManager.
task.cancellation.interval	No	30000	Interval between two successive task cancellation attempts.

For configuration items for MRS 3.x or later, see [Table 6-2](#).

Table 6-2 Parameters

Parameter	Description	Default Value	Mandatory
taskmanager.rpc.port	IPC port range of TaskManager	32326-32390	No
client.rpc.port	Akka system listening port on the Flink client.	32651-32720	No
taskmanager.data.port	Data exchange port range of TaskManager	32391-32455	No
taskmanager.data.ssl.enabled	Whether to enable secure sockets layer (SSL) encryption for data transfer between TaskManagers. This parameter is valid only when the global switch security.ssl is enabled.	false	No

Parameter	Description	Default Value	Mandatory
jobmanager.heap.size	<p>Size of the heap memory of JobManager. In yarn-session mode, the value can be transmitted by only the -jm parameter. In yarn-cluster mode, the value can be transmitted by only the -yjm parameter. If the value is smaller than yarn.scheduler.minimum-allocation-mb in the Yarn configuration file, the Yarn configuration value is used.</p> <p>Unit: B/KB/MB/GB/TB.</p>	1024mb	No

Parameter	Description	Default Value	Mandatory
taskmanager.heap.size	Size of the heap memory of TaskManager. In yarn-session mode, the value can be transmitted by only the -tm parameter. In yarn-cluster mode, the value can be transmitted by only the -ytm parameter. If the value is smaller than yarn.scheduler.minimum-allocation-mb in the Yarn configuration file, the Yarn configuration value is used. The unit is B/KB/MB/GB/TB.	1024mb	No
taskmanager.numberOfTaskSlots	Number of slots occupied by TaskManager. Generally, the value is configured as the number of cores of the physical machine. In yarn-session mode, the value can be transmitted by only the -s parameter. In yarn-cluster mode, the value can be transmitted by only the -ys parameter.	1	No

Parameter	Description	Default Value	Mandatory
parallelism.default	Default degree of parallelism, which is used for jobs for which the degree of parallelism is not specified	1	No
taskmanager.network.numberOfBuffers	Number of TaskManager network transmission buffer stacks. If an error indicates insufficient system buffer, increase the parameter value.	2048	No
taskmanager.memory.fraction	Ratio of JVM heap memory that TaskManager reserves for sorting, hash tables, and caching of intermediate results.	0.7	No
taskmanager.memory.off-heap	Whether TaskManager uses off-heap memory for sorting, hash tables and intermediate status. You are advised to enable this item for large memory needs to improve memory operation efficiency.	false	Yes
taskmanager.memory.segment-size	Size of the memory buffer used by the memory manager and network stack The unit is bytes.	32768	No

Parameter	Description	Default Value	Mandatory
taskmanager.memory.preallocate	Whether TaskManager allocates reserved memory space upon startup. You are advised to enable this item when off-heap memory is used.	false	No
taskmanager.debug.memory.startLogThread	Enable this item for debugging Flink memory and garbage collection (GC)-related problems. TaskManager periodically collects memory and GC statistics, including the current utilization of heap and off-heap memory pools and GC time.	false	No
taskmanager.debug.memory.logIntervalMs	Interval at which TaskManager periodically collects memory and GC statistics.	0	No
taskmanager.maxRegistrationDuration	Maximum duration of TaskManager registration on JobManager. If the actual duration exceeds the value, TaskManager is disabled.	5 min	No

Parameter	Description	Default Value	Mandatory
taskmanager.initial-registration-pause	Initial interval between two consecutive registration attempts. The value must contain a time unit (ms/s/min/h/d), for example, 5 seconds.	500ms NOTE The time value and unit are separated by half-width spaces. ms/s/m/h/d indicates millisecond, second, minute, hour, and day, respectively.	No
taskmanager.max-registration-pause	Maximum registration retry interval in case of TaskManager registration failures. The unit is ms/s/m/h/d.	30s	No
taskmanager.refused-registration-pause	Retry interval when a TaskManager registration connection is rejected by JobManager. The unit is ms/s/m/h/d.	10s	No
task.cancellation.interval	Interval between two successive task cancellation attempts. The unit is millisecond.	30000	No

Parameter	Description	Default Value	Mandatory
classloader.resolve-order	Class resolution policies defined when classes are loaded from user codes, which means whether to first check the user code JAR file (child-first) or the application class path (parent-first). The default setting indicates that the class is first loaded from the user code JAR file, which means that the user code JAR file can contain and load dependencies that are different from those used by Flink.	child-first	No
slot.idle.timeout	Timeout for an idle slot in Slot Pool, in milliseconds.	50000	No
slot.request.timeout	Timeout for requesting a slot from Slot Pool, in milliseconds.	300000	No
task.cancellation.timeout	Timeout of task cancellation, in milliseconds. If a task cancellation times out, a fatal TaskManager error may occur. If this parameter is set to 0 , no error is reported when a task cancellation times out.	180000	No

Parameter	Description	Default Value	Mandatory
taskmanager.network.detailed-metrics	Indicates whether to enable the detailed metrics monitoring of network queue lengths.	false	No
taskmanager.network.memory.buffer-per-channel	Maximum number of network buffers used by each output/input channel (sub-partition/incoming channel). In credit-based flow control mode, this indicates how much credit is in each input channel. It should be configured with at least 2 buffers to deliver good performance. One buffer is used to receive in-flight data in the sub-partition, and the other for parallel serialization.	2	No

Parameter	Description	Default Value	Mandatory
taskmanager.network.memory.floating-buffers-per-gate	<p>Number of extra network buffers used by each output gate (result partition) or input gate, indicating the amount of floating credit shared among all input channels in credit-based flow control mode. Floating buffers are distributed based on the backlog feedback (real-time output buffers in sub-partitions) and can help mitigate back pressure caused by unbalanced data distribution among sub-partitions. Increase this value if the round-trip time between nodes is long and/or the number of machines in the cluster is large.</p>	8	No

Parameter	Description	Default Value	Mandatory
taskmanager.network.memory.fraction	Ratio of JVM memory used for network buffers, which determines how many streaming data exchange channels a TaskManager can have at the same time and the extent of channel buffering. Increase this value or the values of taskmanager.network.memory.min and taskmanager.network.memory.max if the job is rejected or a warning indicating that the system does not have enough buffers is received. Note that the values of taskmanager.network.memory.min and taskmanager.network.memory.max may overwrite this value.	0.1	No
taskmanager.network.memory.max	Maximum memory size of the network buffer. The value must contain a unit (B/KB/MB/GB/TB).	1 GB	No

Parameter	Description	Default Value	Mandatory
taskmanager.network.memory.min	Minimum memory size of the network buffer. The value must contain a unit (B/KB/MB/GB/TB).	64 MB	No
taskmanager.network.request-backoff.initial	Minimum backoff for partition requests of input channels.	100	No
taskmanager.network.request-backoff.max	Maximum backoff for partition requests of input channels.	10000	No
taskmanager.registration.timeout	Timeout for TaskManager registration. TaskManager will be terminated if it is not successfully registered within the specified time. The value must contain a time unit (ms/s/min/h/d).	5 min	No
resourcemanager.taskmanager.timeout	Timeout interval for releasing an idle TaskManager, in milliseconds.	30000	No

6.3.3 Blob

Scenarios

The Blob server on the JobManager node is used to receive JAR files uploaded by users on the client, send JAR files to TaskManager, and transfer log files. Flink provides some items for configuring the Blob server. You can configure them in the **flink-conf.yaml** configuration file.

Configuration Description

Users can configure the port, SSL, retry times, and concurrency.

Table 6-3 Parameters

Parameter	Description	Default Value	Mandatory
blob.server.port	Blob server port	32456 to 32520	No
blob.service.ssl.enabled	Indicates whether to enable the encryption for the blob transmission channel. This parameter is valid only when the global switch security.ssl is enabled.	true	Yes
blob.fetch.retries	Number of times that TaskManager tries to download blob files from JobManager.	50	No
blob.fetch.num-concurrent	Number of concurrent tasks for downloading blob files supported by JobManager.	50	No
blob.fetch.backlog	Number of blob files, such as .jar files, to be downloaded in the queue supported by JobManager. The unit is count.	1000	No
library-cache-manager.cleanup.interval	Interval at which JobManager deletes the JAR files stored on the HDFS when the user cancels the Flink job. The unit is second.	3600	No

 **NOTE**

For versions earlier than MRS 3.x, **library-cache-manager.cleanup.interval** cannot be configured.

6.3.4 Distributed Coordination (via Akka)

Scenarios

The Akka actor model is the basis of communications between the Flink client and JobManager, JobManager and TaskManager, as well as TaskManager and TaskManager. Flink enables you to configure the Akka connection parameters in the **flink-conf.yaml** file based on the network environment or optimization policy.

Configuration Description

You can configure timeout settings of message sending and waiting, and the Akka listening mechanism Deathwatch.

For versions earlier than MRS 3.x, see [Table 6-4](#).

Table 6-4 Parameters

Parameter	Mandatory	Default Value	Description
akka.ask.timeout	No	10 s	Timeout duration of Akka asynchronous and block requests. If a Flink timeout failure occurs, this value can be increased. Timeout occurs when the machine processing speed is slow or the network is blocked. The unit is ms/s/m/h/d.
akka.lookup.timeout	No	10 s	Timeout duration for JobManager actor object searching. The unit is ms/s/m/h/d.
akka.framesize	No	10485760 b	Maximum size of the message transmitted between JobManager and TaskManager. If a Flink error occurs because the message exceeds this limit, the value can be increased. The unit is b/B/KB/MB.
akka.watch.heartbeat.interval	No	10 s	Heartbeat interval at which the Akka DeathWatch mechanism detects disconnected TaskManager. If TaskManager is frequently and incorrectly marked as disconnected due to heartbeat loss or delay, the value can be increased. The unit is ms/s/m/h/d.

Parameter	Mandatory	Default Value	Description
akka.watch.heartbeat.pause	No	60 s	Acceptable heartbeat pause for Akka DeathWatch mechanism. A small value indicates that irregular heartbeat is not accepted. The unit is ms/s/m/h/d.
akka.watch.threshold	No	12	DeathWatch failure detection threshold. A small value is prone to mark normal TaskManager as failed and a large value increases failure detection time.
akka.tcp.timeout	No	20 s	Timeout duration of Transmission Control Protocol (TCP) connection request. If TaskManager connection timeout occurs frequently due to the network congestion, the value can be increased. The unit is ms/s/m/h/d.
akka.throughput	No	15	Number of messages processed by Akka in batches. After an operation, the processing thread is returned to the thread pool. A small value indicates the fair scheduling for actor message processing. A large value indicates improved overall performance but lowered scheduling fairness.
akka.log.lifecycle.events	No	false	Switch of Akka remote time logging, which can be enabled for debugging.
akka.startup-timeout	No	The default value is the same as the value of akka.ask.timeout .	Timeout duration of remote component started by Akka. The unit is ms/s/m/h/d.
akka.ssl.enabled	Yes	true	Switch of Akka communication SSL. This parameter is valid only when the global switch security.ssl is enabled.

For configuration items for MRS 3.x or later, see [Table 6-5](#).

Table 6-5 Parameters

Parameter	Description	Default Value	Mandatory
akka.ask.timeout	Timeout duration of Akka asynchronous and block requests. If a Flink timeout failure occurs, this value can be increased. Timeout occurs when the machine processing speed is slow or the network is blocked. The unit is ms/s/m/h/d.	10s	No
akka.lookup.timeout	Timeout duration for JobManager actor object searching. The unit is ms/s/m/h/d.	10s	No
akka.framesize	Maximum size of the message transmitted between JobManager and TaskManager. If a Flink error occurs because the message exceeds this limit, the value can be increased. The unit is b/B/KB/MB.	10485760b	No

Parameter	Description	Default Value	Mandatory
akka.watch.heartbeat.interval	Heartbeat interval at which the Akka DeathWatch mechanism detects disconnected TaskManager. If TaskManager is frequently and incorrectly marked as disconnected due to heartbeat loss or delay, the value can be increased. The unit is ms/s/m/h/d.	10s	No
akka.watch.heartbeat.pause	Acceptable heartbeat pause for Akka DeathWatch mechanism. A small value indicates that irregular heartbeat is not accepted. The unit is ms/s/m/h/d.	60s	No
akka.watch.threshold	DeathWatch failure detection threshold. A small value may mark normal TaskManager as failed and a large value increases failure detection time.	12	No

Parameter	Description	Default Value	Mandatory
akka.tcp.timeout	Timeout duration of Transmission Control Protocol (TCP) connection request. If TaskManager connection timeout occurs frequently due to the network congestion, the value can be increased. The unit is ms/s/m/h/d.	20s	No
akka.throughput	Number of messages processed by Akka in batches. After an operation, the processing thread is returned to the thread pool. A small value indicates the fair scheduling for actor message processing. A large value indicates improved overall performance but lowered scheduling fairness.	15	No
akka.log.lifecycle.events	Switch of Akka remote time logging, which can be enabled for debugging.	false	No
akka.startup-timeout	Timeout interval before a remote component fails to be started. The value must contain a time unit (ms/s/min/h/d).	The default value is the same as the value of akka.ask.timeout .	No

Parameter	Description	Default Value	Mandatory
akka.ssl.enabled	Switch of Akka communication SSL. This parameter is valid only when the global switch security.ssl is enabled.	true	Yes
akka.client-socket-worker-pool.pool-size-factor	Factor that is used to determine the thread pool size. The pool size is calculated based on the following formula: $\text{ceil}(\text{available processors} * \text{factor})$. The size is bounded by the pool-size-min and pool-size-max values.	1.0	No
akka.client-socket-worker-pool.pool-size-max	Maximum number of threads calculated based on the factor.	2	No
akka.client-socket-worker-pool.pool-size-min	Minimum number of threads calculated based on the factor.	1	No
akka.client.timeout	Timeout duration of the client. The value must contain a time unit (ms/s/min/h/d).	60s	No

Parameter	Description	Default Value	Mandatory
akka.server-socket-worker-pool.pool-size-factor	Factor that is used to determine the thread pool size. The pool size is calculated based on the following formula: $\text{ceil}(\text{available processors} * \text{factor})$. The size is bounded by the pool-size-min and pool-size-max values.	1.0	No
akka.server-socket-worker-pool.pool-size-max	Maximum number of threads calculated based on the factor.	2	No
akka.server-socket-worker-pool.pool-size-min	Minimum number of threads calculated based on the factor.	1	No

6.3.5 SSL

Scenarios

When the secure Flink cluster is required, SSL-related configuration items must be set.

Configuration Description

Configuration items include the SSL switch, certificate, password, and encryption algorithm.

For versions earlier than MRS 3.x, see [Table 6-6](#).

Table 6-6 Parameters

Parameter	Mandatory	Default Value	Description
security.ssl.internal.enabled	Yes	The value is automatically configured according to the cluster installation mode. <ul style="list-style-type: none"> Security mode: The default value is true. Normal mode: The default value is false. 	Main switch of internal communication SSL.
security.ssl.internal.keystore	Yes	-	Java keystore file.
security.ssl.internal.keystore-password	Yes	-	Password used to decrypt the keystore file.
security.ssl.internal.key-password	Yes	-	Password used to decrypt the server key in the keystore file.
security.ssl.internal.truststore	Yes	-	truststore file containing the public CA certificates.
security.ssl.internal.truststore-password	Yes	-	Password used to decrypt the truststore file.
security.ssl.protocol	Yes	TLSv1.2	SSL transmission protocol version

Parameter	Mandatory	Default Value	Description
security.ssl.algorithm	Yes	The default value is TLS_RSA_WITH_AES_128_CBC_SHA256,TLS_DHE_RSA_WITH_AES_128_CBC_SHA256,TLS_DHE_DSS_WITH_AES_128_CBC_SHA256 .	Supported SSL standard algorithm.
security.ssl.rest.enabled	Yes	The value is automatically configured according to the cluster installation mode. <ul style="list-style-type: none"> • Security mode: The default value is true. • Normal mode: The default value is false. 	Main switch of external communication SSL.
security.ssl.rest.keystore	Yes	-	Java keystore file.
security.ssl.rest.keystore-password	Yes	-	Password used to decrypt the keystore file.

Parameter	Mandatory	Default Value	Description
security.ssl.rest.keystore-password	Yes	-	Password used to decrypt the server key in the keystore file.
security.ssl.rest.truststore	Yes	-	truststore file containing the public CA certificates.
security.ssl.rest.truststore-password	Yes	-	Password used to decrypt the truststore file.

For configuration items for MRS 3.x or later, see [Table 6-7](#).

Table 6-7 Parameters

Parameter	Description	Default Value	Mandatory
security.ssl.enabled	Main switch of internal communication SSL.	The value is automatically configured according to the cluster installation mode. <ul style="list-style-type: none"> Security mode: The default value is true. Non-security mode: The default value is false. 	Yes
security.ssl.keystore	Java keystore file.	-	Yes
security.ssl.keystore-password	Password used to decrypt the keystore file.	-	Yes
security.ssl.key-password	Password used to decrypt the server key in the keystore file.	-	Yes
security.ssl.truststore	truststore file containing the public CA certificates.	-	Yes
security.ssl.truststore-password	Password used to decrypt the truststore file.	-	Yes

Parameter	Description	Default Value	Mandatory
security.ssl.protocol	SSL transmission protocol version.	TLSv1.2	Yes
security.ssl.algorithms	Supported SSL standard algorithm.	The default value: "TLS_DHE_RSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,TLS_DHE_RSA_WITH_AES_256_GCM_SHA384,TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384"	Yes

6.3.6 Network communication (via Netty)

Scenario

When Flink runs a job, data transmission and reverse pressure detection between tasks depend on Netty. In certain environments, **Netty** parameters should be configured.

Configuration Description

For advanced optimization, you can modify the following Netty configuration items. The default configuration can meet the requirements of tasks of large-scale clusters with high concurrent throughput. For details about the parameters, visit the Netty official website at <http://netty.io/>.

Table 6-8 Parameter description

Parameter	Description	Default Value	Mandatory
taskmanager.network.netty.num-arenas	Number of Netty memory blocks.	1	No
taskmanager.network.netty.server.numThreads	Number of Netty server threads	1	No
taskmanager.network.netty.client.numThreads	Number of Netty client threads	1	No

Parameter	Description	Default Value	Mandatory
taskmanager.network.netty.client.connectTimeoutSec	Netty client connection timeout duration. Unit: second	120	No
taskmanager.network.netty.sendReceiveBufferSize	Size of Netty sending and receiving buffers. This defaults to the system buffer size (<code>cat /proc/sys/net/ipv4/tcp_[rw]mem</code>) and is 4 MB in modern Linux. Unit: byte	4096	No
taskmanager.network.netty.transport	Netty transport type, either nio or epoll	nio	No

6.3.7 JobManager Web Frontend

Scenarios

When JobManager is started, the web server in the same process is also started.

- You can access the web server to obtain information about the current Flink cluster, including information about JobManager, TaskManager, and running jobs in the cluster.
- You can configure parameters of the web server.

Configuration Description

Configuration items include the port, temporary directory, display items, error redirection, and security-related items.

For versions earlier than MRS 3.x, see [Table 6-9](#).

Table 6-9 Parameters

Parameter	Mandatory	Default Value	Description
jobmanager.web.port	No	32261-32325	Web port. Value range: 32261-32325.

Parameter	Mandatory	Default Value	Description
jobmanager.web.allow-access-address	Yes	*	Web access whitelist. IP addresses are separated by commas (,). Only IP addresses in the whitelist can access the web.

For details about configuration items of MRS 3.x or later, see [Table 6-10](#).

Table 6-10 Parameters

Parameter	Description	Default Value	Mandatory
flink.security.enable	<p>When installing a Flink cluster, you are required to select security mode or normal mode.</p> <ul style="list-style-type: none"> • If security mode is selected, the value of flink.security.enable is automatically set to true. • If normal mode is selected, the value of flink.security.enable is automatically set to false. <p>If you want to check whether Flink cluster is in security mode or normal mode, view the value of flink.security.enable.</p>	The value is automatically configured based on the cluster installation mode.	No
rest.bind-port	Web port. Value range: 32261-32325.	32261-32325	No

Parameter	Description	Default Value	Mandatory
jobmanager.web.history	Number of recent jobs to be displayed.	5	No
jobmanager.web.checkpoints.disable	Indicates whether to disable checkpoint statistics.	false	No
jobmanager.web.checkpoints.history	Number of checkpoint statistical records.	10	No
jobmanager.web.backpressure.clean-up-interval	Interval for clearing unaccessed backpressure records. The unit is millisecond.	600000	No
jobmanager.web.backpressure.refresh-interval	Interval for updating backpressure records. The unit is millisecond.	60000	No
jobmanager.web.backpressure.num-samples	Number of stack tracing records for reverse pressure calculation.	100	No
jobmanager.web.backpressure.delay-between-samples	Sampling interval for reverse pressure calculation. The unit is millisecond.	50	No
jobmanager.web.ssl.enabled	Whether SSL encryption is enabled for web transmission. This parameter is valid only when the global switch security.ssl is enabled.	false	Yes
jobmanager.web.accesslog.enable	Switch to enable or disable web operation logs. The log is stored in webaccess.log .	true	Yes

Parameter	Description	Default Value	Mandatory
jobmanager.web.x-frame-options	Value of the HTTP security header X-Frame-Options . The value can be SAMEORIGIN , DENY , or ALLOW-FROM uri .	DENY	Yes
jobmanager.web.cache-directive	Whether the web page can be cached.	no-store	Yes
jobmanager.web.expires-time	Expiration duration of web page cache. The unit is millisecond.	0	Yes
jobmanager.web.allow-access-address	Web access whitelist. IP addresses are separated by commas (.). Only IP addresses in the whitelist can access the web.	*	Yes
jobmanager.web.access-control-allow-origin	Web page same-origin policy that prevents cross-domain attacks.	*	Yes
jobmanager.web.refresh-interval	Web page refresh interval. The unit is millisecond.	3000	Yes
jobmanager.web.logout-timer	Automatic logout interval when no operation is performed. The unit is millisecond.	600000	Yes
jobmanager.web.403-redirect-url	Web page access error 403. If 403 error occurs, the page switch to a specified page.	Automatic configuration	Yes
jobmanager.web.404-redirect-url	Web page access error 404. If 404 error occurs, the page switch to a specified page.	Automatic configuration	Yes

Parameter	Description	Default Value	Mandatory
jobmanager.web.415-redirect-url	Web page access error 415. If 415 error occurs, the page switch to a specified page.	Automatic configuration	Yes
jobmanager.web.500-redirect-url	Web page access error 500. If 500 error occurs, the page switch to a specified page.	Automatic configuration	Yes
rest.await-leader-timeout	Time of the client waiting for the leader address. The unit is millisecond.	30000	No
rest.client.max-content-length	Maximum content length that the client handles (unit: bytes).	104857600	No
rest.connection-timeout	Maximum time for the client to establish a TCP connection (unit: ms).	15000	No
rest.idleness-timeout	Maximum time for a connection to stay idle before failing (unit: ms).	300000	No
rest.retry.delay	The time that the client waits between retries (unit: ms).	3000	No
rest.retry.max-attempts	The number of retry times if a retrievable operator fails.	20	No
rest.server.max-content-length	Maximum content length that the server handles (unit: bytes).	104857600	No

Parameter	Description	Default Value	Mandatory
rest.server.numThreads	Maximum number of threads for the asynchronous processing of requests.	4	No
web.timeout	Timeout for web monitor (unit: ms).	10000	No

6.3.8 File Systems

Scenario

Result files are created when tasks are running. Flink enables you to configure parameters for file creation.

Configuration Description

Configuration items include overwriting policy and directory creation.

Table 6-11 Parameter description

Parameter	Description	Default Value	Mandatory
fs.overwrite-files	Whether to overwrite the existing file by default when the file is written.	false	No

Parameter	Description	Default Value	Mandatory
fs.output.always-create-directory	<p>When the degree of parallelism (DOP) of file writing programs is greater than 1, a directory is created under the output file path and different result files (one for each parallel writing program) are stored in the directory.</p> <ul style="list-style-type: none"> • If this parameter is set to true, a directory is created for the writing program whose DOP is 1 and a result file is stored in the directory. • If this parameter is set to false, the file of the writing program whose DOP is 1 is created directly in the output path and no directory is created. 	false	No

6.3.9 State Backend

Scenarios

Flink enables HA and job exception, as well as job pause and recovery during version upgrade. Flink depends on state backend to store job states and on the restart strategy to restart a job. You can configure state backend and the restart strategy.

Configuration Description

Configuration items include the state backend type, storage path, and restart strategy.

Table 6-12 Parameters

Parameter	Description	Default Value	Mandatory
state.backend.fs.checkpointdir	Path when the backend is set to filesystem . The path must be accessible by JobManager. Only the local mode is supported. In the cluster mode, use an HDFS path.	hdfs:///flink/checkpoints	No
state.savepoints.dir	Savepoint storage directory used by Flink to restore and update jobs. When a savepoint is triggered, the metadata of the savepoint is saved to this directory.	hdfs:///flink/savepoint	Mandatory in security mode
restart-strategy	Default restart policy, which is used for jobs for which no restart policy is specified. The options are as follows: <ul style="list-style-type: none"> fixed-delay failure-rate none 	none	No
restart-strategy.fixed-delay.attempts	Number of retry times when the fixed-delay restart strategy is used.	<ul style="list-style-type: none"> If the checkpoint is enabled, the default value is the value of Integer.MAX_VALUE. If the checkpoint is disabled, the default value is 3. 	No

Parameter	Description	Default Value	Mandatory
restart-strategy.fixed-delay.delay	Retry interval when the fixed-delay strategy is used. The unit is ms/s/m/h/d.	<ul style="list-style-type: none"> If the checkpoint is enabled, the default value is 10s. If the checkpoint is disabled, the default value is the value of akka.ask.timeout. 	No
restart-strategy.failure-rate.max-failures-per-interval	Maximum number of restart times in a specified period before a job fails when the fault rate policy is used.	1	No
restart-strategy.failure-rate.failure-rate-interval	Retry interval when the failure-rate strategy is used. The unit is ms/s/m/h/d.	60 s	No
restart-strategy.failure-rate.delay	Retry interval when the failure-rate strategy is used. The unit is ms/s/m/h/d.	The default value is the same as the value of akka.ask.timeout .	No

6.3.10 Kerberos-based Security

Scenarios

Flink Kerberos configuration items must be configured in security mode.

Configuration Description

The configuration items include **keytab**, **principal**, and **cookie** of Kerberos.

NOTE

For versions earlier than MRS 3.x, the configuration item does not contain cookie.

Table 6-13 Parameters

Parameter	Description	Default Value	Mandatory
security.kerberos.login.keytab	Keytab file path. This parameter is a client parameter.	Configure the parameter based on actual service requirements.	Yes
security.kerberos.login.principal	A parameter on the client. If security.kerberos.login.keytab and security.kerberos.login.principal are both set, keytab certificate is used by default.	Configure the parameter based on actual service requirements.	No
security.kerberos.login.contexts	Contexts of the jass file generated by Flink. This parameter is a server parameter.	Client, KafkaClient	Yes
security.enable	Certificate enabling switch of the Flink internal module. This parameter is a client parameter.	This parameter is configured automatically according to the cluster installation mode. <ul style="list-style-type: none"> • Security mode: The default value is true. • Non-security mode: The default value is false. 	Yes
security.cookie	Module certificate token. This parameter is a client parameter. It must be configured and cannot be left empty when security.enable is enabled.	Configure the parameter based on actual service requirements.	Yes

 NOTE

For versions earlier than MRS 3.x, the configuration parameters do not include `security.enable` and `security.cookie`.

6.3.11 HA

Scenarios

The Flink HA mode depends on ZooKeeper. Therefore, ZooKeeper-related configuration items must be set.

Configuration Description

Configuration items include the ZooKeeper address, path, and security certificate.

Table 6-14 Parameters

Parameter	Description	Default Value	Mandatory
high-availability	<p>Whether HA is enabled. Only the following two modes are supported currently:</p> <ol style="list-style-type: none"> 1. none: Only a single JobManager is running. The checkpoint is disabled for JobManager. 2. ZooKeeper: <ul style="list-style-type: none"> • In non-Yarn mode, multiple JobManagers are supported and the leader JobManager is elected. • In Yarn mode, only one JobManager exists. 	zookeeper	No

Parameter	Description	Default Value	Mandatory
high-availability.zookeeper.per.quorum	ZooKeeper quorum address.	Automatic configuration	No
high-availability.zookeeper.per.path.root	Root directory that Flink creates on ZooKeeper, storing metadata required in HA mode.	/flink	No
high-availability.storageDir	Directory for storing JobManager metadata of state backend. ZooKeeper stores only pointers to actual data.	hdfs:///flink/recovery	No
high-availability.zookeeper.per.client.session-timeout	Session timeout duration on the ZooKeeper client. The unit is millisecond.	60000	No
high-availability.zookeeper.per.client.connection-timeout	Connection timeout duration on the ZooKeeper client. The unit is millisecond.	15000	No
high-availability.zookeeper.per.client.retry-wait	Retry waiting time on the ZooKeeper client. The unit is millisecond.	5000	No
high-availability.zookeeper.per.client.max-retry-attempts	Maximum retry times on the ZooKeeper client.	3	No
high-availability.job.delay	Delay of job restart when JobManager recovers.	The default value is the same as the value of akka.ask.timeout .	No

Parameter	Description	Default Value	Mandatory
high-availability.zookeeper.client.acl	ACL (open creator) of the ZooKeeper node.	This parameter is configured automatically according to the cluster installation mode. <ul style="list-style-type: none"> Security mode: The default value is creator. Non-security mode: The default value is open. 	Yes
zookeeper.sasl.disable	Simple authentication and security layer (SASL)-based certificate enable switch.	This parameter is configured automatically according to the cluster installation mode. <ul style="list-style-type: none"> Security mode: The default value is false. Non-security mode: The default value is true. 	Yes
zookeeper.sasl.service-name	<ul style="list-style-type: none"> If the ZooKeeper server configures a service whose name is different from ZooKeeper, this configuration item can be set. If service names on the client and server are inconsistent, authentication fails. 	zookeeper	Yes

 **NOTE**

For versions earlier than MRS 3.x, the **high-availability.job.delay** parameter is not supported.

6.3.12 Environment

Scenario

In scenarios raising special requirements on JVM configuration, users can use configuration items to transfer JVM parameters to the client, JobManager, and TaskManager.

Configuration

Configuration items include JVM parameters.

Table 6-15 Parameter description

Parameter	Description	Default Value	Mandatory
env.java.opts	JVM parameter, which is transferred to the startup script, JobManager, TaskManager, and Yarn client. For example, transfer remote debugging parameters.	- Xloggc:<LOG_DIR >/gc.log -XX: +PrintGCDetails -XX:- OmitStackTraceln- FastThrow -XX: +PrintGCTimeSta mps -XX: +PrintGCDateSta mps -XX: +UseGCLogFileRot ation - XX:NumberOfGCL ogFiles=20 - XX:GCLogFileSiz e= 20M - Djdk.tls.ephemera LDHKeySize=2048 - Djava.library.path =\$ {HADOOP_COMM ON_HOME}/lib/ native - Djava.net.preferIP v4Stack=true - Djava.net.preferIP v6Addresses=false - Dbeetle.applicatio n.home.path=\$ {BIGDATA_HOME }/common/ runtime/security/ config	No

6.3.13 Yarn

Scenario

Flink runs on a Yarn cluster and JobManager runs on ApplicationMaster. Certain configuration parameters of JobManager depend on Yarn. By setting Yarn-related configuration items, Flink is enabled to run better on Yarn.

Configuration Description

The configuration items include the memory, virtual kernel, and port of the Yarn container.

Table 6-16 Parameter description

Parameter	Description	Default Value	Mandatory
yarn.maximum-failed-containers	Maximum number of containers the system is going to reallocate in case of a container failure of TaskManager. The default value is the number of TaskManagers when the Flink cluster is started.	5	No
yarn.application-attempts	Number of ApplicationMaster restarts. The value is the maximum value in the validity interval that is set to Akka's timeout in Flink. After the restart, the IP address and port number of ApplicationMaster will change and you will need to connect to the client manually.	2	No
yarn.heartbeat-delay	Time between heartbeats with the ApplicationMaster and Yarn ResourceManager in seconds. Unit: second	5	No
yarn.containers.vcores	Number of virtual cores of each Yarn container	The default value is the number of TaskManager slots.	No

Parameter	Description	Default Value	Mandatory
yarn.application-master.port	ApplicationMaster port number setting. A port number range is supported.	32586-32650	No

6.3.14 Pipeline

Scenarios

The Netty connection is used among multiple jobs to reduce latency. In this case, NettySink is used on the server and NettySource is used on the client for data transmission.

This section applies to MRS 3.x or later.

Configuration Description

Configuration items include NettySink information storing path, range of NettySink listening port, whether to enable SSL encryption, domain of the network used for NettySink monitoring.

Table 6-17 Parameters

Parameter	Description	Default Value	Mandatory
nettyconnector.registerserver.topic.storage	Path (on a third-party server) to information about IP address, port numbers, and concurrency of NettySink. ZooKeeper is recommended for storage.	/flink/nettyconnector	No. However, if pipeline is enabled, the feature is mandatory.
nettyconnector.sinkserver.port.range	Port range of NettySink.	If MRS cluster is used, the default value is 28444-28843.	No. However, if pipeline is enabled, the feature is mandatory.

Parameter	Description	Default Value	Mandatory
nettyconnector.ssl.enabled	Whether SSL encryption for the communication between NettySink and NettySource is enabled. For details about the encryption key and protocol, see SSL .	false	No. However, if pipeline is enabled, the feature is mandatory.
nettyconnector.message.delimiter	Delimiter used to configure the message sent by NettySink to the NettySource, which is 2-4 bytes long, and cannot contain \n, #, or space.	The default value is \$._.	No. However, if pipeline is enabled, the feature is mandatory.

6.4 Security Configuration

6.4.1 Security Features

Security Features of Flink

- All Flink cluster components support authentication.
 - The Kerberos authentication is supported between Flink cluster components and external components, such as Yarn, HDFS, and ZooKeeper.
 - The security cookie authentication between Flink cluster components, for example, Flink client and JobManager, JobManager and TaskManager, and TaskManager and TaskManager, are supported.
- SSL encrypted transmission is supported by Flink cluster components.
- SSL encrypted transmission between Flink cluster components, for example, Flink client and JobManager, JobManager and TaskManager, and TaskManager and TaskManager, are supported.
- Following security hardening approaches for Flink web are supported:
 - Whitelist filtering. Flink web can only be accessed through Yarn proxy.
 - Security header enhancement.
- In Flink clusters, ranges of listening ports of components can be configured.
- In HA mode, ACL control is supported.

6.4.2 Configuring Kafka

Sample project data of Flink is stored in Kafka. A user with Kafka permission can send data to Kafka and receive data from it.

Step 1 Ensure that clusters, including HDFS, Yarn, Flink, and Kafka are installed.

Step 2 Create a topic.

- Run Linux command line to create a topic. Before running commands, ensure that the kinit command, for example, **kinit flinkuser**, is run for authentication.

NOTE

To create a Flink user, you need to have the permission to create Kafka topics.

The format of the command is shown as follows, in which **{zkQuorum}** indicates ZooKeeper cluster information and the format is *IP.port*, and **{Topic}** indicates the topic name.

```
bin/kafka-topics.sh --create --zookeeper {zkQuorum}/kafka --replication-factor 1 --partitions 5 --topic {Topic}
```

Assume the topic name is **topic 1**. The command for creating this topic is displayed as follows:

```
/opt/client/Kafka/kafka/bin/kafka-topics.sh --create --zookeeper  
10.96.101.32:2181,10.96.101.251:2181,10.96.101.177:2181,10.91.8.160:2181/kafka --replication-factor  
1 --partitions 5 --topic topic1
```

- Configure the permission of the topic on the server.
Set the **allow.everyone.if.no.acl.found** parameter of Kafka Broker to **true**.

Step 3 Perform the security authentication.

The Kerberos authentication, SSL encryption authentication, or Kerberos + SSL authentication mode can be used.

NOTE

For versions earlier than MRS 3.x, only Kerberos authentication is supported.

- **Kerberos authentication**

– Client configuration

In the Flink configuration file **flink-conf.yaml**, add configurations about Kerberos authentication. For example, add **KafkaClient** in **contexts** as follows:

```
security.kerberos.login.keytab: /home/demo//keytab/flinkuser.keytab  
security.kerberos.login.principal: flinkuser  
security.kerberos.login.contexts: Client,KafkaClient  
security.kerberos.login.use-ticket-cache: false
```

NOTE

For versions earlier than MRS 3.x, set **security.kerberos.login.keytab** to **/home/demo/flink/release/keytab/flinkuser.keytab**.

– Running parameter

Running parameters about the **SASL_PLAINTEXT** protocol are as follows:

```
--topic topic1 --bootstrap.servers 10.96.101.32:21007 --security.protocol SASL_PLAINTEXT --  
sasl.kerberos.service.name kafka //10.96.101.32:21007 indicates the IP.port of the Kafka server.
```

- **SSL encryption**
 - Configuration on the server
Log in to FusionInsight Manager, choose **Cluster > Services > Kafka > Configurations**, and set **Type** to **All**. Search for **ssl.mode.enable** and set it to **true**.
 - Configuration on the client
 - i. Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Kafka > More > Download Client** to download Kafka client.
 - ii. Use the **ca.crt** certificate file in the client root directory to generate the **truststore** file for the client.

Run the following command:

```
keytool -noprompt -import -alias myservcert -file ca.crt -keystore truststore.jks
```

The command execution result is similar to the following:

```
drwx-----, 5 zgd users 4096 Feb 4 16:22 .
drwxr-xr-x, 10 zgd users 4096 Jan 22 17:38 ..
-rwx-----, 1 zgd users 135 Jan 22 17:31 application.properties
-rwx-----, 1 zgd users 790 Jan 22 17:31 bigdata_env.sample
-rw-----, 1 zgd users 1322 Jan 22 17:31 ca.crt
-rwx-----, 1 zgd users 4508 Jan 22 17:31 conf.py
-rw-----, 1 zgd users 120 Jan 22 17:31 hosts
-rwx-----, 1 zgd users 745 Jan 22 17:31 install.bat
-rwx-----, 1 zgd users 15082 Jan 22 17:31 install.sh
drwx-----, 2 zgd users 4096 Jan 22 17:38 JDK
-rwx-----, 1 zgd users 37021723 Jan 22 17:31 jython-standalone-2.7.0.jar
drwx-----, 5 zgd users 4096 Jan 22 17:38 Kafka
drwx-----, 3 zgd users 4096 Jan 22 17:38 KrbClient
-rwx-----, 1 zgd users 473 Jan 22 17:31 log4j.properties
-rwx-----, 1 zgd users 2107 Jan 22 17:31 README
-rwx-----, 1 zgd users 6949 Jan 22 17:31 refreshConfig.sh
-rwx-----, 1 zgd users 1736 Jan 22 17:31 switchuser.py
-rw-r--r--, 1 root root 1004 Feb 4 16:22 truststore.jks
```

- iii. Run parameters.

The value of **ssl.truststore.password** must be the same as the password you entered when creating **truststore**. Run the following command to run parameters:

```
--topic topic1 --bootstrap.servers 10.96.101.32:9093 --security.protocol SSL --
ssl.truststore.location /home/zgd/software/FusionInsight_Kafka_ClientConfig/truststore.jks
--ssl.truststore.password XXX
```

- **Kerberos+SSL encryption**

After completing preceding configurations of the client and server of Kerberos and SSL, modify the port number and protocol type in running parameters to enable the Kerberos+SSL encryption mode.

```
--topic topic1 --bootstrap.servers 10.96.101.32:21009 --security.protocol SASL_SSL --
sasl.kerberos.service.name kafka --ssl.truststore.location /home/zgd/software/
FusionInsight_Kafka_ClientConfig/truststore.jks --ssl.truststore.password XXX
```

----End

6.4.3 Configuring Pipeline

This section applies to MRS 3.x or later.

1. File configuration

- **nettyconnector.registerserver.topic.storage**: (Mandatory) Configures the path (on a third-party server) to information about IP address, port numbers, and concurrency of NettySink. For example:

```
nettyconnector.registerserver.topic.storage: /flink/nettyconnector
```

- **nettyconnector.sinkserver.port.range:** (Mandatory) Configures the range of port numbers of NettySink. For example:

```
nettyconnector.sinkserver.port.range: 28444-28843
```
- **nettyconnector.ssl.enabled:** Configures whether to enable SSL encryption between NettySink and NettySource. The default value is **false**. For example:

```
nettyconnector.ssl.enabled: true
```

2. Security authentication configuration

- SASL authentication of ZooKeeper depends on the HA configuration in the **flink-conf.yaml** file.
- SSL configurations such as keystore, truststore, keystore password, truststore password, and password inherit from **flink-conf.yaml**. For details, see [Encrypted Transmission](#).

6.5 Security Hardening

6.5.1 Authentication and Encryption

Security Authentication

Flink uses the following three authentication modes:

- Kerberos authentication: It is used between the Flink Yarn client and Yarn ResourceManager, JobManager and ZooKeeper, JobManager and HDFS, TaskManager and HDFS, Kafka and TaskManager, as well as TaskManager and ZooKeeper.
- Security cookie authentication: Security cookie authentication is used between Flink Yarn client and JobManager, JobManager and TaskManager, as well as TaskManager and TaskManager.
- Internal authentication of Yarn: The Internal authentication mechanism of Yarn is used between Yarn ResourceManager and ApplicationMaster (AM).

NOTE

- Flink JobManager and Yarn ApplicationMaster are in the same process.
- If Kerberos authentication is enabled for the user's cluster, Kerberos authentication is required.
- For versions earlier than MRS 3.x, Flink does not support the security cookie authentication mode.

Table 6-18 Authentication modes

Authen- tication Mode	Descrip- tion	Configuration Method
Kerbero- s authent- ication	Current- ly, only keytab authent- ication mode is support- ed.	<ol style="list-style-type: none"> 1. Download the user keytab from the KDC server, and place the keytab to a directory on the host of the Flink client. 2. Configure the following parameters in the flink-conf.yaml file: <ol style="list-style-type: none"> a. Keytab path <code>security.kerberos.login.keytab: /home/flinkuser/keytab/abc222.keytab</code> Note: /home/flinkuser/keytab/abc222.keytab indicates the user directory. b. Principal name <code>security.kerberos.login.principal: abc222</code> c. In HA mode, if ZooKeeper is configured, the Kerberos authentication configuration items must be configured as follows: <code>zookeeper.sasl.disable: false</code> <code>security.kerberos.login.contexts: Client</code> d. If you want to perform Kerberos authentication between Kafka client and Kafka broker, set the value as follows: <code>security.kerberos.login.contexts: Client,KafkaClient</code>

Authen- tication Mode	Descrip- tion	Configuration Method
Security cookie authent- ication	-	<p>1. In the bin directory of the Flink client, run the generate_keystore.sh script to generate security cookie, flink.keystore, and flink.truststore. Run the sh generate_keystore.sh command and enter the user-defined password. The password cannot contain #.</p> <p>NOTE After the script is executed, the flink.keystore and flink.truststore files are generated in the conf directory on the Flink client. In the flink-conf.yaml file, default values are specified for following parameters:</p> <ul style="list-style-type: none"> • Set security.ssl.keystore to the absolute path of the flink.keystore file. • Set security.ssl.truststore to the absolute path of the flink.truststore file. • Set security.cookie to a random password automatically generated by the generate_keystore.sh script. • By default, security.ssl.encrypt.enabled: false is set in the flink-conf.yaml file by default. The generate_keystore.sh script sets security.ssl.key-password, security.ssl.keystore-password, and security.ssl.truststore-password to the password entered when the generate_keystore.sh script is called. <p>2. Set security.enable: true in the flink-conf.yaml file and check whether security cookie is configured successfully. Example: <pre>security.cookie: ae70acc9-9795-4c48- ad35-8b5adc8071744f605d1d-2726-432e-88ae-dd39bfec40a9</pre></p> <p>NOTE Obtain the SSL certificate and save it to the Flink client.</p>
Internal authent- ication of Yarn	This authent- ication mode does not need to be config- ured by the user.	-

 **NOTE**

One Flink cluster supports only one user. One user can create multiple Flink clusters.

Encrypted Transmission

Flink uses following encrypted transmission modes:

- Encrypted transmission inside Yarn: It is used between the Flink Yarn client and Yarn ResourceManager, as well as Yarn ResourceManager and JobManager.
- SSL transmission: SSL transmission is used between Flink Yarn client and JobManager, JobManager and TaskManager, as well as TaskManager and TaskManager.
- Encrypted transmission inside Hadoop: The internal encrypted transmission mode of Hadoop used between JobManager and HDFS, TaskManager and HDFS, JobManager and ZooKeeper, as well as TaskManager and ZooKeeper.

NOTE

Configuration about SSL encrypted transmission is mandatory while configuration about encryption of Yarn and Hadoop is not required.

To configure SSL encrypted transmission, configure the following parameters in the **flink-conf.yaml** file on the client:

1. Enable SSL and configure the SSL encryption algorithm. For MRS 3.x or later, see [Table 6-19](#). Modify the parameters as required.

Table 6-19 Parameter description

Parameter	Example Value	Description
security.ssl.enabled	true	Enable SSL.
akka.ssl.enabled	true	Enable Akka SSL.
blob.service.ssl.enabled	true	Enable SSL for the Blob channel.
taskmanager.data.ssl.enabled	true	Enable SSL transmissions between TaskManagers.
security.ssl.algorithms	TLS_DHE_RSA_WITH_AES_128_CBC_SHA256,TLS_DHE_DSS_WITH_AES_128_CBC_SHA256	Configure the SSL encryption algorithm.

For versions earlier than MRS 3.x, see [Table 6-20](#).

Table 6-20 Parameter description

Parameter	Example Value	Description
security.ssl.internal.enabled	true	Enable internal SSL.

Parameter	Example Value	Description
akka.ssl.enabled	true	Enable Akka SSL.
blob.service.ssl.enabled	true	Enable SSL for the Blob channel.
taskmanager.data.ssl.enabled	true	Enable SSL transmissions between TaskManagers.
security.ssl.algorithms	TLS_RSA_WITH_AES128_CBC_SHA256	Configure the SSL encryption algorithm.

For versions earlier than MRS 3.x, the following parameters in [Table 6-21](#) do not exist in the default Flink configuration of MRS. If you want to enable SSL for external connections, add the following parameters. After SSL for external connection is enabled, the native Flink page cannot be accessed using a Yarn proxy, because the Yarn open-source version cannot process HTTPS requests using a proxy. However, you can create a Windows VM in the same VPC of the cluster and access the native Flink page from the VM.

Table 6-21 Parameter description

Parameter	Example Value	Description
security.ssl.rest.enabled	true	Enable external SSL. If this parameter is set to true , set the related parameters by referring to Table 6-21 .
security.ssl.rest.keystore	\${path}/flink.keystore	Path for storing the keystore .
security.ssl.rest.keystore-password	123456	Password of the keystore . 123456 indicates a user-defined password is required.
security.ssl.rest.key-password	123456	Password of the SSL key. 123456 indicates a user-defined password is required.
security.ssl.rest.truststore	\${path}/flink.truststore	Path for storing the truststore .
security.ssl.rest.truststore-password	123456	Password of the truststore . 123456 indicates a user-defined password is required.

 NOTE

Enabling SSL for data transmission between TaskManagers may pose great impact on the system performance.

2. In the **bin** directory of the Flink client, run the ***sh generate_keystore.sh*** *<password>* command. For details, see [Authentication and Encryption](#). The configuration items in [Table 6-22](#) are set by default for MRS 3.x or later. You can also configure them manually.

Table 6-22 Parameter description

Parameter	Example Value	Description
security.ssl.keystore	\${path}/flink.keystore	Path for storing the keystore . flink.keystore indicates the name of the keystore file generated by the generate_keystore.sh* tool.
security.ssl.keystore-password	123456	Password of the keystore . 123456 indicates a user-defined password is required.
security.ssl.key-password	123456	Password of the SSL key. 123456 indicates a user-defined password is required.
security.ssl.truststore	\${path}/flink.truststore	Path for storing the truststore . flink.truststore indicates the name of the truststore file generated by the generate_keystore.sh* tool.
security.ssl.truststore-password	123456	Password of the truststore . 123456 indicates a user-defined password is required.

For versions earlier than MRS 3.x, the ***generate_keystore.sh*** command is generated automatically, and the configuration items in [Table 6-23](#) are set by default. You can also configure them manually.

Table 6-23 Parameter description

Parameter	Example Value	Description
security.ssl.internal.keystore	\${path}/flink.keystore	Path for storing the keystore . flink.keystore indicates the name of the keystore file generated by the generate_keystore.sh* tool.
security.ssl.internal.keystore-password	123456	Password of the keystore . 123456 indicates a user-defined password is required.
security.ssl.internal.key-password	123456	Password of the SSL key. 123456 indicates a user-defined password is required.
security.ssl.internal.truststore	\${path}/flink.truststore	Path for storing the truststore . flink.truststore indicates the name of the truststore file generated by the generate_keystore.sh* tool.
security.ssl.internal.truststore-password	123456	Password of the truststore . 123456 indicates a user-defined password is required.

For versions earlier than MRS 3.x, if SSL for external connections is enabled, that is, **security.ssl.rest.enabled** is set to **true**, you need to configure the parameters listed in [Table 6-24](#).

Table 6-24 Parameters

Parameter	Example Value	Description
security.ssl.rest.enabled	true	Enable external SSL. If this parameter is set to true , set the related parameters by referring to Table 6-24 .
security.ssl.rest.keystore	\${path}/flink.keystore	Path for storing the keystore .

Parameter	Example Value	Description
security.ssl.rest.keystore -password	123456	Password of the keystore . 123456 indicates a user-defined password is required.
security.ssl.rest.key- password	123456	Password of the SSL key. 123456 indicates a user-defined password is required.
security.ssl.rest.truststor e	\${path}/flink.truststore	Path for storing the truststore .
security.ssl.rest.truststor e-password	123456	Password of the truststore . 123456 indicates a user-defined password is required.

 **NOTE**

The **path** directory is a user-defined directory for storing configuration files of the SSL keystore and truststore. The commands vary according to the relative path and absolute path. For details, see [3](#) and [4](#).

3. If the **keystore** or **truststore** file path is a relative path, the Flink client directory where the command is executed needs to access this relative path directly. Either of the following method can be used to transmit the keystore and truststore file:

- Add **-t** option to the **CLI yarn-session.sh** command to transfer the **keystore** and **truststore** file to execution nodes. Example:

```
./bin/yarn-session.sh -t ssl/
```

- Add **-yt** option to the **flink run** command to transfer the **keystore** and **truststore** file to execution nodes. Example:

```
./bin/flink run -yt ssl/ -ys 3 -m yarn-cluster -c  
org.apache.flink.examples.java.wordcount.WordCount /opt/client/Flink/flink/examples/batch/  
WordCount.jar
```

 **NOTE**

- In the preceding example, **ssl/** is the sub-directory of the Flink client directory. It is used to store configuration files of the SSL keystore and truststore.
- The relative path of **ssl/** must be accessible from the current path where the Flink client command is run.

4. If the keystore or truststore file path is an absolute path, the keystore and truststore files must exist in the absolute path on Flink Client and all nodes.

 **NOTE**

For versions earlier than MRS 3.x, the user who submits the job must have the permission to read the keystore and truststore files.

Either of the following methods can be used to execute applications. The **-t** or **-yt** option does not need to be added to transmit the **keystore** and **truststore** files.

- Run the **CLI `yarn-session.sh`** command of Flink to execute applications.
Example:

```
./bin/yarn-session.sh
```
- Run the **Flink `run`** command to execute applications. Example:

```
./bin/flink run -ys 3 -m yarn-cluster -c  
org.apache.flink.examples.java.wordcount.WordCount /opt/client/Flink/flink/examples/batch/  
WordCount.jar
```

6.5.2 ACL Control

In HA mode of Flink, ZooKeeper can be used to manage clusters and discover services. Zookeeper supports SASL ACL control. Only users who have passed the SASL (Kerberos) authentication have the permission to operate files on ZooKeeper. To enable SASL ACL control, perform following configurations in the Flink configuration file.

```
high-availability.zookeeper.client.acl: creator  
zookeeper.sasl.disable: false
```

For details about configuration items, see [Table 6-14](#).

6.5.3 Web Security

Coding Specifications

Note: The same coding mode is used on the web service client and server to prevent garbled characters and to enable input verification.

Security hardening: apply UTF-8 to response messages of web server.

Whitelist-based Filter of IP Addresses

Note: IP filter must be added to the web server to filter unauthorized requests from the source IP address and prevent unauthorized login.

Security: Add **`jobmanager.web.allow-access-address`** to enable the IP filter. By default, only Yarn users are supported.

NOTE

After the client is installed, you need to add the IP address of the client node to the **`jobmanager.web.allow-access-address`** configuration item.

Preventing Sending the Absolute Paths to the Client

Note: If an absolute path is sent to a client, the directory structure of the server is exposed, increasing the risk that attackers know and attack the system.

Security hardening: If the Flink configuration file contains a parameter starting with a slash (/), the first-level directory is deleted.

Same-origin Policy

The same-source policy applies to MRS 3.x or later.

If two URL protocols have same hosts and ports, they are of the same origin. Protocols of different origins cannot access each other, unless the source of the visitor is specified on the host of the service to be visited.

Security hardening: The default value of the header of the response header **Access-Control-Allow-Origin** is the IP address of ResourceManager on Yarn clusters. If the IP address is not from Yarn, mutual access is not allowed.

Defense Against XSS

Defense Against XSS applies to MRS 3.x or later.

Disabling XSS filter incurs the following risks:

- Leakage of sensitive information, for example, **sessionid**.
- Tampering of pages.
- Redirection to malicious websites.
- DoS attacks from other websites to Flink web.

Security hardening: Add the **X-XSS-Protection** security header to enable XSS filter. The default value configuration is **X-XSS-Protection: 1; mode=block**. If XSS attack is detected, rendering of the page is stopped. For example, if XSS attacks are detected in Internet Explorer 8, all contents of the page is replaced by #.

Preventing Sensitive Information Disclosure

Sensitive information disclosure prevention is applicable to MRS 3.x or later.

Web pages containing sensitive data must not be cached, to avoid leakage of sensitive information or data crosstalk among users who visit the internet through the proxy server.

Security hardening: Add **Cache-control**, **Pragma**, **Expires** security header. The default value is **Cache-Control: no-store**, **Pragma: no-cache**, and **Expires: 0**.

The security hardening stops contents interacted between Flink and web server from being cached.

Anti-Hijacking

Anti-hijacking applies to MRS 3.x or later.

Since hotlinking and clickjacking use framing technologies, security hardening is required to prevent attacks.

Security hardening: Add **X-Frame-Options** security header to specify whether the browser will load the pages from **iframe**, **frame** or **object**. The default value is **X-Frame-Options: DENY**, indicating that no pages can be nested to **iframe**, **frame** or **object**.

Logging calls of the Web Service APIs

This function applies to MRS 3.x or later.

Calls of the **Flink webmonitor restful** APIs are logged.

The **jobmanager.web.accesslog.enable** can be added in the **access log**. The default value is **true**. Logs are stored in a separate **webaccess.log** file.

Cross-Site Request Forgery Prevention

Cross-site request forgery (CSRF) prevention applies to MRS 3.x or later.

In **Browser/Server** applications, CSRF must be prevented for operations involving server data modification, such as adding, modifying, and deleting. The CSRF forces end users to execute non-intended operations on the current web application.

Security hardening: Only two post APIs, one delete API, and get interfaces are reserve for modification requests. All other APIs are deleted.

Troubleshooting

This function applies to MRS 3.x or later.

When the application is abnormal, exception information is filtered, logged, and returned to the client.

Security hardening

- A default error message page to filter information and log detailed error information.
- Four configuration parameters are added to ensure that the error page is switched to a specified URL provided by FusionInsight, preventing exposure of unnecessary information.

Table 6-25 Parameter description

Parameter	Description	Default Value	Mandatory
jobmanager.web.403-redirect-url	Web page access error 403. If 403 error occurs, the page switch to a specified page.	-	Yes
jobmanager.web.404-redirect-url	Web page access error 404. If 404 error occurs, the page switch to a specified page.	-	Yes
jobmanager.web.415-redirect-url	Web page access error 415. If 415 error occurs, the page switch to a specified page.	-	Yes

Parameter	Description	Default Value	Mandatory
jobmanager.web.500-redirect-url	Web page access error 500. If 500 error occurs, the page switch to a specified page.	-	Yes

HTML5 Security

HTML5 security applies to MRS 3.x or later.

HTML5 is a next generation web development specification that provides new functions and extend the labels for developers. These new labels add the attack surfaces and may incur risks of attacks. For example, cross-domain resource sharing, storage on the client, WebWorker, WebRTC, and WebSocket.

Security hardening: Add the **Access-Control-Allow-Origin** parameter. For example, if you want to enable the cross-domain resource sharing, configure the **Access-Control-Allow-Origin** parameter of the HTTP response header.

NOTE

Flink does not involve security risks of functions such as storage on the client, WebWorker, WebRTC, and WebSocket.

6.6 Security Statement

- All security functions of Flink are provided by the open source community or self-developed. Security features that need to be configured by users, such as authentication and SSL encrypted transmission, may affect performance.
- As a big data computing and analysis platform, Flink does not detect sensitive information. Therefore, you need to ensure that the input data is not sensitive.
- You can evaluate whether configurations are secure as required.
- For any security-related problems, contact O&M support.

6.7 Flink Log Overview

Log Description

Log path:

- Run logs of a Flink job: `${BIGDATA_DATA_HOME}/hadoop/data${i}/nm/containerlogs/application_${appid}/container_${$contid}`

NOTE

The logs of executing tasks are stored in the preceding path. After the execution is complete, the Yarn configuration determines whether these logs are gathered to the HDFS directory.

- FlinkResource run logs: `/var/log/Bigdata/flink/flinkResource`

Log archive rules:

1. FlinkResource run logs:

- By default, service logs are backed up each time when the log size reaches 20 MB. A maximum of 20 logs can be reserved without being compressed.

 **NOTE**

For versions earlier than MRS 3.x, The executor logs are backed up each time when the log size reaches 30 MB. A maximum of 20 logs can be reserved without being compressed.

- You can set the log size and number of compressed logs on the Manager page or modify the corresponding configuration items in **log4j-cli.properties**, **log4j.properties**, and **log4j-session.properties** in **/opt/client/Flink/flink/conf/** on the client. **/opt/client** is the client installation directory.

Table 6-26 FlinkResource log list

Type	Name	Description
FlinkResource run logs	checkService.log	Health check log
	kinit.log	Initialization log
	postinstall.log	Service installation log
	prestart.log	Prestart script log
	start.log	Startup log

Log Level

Table 6-27 describes the log levels supported by Flink. The priorities of log levels are ERROR, WARN, INFO, and DEBUG in descending order. Logs whose levels are higher than or equal to the specified level are printed. The number of printed logs decreases as the specified log level increases.

Table 6-27 Log levels

Level	Description
ERROR	Error information about the current event processing
WARN	Exception information about the current event processing
INFO	Normal running status information about the system and events
DEBUG	System information and system debugging information

To modify log levels, perform the following steps:

- Step 1** Go to the **All Configurations** page of Flink by referring to **Modifying Cluster Service Configuration Parameters**.
- Step 2** On the menu bar on the left, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.

----End

 **NOTE**

- After the configuration is complete, you do not need to restart the service. Download the client again for the configuration to take effect.
- You can also change the configuration items corresponding to the log level in **log4j-cli.properties**, **log4j.properties**, and **log4j-session.properties** in **/opt/client/Flink/flink/conf/** on the client. **/opt/client** is the client installation directory.
- When a job is submitted using a client, a log file is generated in the **log** folder on the client. The default umask value is **0022**. Therefore, the default log permission is **644**. To change the file permission, you need to change the umask value. For example, to change the umask value of user **omm**:
 - Add **umask 0026** to the end of the **/home/omm/.baskrc** file.
 - Run the **source /home/omm/.baskrc** command to make the file permission take effect.

Log Format

Table 6-28 Log formats

Type	Format	Example
Run log	<i><yyyy-MM-dd HH:mm:ss,SSS> <Log level> <Name of the thread that generates the log> <Message in the log> <Location where the log event occurs></i>	2019-06-27 21:30:31,778 INFO [flink- akka.actor.default- dispatcher-3] TaskManager container_e10_14982906 98388_0004_02_000007 has started. org.apache.flink.yarn.Yar nFlinkResourceManager (FlinkResourceManag- er.java:368)

6.8 Flink Performance Tuning

6.8.1 Optimization DataStream

6.8.1.1 Memory Configuration Optimization

Scenarios

The computing of Flink depends on memory. If the memory is insufficient, the performance of Flink will be greatly deteriorated. One solution is to monitor garbage collection (GC) to evaluate the memory usage. If the memory becomes the performance bottleneck, optimize the memory usage according to the actual situation.

If **Full GC** is frequently reported in the Container GC on the Yarn that monitors the node processes, the GC needs to be optimized.

NOTE

In the `env.java.opts` configuration item of the `conf/flink-conf.yaml` file on the client, add the `-Xloggc:<LOG_DIR>/gc.log -XX:+PrintGCDetails -XX:-OmitStackTraceInFastThrow -XX:+PrintGCTimeStamps -XX:+PrintGCDateStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=20 -XX:GCLogFileSize=20M` parameter. The GC log is configured by default.

Procedure

- Optimize GC.
Adjust the ratio of tenured generation memory to young generation memory. In the `conf/flink-conf.yaml` configuration file on the client, add the `-XX:NewRatio` parameter to the `env.java.opts` configuration item. For example, `-XX:NewRatio=2` indicates that ratio of tenured generation memory to young generation memory is 2:1, that is, the young generation memory occupies one third and tenured generation memory occupies two thirds.
- When developing Flink applications, optimize the partitioning or grouping operation of `DataStream`.
 - If partitioning causes data skew, partitions need to be optimized.
 - Do not perform concurrent operations, because some operations, `WindowAll` for example, to `DataStream` do not support parallelism.
 - Do not use `set keyBy` to string type.

6.8.1.2 Configuring DOP

Scenario

The degree of parallelism (DOP) indicates the number of tasks to be executed concurrently. It determines the number of data blocks after the operation. Configuring the DOP will optimize the number of tasks, data volume of each task, and the host processing capability.

Query the CPU and memory usage. If data and tasks are not evenly distributed among nodes, increase the DOP for even distribution.

Procedure

Configure the DOP at one of the following layers (the priorities of which are in the descending order) based on the actual memory, CPU, data, and application logic conditions:

- Operator

Call the **setParallelism()** method to specify the DOP of an operator, data source, and sink. For example:

```
final StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

DataStream<String> text = [...]
DataStream<Tuple2<String, Integer>> wordCounts = text
    .flatMap(new LineSplitter())
    .keyBy(0)
    .timeWindow(Time.seconds(5))
    .sum(1).setParallelism(5);

wordCounts.print();

env.execute("Word Count Example");
```

- Execution environment

Flink runs in the execution environment which defines a default DOP for operators, data source and data sink.

Call the **setParallelism()** method to specify the default DOP of the execution environment. Example:

```
final StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();
env.setParallelism(3);
DataStream<String> text = [...]
DataStream<Tuple2<String, Integer>> wordCounts = [...]
wordCounts.print();
env.execute("Word Count Example");
```

- Client

Specify the DOP when submitting jobs to Flink on the client. If you use the CLI client, specify the DOP using the **-p** parameter. Example:

```
./bin/flink run -p 10 ../examples/*WordCount-java*.jar
```

- System

On the Flink client, modify the **parallelism.default** parameter in the **flink-conf.yaml** file under the conf to specify the DOP for all execution environments.

6.8.1.3 Configuring Process Parameters

Scenario

In Flink on Yarn mode, there are JobManagers and TaskManagers. JobManagers and TaskManagers schedule and run tasks.

Therefore, configuring parameters of JobManagers and TaskManagers can optimize the execution performance of a Flink application. Perform the following steps to optimize the Flink cluster performance.

Procedure

- Step 1** Configure JobManager memory.

JobManagers are responsible for task scheduling and message communications between TaskManagers and ResourceManagers. JobManager memory needs to be increased as the number of tasks and the DOP increases.

JobManager memory needs to be configured based on the number of tasks.

- When running the **yarn-session** command, add the **-jm MEM** parameter to configure the memory.
- When running the **yarn-cluster** command, add the **-yjm MEM** parameter to configure the memory.

Step 2 Configure the number of TaskManagers.

Each core of a TaskManager can run a task at the same time. Increasing the number of TaskManagers has the same effect as increasing the DOP. Therefore, you can increase the number of TaskManagers to improve efficiency when there are sufficient resources.

Step 3 Configure the number of TaskManager slots.

Multiple cores of a TaskManager can process multiple tasks at the same time. This has the same effect as increasing the DOP. However, the balance between the number of cores and the memory must be maintained, because all cores of a TaskManager share the memory.

- When running the **yarn-session** command, add the **-s NUM** parameter to configure the number of slots.
- When running the **yarn-cluster** command, add the **-ys NUM** parameter to configure the number of slots.

Step 4 Configure TaskManager memory.

TaskManager memory is used for task execution and communication. A large-size task requires more resources. In this case, you can increase the memory.

- When running the **yarn-session** command, add the **-tm MEM** parameter to configure the memory.
- When running the **yarn-cluster** command, add the **-ytm MEM** parameter to configure the memory.

----End

6.8.1.4 Optimizing the Design of Partitioning Method

Scenarios

The divide of tasks can be optimized by optimizing the partitioning method. If data skew occurs in a certain task, the whole execution process is delayed. Therefore, when designing the partitioning method, ensure that partitions are evenly assigned.

Procedure

Partitioning methods are as follows:

- **Random partitioning:** randomly partitions data.

```
dataStream.shuffle();
```

- **Rebalancing (round-robin partitioning):** evenly partitions data based on round-robin. The partitioning method is useful to optimize data with data skew.
dataStream.rebalance();
- **Rescaling:** assign data to downstream subsets in the form of round-robin. The partitioning method is useful if you want to deliver data from each parallel instance of a data source to subsets of some mappers without the using rebalance (), that is, the complete rebalance operation.
dataStream.rescale();
- **Broadcast:** broadcast data to all partitions.
dataStream.broadcast();
- **User-defined partitioning:** use a user-defined partitioner to select a target task for each element. The user-defined partitioning allows user to partition data based on a certain feature to achieve optimized task execution.

The following is an example:

```
// fromElements builds simple Tuple2 stream
DataStream<Tuple2<String, Integer>> dataStream = env.fromElements(Tuple2.of("hello",1),
Tuple2.of("test",2), Tuple2.of("world",100));

// Defines the key value used for partitioning. Adding one to the value equals to the id.
Partitioner<Tuple2<String, Integer>> strPartitioner = new Partitioner<Tuple2<String, Integer>>() {
    @Override
    public int partition(Tuple2<String, Integer> key, int numPartitions) {
        return (key.f0.length() + key.f1) % numPartitions;
    }
};

// The Tuple2 data is used as the basis for partitioning.

dataStream.partitionCustom(strPartitioner, new KeySelector<Tuple2<String, Integer>, Tuple2<String,
Integer>>() {
    @Override
    public Tuple2<String, Integer> getKey(Tuple2<String, Integer> value) throws Exception {
        return value;
    }
}).print();
```

6.8.1.5 Configuring the Netty Network Communication

Scenarios

The communication of Flink is based on Netty network. The network performance determines the data switching speed and task execution efficiency. Therefore, the performance of Flink can be optimized by optimizing the Netty network.

Procedure

In the **conf/flink-conf.yaml** file on the client, change configurations as required. Exercise caution when changing default values, because default values are optimal.

- **taskmanager.network.netty.num-arenas:** Specifies the number of arenas of Netty. The default value is **taskmanager.numberOfTaskSlots**.
- **taskmanager.network.netty.server.numThreads** and **taskmanager.network.netty.client.numThreads:** specify the number of

threads on the client and server. The default value is **taskmanager.numberOfTaskSlots**.

- **taskmanager.network.netty.client.connectTimeoutSec**: specifies the timeout interval for connection of TaskManager client. The default value is **120s**.
- **taskmanager.network.netty.sendReceiveBufferSize**: specifies the buffer size of the Netty network. The default value is the buffer size (cat /proc/sys/net/ipv4/tcp_[rw]mem) of the system and the value is usually 4 MB.
- **taskmanager.network.netty.transport**: specifies the transmission method of the Netty network. The default value is **nio**. The value can only be **nio** and **epoll**.

6.8.1.6 Summarization

Avoiding Data Skew

If data skew occurs (certain data volume is large), the execution time of tasks is inconsistent even if no garbage collection is performed.

- Redefine keys. Use keys of smaller granularity to optimize the task size.
- Modify the DOP.
- Call the rebalance operation to balance data partitions.

Setting Timeout Interval for the Buffer

- During the execution of tasks, data is switched through network switching. You can configure the **setBufferTimeout** parameter to specify the timeout interval for the buffer.
- If **setBufferTimeout** is set to **-1**, the refreshing operation is performed when the buffer full, maximizing the throughput. If **setBufferTimeout** is set to **0**, the refreshing operation is performed each time data is received, minimizing the delay. If **setBufferTimeout** is set to a value greater than **0**, the refreshing operation is performed after the buffer times out.

The following is an example:

```
env.setBufferTimeout(timeoutMillis);  
  
env.generateSequence(1,10).map(new MyMapper()).setBufferTimeout(timeoutMillis);
```

6.9 Common Flink Shell Commands

This section applies to MRS 3.x or later.

Before running the Flink shell commands, perform the following steps:

Step 1 Install the Flink client in a directory, for example, **/opt/client**.

Step 2 Run the following command to initialize environment variables:

```
source /opt/client/bigdata_env
```

Step 3 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled, skip this step.

kinit *Service user*

Step 4 Run the related commands according to [Table 6-29](#).

Table 6-29 Flink Shell commands

Command	Description	Description
yarn-session.sh	<p>-at,--applicationType <arg>: Defines the Yarn application type.</p> <p>-D <property=value>: Configures dynamic parameter.</p> <p>-d,--detached: Disables the interactive mode and starts a separate Flink Yarn session.</p> <p>-h,--help: Displays the help information about the Yarn session CLI.</p> <p>-id,--applicationId <arg>: Binds to a running Yarn session.</p> <p>-j,--jar <arg>: Sets the path of the user's JAR file.</p> <p>-jm,--jobManagerMemory <arg>: Sets the JobManager memory.</p> <p>-m,--jobmanager <arg>: Address of the JobManager (master) to which to connect. Use this parameter to connect to a specified JobManager.</p> <p>-nl,--nodeLabel <arg>: Specifies the nodeLabel of the Yarn application.</p> <p>-nm,--name <arg>: Customizes a name for the application on Yarn.</p> <p>-q,--query: Queries available Yarn resources.</p> <p>-qu,--queue <arg>: Specifies a Yarn queue.</p> <p>-s,--slots <arg>: Sets the number of slots for each TaskManager.</p> <p>-t,--ship <arg>: specifies the directory of the file to be sent.</p> <p>-tm,--taskManagerMemory <arg>: sets the TaskManager memory.</p> <p>-yd,--yarndetached: starts Yarn in the detached mode.</p> <p>-z,--zookeeperNamespace <args>: specifies the namespace of ZooKeeper.</p> <p>-h: Gets help information.</p>	Start a resident Flink cluster to receive tasks from the Flink client.

Command	Description	Description
flink run	<p>-c,--class <classname>: Specifies a class as the entry for running programs.</p> <p>-C,--classpath <url>: Specifies classpath.</p> <p>-d,--detached: Runs a job in the detached mode.</p> <p>-files,--dependencyFiles <arg>: File on which the Flink program depends.</p> <p>-n,--allowNonRestoredState: A state that cannot be restored can be skipped during restoration from a snapshot point in time. For example, if an operator in the program is deleted, you need to add this parameter when restoring the snapshot point.</p> <p>-m,--jobmanager <host:port>: Specifies the JobManager.</p> <p>-p,--parallelism <parallelism>: Specifies the job DOP, which will overwrite the DOP parameter in the configuration file.</p> <p>-q,--sysoutLogging: Disables the function of outputting Flink logs to the console.</p> <p>-s,--fromSavepoint <savepointPath>: Specifies a savepoint path for recovering jobs.</p> <p>-z,--zookeeperNamespace <zookeeperNamespace>: specifies the namespace of ZooKeeper.</p> <p>-yat,--yarnapplicationType <arg>: Defines the Yarn application type.</p> <p>-yD <arg>: Dynamic parameter configuration.</p> <p>-yd,--yarndetached: Starts Yarn in the detached mode.</p> <p>-yh,--yarnhelp: Obtains the Yarn help.</p> <p>-yid,--yarnapplicationId <arg>: Binds a job to a Yarn session.</p> <p>-yj,--yarnjar <arg>: Sets the path to Flink jar file.</p>	<p>Submit a Flink job.</p> <ol style="list-style-type: none"> 1. The -y* parameter is used in the yarn-cluster mode. 2. If the parameter is not -y*, you need to run the yarn-session command to start the Flink cluster before running this command to submit a task.

Command	Description	Description
	<p>-yjm,--yarnjobManagerMemory <arg>: Sets the JobManager memory (MB).</p> <p>-ynm,--yarnname <arg>: Customizes a name for the application on Yarn.</p> <p>-yq,--yarnquery: Queries available Yarn resources (memory and CPUs).</p> <p>-yqu,--yarnqueue <arg>: Specifies a Yarn queue.</p> <p>-ys,--yarnslots: Sets the number of slots for each TaskManager.</p> <p>-yt,--yarnship <arg>: Specifies the path of the file to be sent.</p> <p>-ytm,--yarntaskManagerMemory <arg>: Sets the TaskManager memory (MB).</p> <p>-yz,--yarnzookeeperNamespace <arg>: Specifies the namespace of ZooKeeper. The value must be the same as the value of yarn-session.sh -z.</p> <p>-h: Gets help information.</p>	
flink info	<p>-c,--class <classname>: Specifies a class as the entry for running programs.</p> <p>-p,--parallelism <parallelism>: Specifies the DOP for running programs.</p> <p>-h: Gets help information.</p>	Display the execution plan (JSON) of the running program.
flink list	<p>-a,--all: displays all jobs.</p> <p>-m,--jobmanager <host:port>: specifies the JobManager.</p> <p>-r,--running: displays only jobs in the running state.</p> <p>-s,--scheduled: displays only jobs in the scheduled state.</p> <p>-z,--zookeeperNamespace <zookeeperNamespace>: specifies the namespace of ZooKeeper.</p> <p>-yid,--yarnapplicationId <arg>: binds a job to a Yarn session.</p> <p>-h: gets help information.</p>	Query running programs in the cluster.

Command	Description	Description
flink stop	<p>-d,--drain: sends MAX_WATERMARK before the savepoint is triggered and the job is stopped.</p> <p>-p,--savepointPath <savepointPath>: path for storing savepoints. The default value is state.savepoints.dir.</p> <p>-m,--jobmanager <host:port>: specifies the JobManager.</p> <p>-z,--zookeeperNamespace <zookeeperNamespace>: specifies the namespace of ZooKeeper.</p> <p>-yid,--yarnapplicationId <arg>: binds a job to a Yarn session.</p> <p>-h: gets help information.</p>	<p>Forcibly stop a running job (only streaming jobs are supported).</p> <p>StoppableFunction needs to be implemented on the source side in service code).</p>
flink cancel	<p>-m,--jobmanager <host:port>: specifies the JobManager.</p> <p>-s,--withSavepoint <targetDirectory>: triggers a savepoint when a job is canceled. The default directory is state.savepoints.dir.</p> <p>-z,--zookeeperNamespace <zookeeperNamespace>: specifies the namespace of ZooKeeper.</p> <p>-yid,--yarnapplicationId <arg>: binds a job to a Yarn session.</p> <p>-h: gets help information.</p>	<p>Cancel a running job.</p>
flink savepoint	<p>-d,--dispose <arg>: specifies a directory for storing the savepoint.</p> <p>-m,--jobmanager <host:port>: specifies the JobManager.</p> <p>-z,--zookeeperNamespace <zookeeperNamespace>: specifies the namespace of ZooKeeper.</p> <p>-yid,--yarnapplicationId <arg>: binds a job to a Yarn session.</p> <p>-h: gets help information.</p>	<p>Trigger a savepoint.</p>

Command	Description	Description
<code>source Client installation directory/bigdata_env</code>	None	<p>Import client environment variables.</p> <p>Restriction: If the user uses a custom script (for example, A.sh) and runs this command in the script, variables cannot be imported to the A.sh script. If variables need to be imported to the custom script A.sh, the user needs to use the secondary calling method.</p> <p>For example, first call the B.sh script in the A.sh script, and then run this command in the B.sh script. Parameters can be imported to the A.sh script but cannot be imported to the B.sh script.</p>
<code>start-scala-shell.sh</code>	local remote <host> <port> yarn: running mode	Start the scala shell.
<code>sh generate_keystore.sh</code>	-	Run the generate_keystore.sh script to generate security cookie, flink.keystore , and flink.truststore . You need to enter a user-defined password that does not contain number signs (#).

----End

7 Using Flume

7.1 Using Flume from Scratch

Scenario

You can use Flume to import collected log information to Kafka.

Prerequisites

- A streaming cluster with Kerberos authentication enabled has been created.
- The Flume client has been installed on the node where logs are generated, for example, `/opt/Flumeclient`. For details about how to install the Flume client, see [Installing the Flume Client](#). The client directory in the following operations is only an example. Change it to the actual installation directory.
- The streaming cluster can properly communicate with the node where logs are generated.

Using the Flume Client (Versions Earlier Than MRS 3.x)

NOTE

You do not need to perform [Step 2](#) to [Step 6](#) for a normal cluster.

Step 1 Install the client.

For details, see [Installing the Flume Client](#).

Step 2 Copy the configuration file of the authentication server from the Master1 node to the *Flume client installation directory/fusioninsight-flume-Flume component version number/conf* directory on the node where the Flume client resides.

The full file path is `${BIGDATA_HOME}/MRS_Current/1_X_KerberosClient/etc/kdc.conf`.

In the preceding paths, **X** indicates a random number. Change it based on the site requirements. The file must be saved by the user who installs the Flume client, for example, user **root**.

Step 3 Check the service IP address of any node where the Flume role is deployed.

Log in to the cluster details page, choose *Name of the desired cluster* > **Components** > **Flume** > **Instances**, and check the service IP address of any node where the Flume role is deployed.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

Step 4 Copy the user authentication file from this node to the *Flume client installation directory/fusioninsight-flume-Flume component version number/conf* directory on the Flume client node.

The full file path is `${BIGDATA_HOME}/MRS_XXX/install/FusionInsight-Flume-Flume component version number/flume/conf/flume.keytab`.

In the preceding paths, **XXX** indicates the product version number. Change it based on the site requirements. The file must be saved by the user who installs the Flume client, for example, user **root**.

Step 5 Copy the **jaas.conf** file from this node to the **conf** directory on the Flume client node.

The full file path is `${BIGDATA_HOME}/MRS_Current/1_X_Flume/etc/jaas.conf`.

In the preceding path, **X** indicates a random number. Change it based on the site requirements. The file must be saved by the user who installs the Flume client, for example, user **root**.

Step 6 Log in to the Flume client node and go to the client installation directory. Run the following command to modify the file:

```
vi conf/jaas.conf
```

Change the full path of the user authentication file defined by **keyTab** to the **Flume client installation directory/fusioninsight-flume-Flume component version number/conf** saved in [Step 4](#), and save the modification and exit.

Step 7 Run the following command to modify the **flume-env.sh** configuration file of the Flume client:

```
vi Flume client installation directory/fusioninsight-flume-Flume component version number/conf/flume-env.sh
```

Add the following information after **-XX:+UseCMSCompactAtFullCollection**:

```
-Djava.security.krb5.conf=Flume client installation directory/fusioninsight-flume-1.9.0/conf/kdc.conf -  
Djava.security.auth.login.config=Flume client installation directory/fusioninsight-flume-1.9.0/conf/jaas.conf -  
Dzookeeper.request.timeout=120000
```

For example, "**-XX:+UseCMSCompactAtFullCollection -
Djava.security.krb5.conf=Flume client installation directory/fusioninsight-flume-Flume component version number/conf/kdc.conf -
Djava.security.auth.login.config=Flume client installation directory/fusioninsight-flume-Flume component version number/conf/jaas.conf -
Dzookeeper.request.timeout=120000**"

Change *Flume client installation directory* to the actual installation directory. Then save and exit.

Step 8 Assume that the Flume client installation path is **/opt/FlumeClient**. Run the following command to restart the Flume client:

```
cd /opt/FlumeClient/fusioninsight-flume-Flume component version number/bin  
./flume-manage.sh restart
```

Step 9 Run the following command to modify the **properties.properties** configuration file of the Flume client:

```
vi Flume client installation directory/fusioninsight-flume-Flume component version number/conf/properties.properties
```

Add the following information to the file:

```
#####  
#####  
client.sources = static_log_source  
client.channels = static_log_channel  
client.sinks = kafka_sink  
#####  
#####  
#LOG_TO_HDFS_ONLINE_1  
  
client.sources.static_log_source.type = spoolDir  
client.sources.static_log_source.spoolDir = PATH  
client.sources.static_log_source.fileSuffix = .COMPLETED  
client.sources.static_log_source.ignorePattern = ^$  
client.sources.static_log_source.trackerDir = PATH  
client.sources.static_log_source.maxBlobLength = 16384  
client.sources.static_log_source.batchSize = 51200  
client.sources.static_log_source.inputCharset = UTF-8  
client.sources.static_log_source.deserializer = LINE  
client.sources.static_log_source.selector.type = replicating  
client.sources.static_log_source.fileHeaderKey = file  
client.sources.static_log_source.fileHeader = false  
client.sources.static_log_source.basenameHeader = true  
client.sources.static_log_source.basenameHeaderKey = basename  
client.sources.static_log_source.deletePolicy = never  
  
client.channels.static_log_channel.type = file  
client.channels.static_log_channel.dataDirs = PATH  
client.channels.static_log_channel.checkpointDir = PATH  
client.channels.static_log_channel.maxFileSize = 2146435071  
client.channels.static_log_channel.capacity = 1000000  
client.channels.static_log_channel.transactionCapacity = 612000  
client.channels.static_log_channel.minimumRequiredSpace = 524288000  
  
client.sinks.kafka_sink.type = org.apache.flume.sink.kafka.KafkaSink  
client.sinks.kafka_sink.kafka.topic = flume_test  
client.sinks.kafka_sink.kafka.bootstrap.servers = XXX.XXX.XXX.XXX:210079092,XXX.XXX.XXX.XXX:  
21007,XXX.XXX.XXX.XXX:21007  
client.sinks.kafka_sink.flumeBatchSize = 1000  
client.sinks.kafka_sink.kafka.producer.type = sync  
client.sinks.kafka_sink.kafka.security.protocol = SASL_PLAINTEXT  
client.sinks.kafka_sink.kafka.kerberos.domain.name = hadoop.XXX.com  
client.sinks.kafka_sink.requiredAcks = 0  
  
client.sources.static_log_source.channels = static_log_channel  
client.sinks.kafka_sink.channel = static_log_channel
```

Modify the following parameters as required. Then save and exit the file.

- spoolDir

- trackerDir
- dataDirs
- checkpointDir
- topic
If the topic does not exist in Kafka, the topic is automatically created by default.
- kafka.bootstrap.servers
By default, the port for a security cluster is port 21007 and that for a normal cluster is port 9092.
- kafka.security.protocol
Set this parameter to **SASL_PLAINTEXT** for a security cluster and **PLAINTEXT** for a normal cluster.
- **kafka.kerberos.domain.name**
You do not need to set this parameter for a normal cluster. For a security cluster, the value of this parameter is the value of **kerberos.domain.name** in the Kafka cluster.
You can check **`\${BIGDATA_HOME}/MRS_Current/1_X_Broker/etc/server.properties** on the node where the broker instance resides.
In the preceding paths, **X** indicates a random number. Change it based on the site requirements. The file must be saved by the user who installs the Flume client, for example, user **root**.

Step 10 The Flume client automatically loads the information in the **properties.properties** file.

After new log files are generated in the directory specified by **spoolDir**, the logs will be sent to Kafka producers and can be consumed by Kafka consumers.

----End

Using the Flume Client (MRS 3.x or Later)

NOTE

You do not need to perform [Step 2](#) to [Step 6](#) for a normal cluster.

Step 1 Install the client.

For details, see [Installing the Flume Client](#).

Step 2 Copy the configuration file of the authentication server from the Master1 node to the *Flume client installation directory*/**fusioninsight-flume-Flume component version number/conf** directory on the node where the Flume client resides.

The full file path is **`\${BIGDATA_HOME}/FusionInsight_Current/1_X_KerberosClient/etc/kdc.conf**. In the preceding path, **X** indicates a random number. Change it based on the site requirements. The file must be saved by the user who installs the Flume client, for example, user **root**.

Step 3 Check the service IP address of any node where the Flume role is deployed.

Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster > Services > Flume > Instance**. Check the service IP address of any node where the Flume role is deployed.

 NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

Step 4 Copy the user authentication file from this node to the *Flume client installation directory/fusioninsight-flume-Flume component version number/conf* directory on the Flume client node.

The full file path is `${BIGDATA_HOME}/FusionInsight_Porter_XXX/install/FusionInsight-Flume-Flume component version number/flume/conf/flume.keytab`.

In the preceding paths, **XXX** indicates the product version number. Change it based on the site requirements. The file must be saved by the user who installs the Flume client, for example, user **root**.

Step 5 Copy the **jaas.conf** file from this node to the **conf** directory on the Flume client node.

The full file path is `${BIGDATA_HOME}/FusionInsight_Current/1_X_Flume/etc/jaas.conf`.

In the preceding path, **X** indicates a random number. Change it based on the site requirements. The file must be saved by the user who installs the Flume client, for example, user **root**.

Step 6 Log in to the Flume client node and go to the client installation directory. Run the following command to modify the file:

```
vi conf/jaas.conf
```

Change the full path of the user authentication file defined by **keyTab** to the **Flume client installation directory/fusioninsight-flume-Flume component version number/conf** saved in [Step 4](#), and save the modification and exit.

Step 7 Run the following command to modify the **flume-env.sh** configuration file of the Flume client:

```
vi Flume client installation directory/fusioninsight-flume-Flume component version number/conf/flume-env.sh
```

Add the following information after **-XX:+UseCMSCompactAtFullCollection**:

```
-Djava.security.krb5.conf=Flume client installation directory/fusioninsight-flume-1.9.0/conf/kdc.conf -  
Djava.security.auth.login.config=Flume client installation directory/fusioninsight-flume-1.9.0/conf/jaas.conf -  
Dzookeeper.request.timeout=120000
```

For example, **"-XX:+UseCMSCompactAtFullCollection -
Djava.security.krb5.conf=Flume client installation directory/fusioninsight-flume-Flume component version number/conf/kdc.conf -
Djava.security.auth.login.config=Flume client installation directory/fusioninsight-flume-Flume component version number/conf/jaas.conf -
Dzookeeper.request.timeout=120000"**

Change *Flume client installation directory* to the actual installation directory. Then save and exit.

Step 8 Assume that the Flume client installation path is **/opt/FlumeClient**. Run the following command to restart the Flume client:

```
cd /opt/FlumeClient/fusioninsight-flume-Flume component version number/bin
./flume-manage.sh restart
```

Step 9 Run the following command to modify the **properties.properties** configuration file of the Flume client:

```
vi Flume client installation directory/fusioninsight-flume-Flume component version number/conf/properties.properties
```

Add the following information to the file:

```
#####
#####
client.sources = static_log_source
client.channels = static_log_channel
client.sinks = kafka_sink
#####
#####
#LOG_TO_HDFS_ONLINE_1

client.sources.static_log_source.type = spoolDir
client.sources.static_log_source.spoolDir = PATH
client.sources.static_log_source.fileSuffix = .COMPLETED
client.sources.static_log_source.ignorePattern = ^$
client.sources.static_log_source.trackerDir = PATH
client.sources.static_log_source.maxBlobLength = 16384
client.sources.static_log_source.batchSize = 51200
client.sources.static_log_source.inputCharset = UTF-8
client.sources.static_log_source.deserializer = LINE
client.sources.static_log_source.selector.type = replicating
client.sources.static_log_source.fileHeaderKey = file
client.sources.static_log_source.fileHeader = false
client.sources.static_log_source.basenameHeader = true
client.sources.static_log_source.basenameHeaderKey = basename
client.sources.static_log_source.deletePolicy = never

client.channels.static_log_channel.type = file
client.channels.static_log_channel.dataDirs = PATH
client.channels.static_log_channel.checkpointDir = PATH
client.channels.static_log_channel.maxFileSize = 2146435071
client.channels.static_log_channel.capacity = 1000000
client.channels.static_log_channel.transactionCapacity = 612000
client.channels.static_log_channel.minimumRequiredSpace = 524288000

client.sinks.kafka_sink.type = org.apache.flume.sink.kafka.KafkaSink
client.sinks.kafka_sink.kafka.topic = flume_test
client.sinks.kafka_sink.kafka.bootstrap.servers = XXX.XXX.XXX.XXX:210079092,XXX.XXX.XXX.XXX:21007,XXX.XXX.XXX.XXX:21007
client.sinks.kafka_sink.flumeBatchSize = 1000
client.sinks.kafka_sink.kafka.producer.type = sync
client.sinks.kafka_sink.kafka.security.protocol = SASL_PLAINTEXT
client.sinks.kafka_sink.kafka.kerberos.domain.name = hadoop.XXX.com
client.sinks.kafka_sink.requiredAcks = 0

client.sources.static_log_source.channels = static_log_channel
client.sinks.kafka_sink.channel = static_log_channel
```

Modify the following parameters as required. Then save and exit the file.

- spoolDir

- trackerDir
- dataDirs
- checkpointDir
- topic
If the topic does not exist in Kafka, the topic is automatically created by default.
- kafka.bootstrap.servers
By default, the port for a security cluster is port 21007 and that for a normal cluster is port 9092.
- kafka.security.protocol
Set this parameter to **SASL_PLAINTEXT** for a security cluster and **PLAINTEXT** for a normal cluster.
- **kafka.kerberos.domain.name**
You do not need to set this parameter for a normal cluster. For a security cluster, the value of this parameter is the value of **kerberos.domain.name** in the Kafka cluster.
For details, check **\${BIGDATA_HOME}/FusionInsight_Current/1_X_Broker/etc/server.properties** on the node where the broker instance resides.
In the preceding paths, **X** indicates a random number. Change it based on the site requirements. The file must be saved by the user who installs the Flume client, for example, user **root**.

Step 10 The Flume client automatically loads the information in the **properties.properties** file.

After new log files are generated in the directory specified by **spoolDir**, the logs will be sent to Kafka producers and can be consumed by Kafka consumers.

----End

7.2 Overview

Flume is a distributed, reliable, and highly available system for aggregating massive logs, which can efficiently collect, aggregate, and move massive log data from different data sources and store the data in a centralized data storage system. Various data senders can be customized in the system to collect data. Additionally, Flume provides simple data processes capabilities and writes data to data receivers (which is customizable).

Flume consists of the client and server, both of which are FlumeAgents. The server corresponds to the FlumeServer instance and is directly deployed in a cluster. The client can be deployed inside or outside the cluster. The client-side and service-side FlumeAgents work independently and provide the same functions.

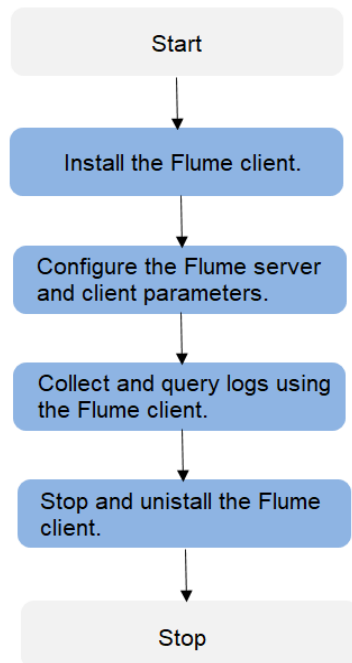
The client-side FlumeAgent needs to be independently installed. Data can be directly imported to components such as HDFS and Kafka. Additionally, the client-side and service-side FlumeAgents can also work together to provide services.

Process

The process for collecting logs using Flume is as follows:

1. Installing the flume client
2. Configuring the Flume server and client parameters
3. Collecting and querying logs using the Flume client
4. Stopping and uninstalling the Flume client

Figure 7-1 Log collection process



Flume Client

A Flume client consists of the source, channel, and sink. The source sends the data to the channel, and then the sink transmits the data from the channel to the external device. [Table 7-1](#) describes Flume modules.

Table 7-1 Module description

Name	Description
Source	<p>A source receives or generates data and sends the data to one or multiple channels. The source can work in either data-driven or polling mode.</p> <p>Typical sources include:</p> <ul style="list-style-type: none"> • Sources that are integrated with the system and receives data, such as Syslog and Netcat • Sources that automatically generate event data, such as Exec and SEQ • IPC sources that are used for communication between agents, such as Avro <p>A Source must associate with at least one channel.</p>
Channel	<p>A channel is used to buffer data between a source and a sink. After the sink transmits the data to the next channel or the destination, the cache is deleted automatically.</p> <p>The persistency of the channels varies with the channel types:</p> <ul style="list-style-type: none"> • Memory channel: non-persistency • File channel: persistency implemented based on write-ahead logging (WAL) • JDBC channel: persistency implemented based on the embedded database <p>Channels support the transaction feature to ensure simple sequential operations. A channel can work with sources and sinks of any quantity.</p>
Sink	<p>Sink is responsible for sending data to the next hop or final destination and removing the data from the channel after successfully sending the data.</p> <p>Typical sinks include:</p> <ul style="list-style-type: none"> • Sinks that send storage data to the final destination, such as HDFS and Kafka • Sinks that are consumed automatically, such as Null Sink • IPC sinks that are used for communication between agents, such as Avro <p>A sink must associate with at least one channel.</p>

A Flume client can have multiple sources, channels, and sinks. A source can send data to multiple channels, and then multiple sinks send the data out of the client.

Multiple Flume clients can be cascaded. That is, a sink can send data to the source of another client.

Supplementary Information

1. Flume provides the following reliability measures:
 - The transaction mechanism is implemented between sources and channels, and between channels and sinks.
 - The sink processor supports the failover and load balancing (load_balance) mechanisms.

The following is an example of the load balancing (load_balance) configuration:

```
server.sinkgroups=g1
server.sinkgroups.g1.sinks=k1 k2
server.sinkgroups.g1.processor.type=load_balance
server.sinkgroups.g1.processor.backoff=true
server.sinkgroups.g1.processor.selector=random
```

2. The following are precautions for the aggregation and cascading of multiple Flume clients:
 - Avro or Thrift protocol can be used for cascading.
 - When the aggregation end contains multiple nodes, evenly distribute the clients to these nodes. Do not connect all the clients to a single node.
3. The Flume client can contain multiple independent data flows. That is, multiple sources, channels, and sinks can be configured in the **properties.properties** configuration file. These components can be linked to form multiple flows.

For example, to configure two data flows in a configuration, run the following commands:

```
server.sources = source1 source2
server.sinks = sink1 sink2
server.channels = channel1 channel2

#dataflow1
server.sources.source1.channels = channel1
server.sinks.sink1.channel = channel1

#dataflow2
server.sources.source2.channels = channel2
server.sinks.sink2.channel = channel2
```

7.3 Installing the Flume Client

7.3.1 Installing the Flume Client on Clusters of Versions Earlier Than MRS 3.x

Scenario

To use Flume to collect logs, you must install the Flume client on a log host. You can create an ECS and install the Flume client on it.

This section applies to versions earlier than MRS 3.x.

Prerequisites

- A streaming cluster with the Flume component has been created.

- The log host is in the same VPC and subnet with the MRS cluster.
- You have obtained the username and password for logging in to the log host.

Procedure

Step 1 Create an ECS that meets the requirements.

Step 2 Go to the cluster details page and choose **Components**.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

Step 3 Click **Download Client**.

1. In **Client Type**, select **All client files**.
2. In **Download to**, select **Remote host**.
3. Set **Host IP Address** to the IP address of the ECS, **Host Port** to **22**, and **Save Path** to **/tmp**.

- If the default port **22** for logging in to an ECS through SSH has been changed, set **Host Port** to a new port.
- The value of **Save Path** contains a maximum of 256 characters.

4. Set **Login User** to **root**.

If another user is used, ensure that the user has permissions to read, write, and execute the save path.

5. Select **Password** or **SSH Private Key** for **Login Mode**.

- **Password**: Enter the password of user **root** set during cluster creation.
- **SSH Private Key**: Select and upload the key file used for creating the cluster.

6. Click **OK** to generate a client file.

If the following information is displayed, the client package is saved.

Client files downloaded to the remote host successfully.

If the following information is displayed, check the username, password, and security group configurations of the remote host. Ensure that the username and password are correct and an inbound rule of the SSH (22) port has been added to the security group of the remote host. And then, go to **Step 3** to download the client again.

Failed to connect to the server. Please check the network connection or parameter settings.

Step 4 Choose **Flume > Instance**. Query the **Business IP Address** of any Flume instance and any two MonitorServer instances.

Step 5 Log in to the ECS using VNC. See section "Login Using VNC" in the *Elastic Cloud Service User Guide* (**Instances > Logging In to a Linux ECS > Login Using VNC**).

All images support Cloud-Init. The preset username for Cloud-Init is **root** and the password is the one you set during cluster creation. You are advised to change the password upon the first login.

Step 6 On the ECS, switch to user **root** and copy the installation package to the **/opt** directory.

```
sudo su - root
```

```
cp /tmp/MRS_Flume_Client.tar /opt
```

Step 7 Run the following command in the **/opt** directory to decompress the package and obtain the verification file and the configuration package of the client:

```
tar -xvf MRS_Flume_Client.tar
```

Step 8 Run the following command to verify the configuration package of the client:

```
sha256sum -c MRS_Flume_ClientConfig.tar.sha256
```

If the following information is displayed, the file package is successfully verified:

```
MRS_Flume_ClientConfig.tar: OK
```

Step 9 Run the following command to decompress **MRS_Flume_ClientConfig.tar**:

```
tar -xvf MRS_Flume_ClientConfig.tar
```

Step 10 Run the following command to install the client running environment to a new directory, for example, **/opt/Flumeenv**. A directory is automatically generated during the client installation.

```
sh /opt/MRS_Flume_ClientConfig/install.sh /opt/Flumeenv
```

If the following information is displayed, the client running environment is successfully installed:

```
Components client installation is complete.
```

Step 11 Run the following command to configure environment variables:

```
source /opt/Flumeenv/bigdata_env
```

Step 12 Run the following commands to decompress the Flume client package:

```
cd /opt/MRS_Flume_ClientConfig/Flume
```

```
tar -xvf FusionInsight-Flume-1.6.0.tar.gz
```

Step 13 Run the following command to check whether the password of the current user has expired:

```
chage -l root
```

If the value of **Password expires** is earlier than the current time, the password has expired. Run the **chage -M -1 root** command to validate the password.

Step 14 Run the following command to install the Flume client to a new directory, for example, **/opt/FlumeClient**. A directory is automatically generated during the client installation.

```
sh /opt/MRS_Flume_ClientConfig/Flume/install.sh -d /opt/FlumeClient -f  
service IP address of the MonitorServer instance -c path of the Flume  
configuration file -l /var/log/ -e service IP address of Flume -n name of the Flume  
client
```

The parameters are described as follows:

- **-d**: indicates the installation path of the Flume client.
- (Optional) **-f**: indicates the service IP addresses of the two MonitorServer instances, separated by a comma (.). If the IP addresses are not configured, the Flume client will not send alarm information to MonitorServer, and the client information will not be displayed on MRS Manager.
- (Optional) **-c**: indicates the **properties.properties** configuration file that the Flume client loads after installation. If this parameter is not specified, the **fusioninsight-flume-1.6.0/conf/properties.properties** file in the client installation directory is used by default. The configuration file of the client is empty. You can modify the configuration file as required and the Flume client will load it automatically.
- (Optional) **-l**: indicates the log directory. The default value is **/var/log/Bigdata**.
- (Optional) **-e**: indicates the service IP address of the Flume instance. It is used to receive the monitoring indicators reported by the client.
- (Optional) **-n**: indicates the name of the Flume client.
- IBM JDK does not support **-Xloggc**. You must change **-Xloggc** to **-Xverbosegclog** in **flume/conf/flume-env.sh**. For 32-bit JDK, the value of **-Xmx** must not exceed 3.25 GB.
- In **flume/conf/flume-env.sh**, the default value of **-Xmx** is 4 GB. If the client memory is too small, you can change it to 512 MB or even 1 GB.

For example, run **sh install.sh -d /opt/FlumeClient**.

If the following information is displayed, the client is successfully installed:

```
install flume client successfully.
```

```
----End
```

7.3.2 Installing the Flume Client on Clusters of MRS 3.x or a Later Version

Scenario

To use Flume to collect logs, you must install the Flume client on a log host. You can create an ECS and install the Flume client on it.

This section applies to MRS 3.x and later versions.

Prerequisites

- A cluster with the Flume component has been created.
- The log host is in the same VPC and subnet with the MRS cluster.
- You have obtained the username and password for logging in to the log host.
- The installation directory is automatically created if it does not exist. If it exists, the directory must be left blank. The directory path cannot contain any space.

Procedure

Step 1 Obtain the software package.

Log in to the FusionInsight Manager. Choose **Cluster** > *Name of the target cluster* > **Services** > **Flume**. On the Flume service page that is displayed, choose **More** > **Download Client** in the upper right corner and set **Select Client Type** to **Complete Client** to download the Flume service client file.

The file name of the client is **FusionInsight_Cluster_<Cluster ID>_Flume_Client.tar**. This section takes the client file **FusionInsight_Cluster_1_Flume_Client.tar** as an example.

Step 2 Upload the software package.

Upload the software package to a directory, for example, **/opt/client** on the node where the Flume service client will be installed as user **user**.

NOTE

user is the user who installs and runs the Flume client.

Step 3 Decompress the software package.

Log in to the node where the Flume service client is to be installed as user **user**. Go to the directory where the installation package is installed, for example, **/opt/client**, and run the following command to decompress the installation package to the current directory:

```
cd /opt/client
```

```
tar -xvf FusionInsight_Cluster_1_Flume_Client.tar
```

Step 4 Verify the software package.

Run the **sha256sum -c** command to verify the decompressed file. If **OK** is returned, the verification is successful. Example:

```
sha256sum -c FusionInsight_Cluster_1_Flume_ClientConfig.tar.sha256
```

```
FusionInsight_Cluster_1_Flume_ClientConfig.tar: OK
```

Step 5 Decompress the package.

```
tar -xvf FusionInsight_Cluster_1_Flume_ClientConfig.tar
```

Step 6 Run the following command in the Flume client installation directory to install the client to a specified directory (for example, **opt/FlumeClient**): After the client is installed successfully, the installation is complete.

```
cd /opt/client/FusionInsight_Cluster_1_Flume_ClientConfig/Flume/FlumeClient
```

```
./install.sh -d /opt/FlumeClient -f MonitorServerService IP address or host name  
of the role -c User service configuration filePath for storing properties.properties -s  
CPU threshold -l /var/log/Bigdata -e FlumeServer service IP address or host name  
-n Flume
```


 NOTE

- **-d**: Flume client installation path
- (Optional) **-f**: IP addresses or host names of two MonitorServer roles. The IP addresses or host names are separated by commas (.). If this parameter is not configured, the Flume client does not send alarm information to MonitorServer and information about the client cannot be viewed on the FusionInsight Manager GUI.
- (Optional) **-c**: Service configuration file, which needs to be generated by the user based on the service. For details about how to generate the file on the configuration tool page of the Flume server, see [Flume Service Configuration Guide](#). Upload the file to any directory on the node where the client is to be installed. If this parameter is not specified during the installation, you can upload the generated service configuration file **properties.properties** to the **/opt/FlumeClient/fusioninsight-flume-1.9.0/conf** directory after the installation.
- (Optional) **-s**: cgroup threshold. The value is an integer ranging from 1 to 100 x *N*. *N* indicates the number of CPU cores. The default threshold is **-1**, indicating that the processes added to the cgroup are not restricted by the CPU usage.
- (Optional) **-l**: Log path. The default value is **/var/log/Bigdata**. The user **user** must have the write permission on the directory. When the client is installed for the first time, a subdirectory named **flume-client** is generated. After the installation, subdirectories named **flume-client-*n*** will be generated in sequence. The letter *n* indicates a sequence number, which starts from 1 in ascending order. In the **/conf/** directory of the Flume client installation directory, modify the **ENV_VARS** file and search for the **FLUME_LOG_DIR** attribute to view the client log path.
- (Optional) **-e**: Service IP address or host name of FlumeServer, which is used to receive statistics for the monitoring indicator reported by the client.
- (Optional) **-n**: Name of the Flume client. You can choose **Cluster > Name of the desired cluster > Service > Flume > Flume Management** on FusionInsight Manager to view the client name on the corresponding node.
- If the following error message is displayed, run the **export JAVA_HOME=*JDK path*** command.
JAVA_HOME is null in current user,please install the JDK and set the JAVA_HOME
- IBM JDK does not support **-Xloggc**. You must change **-Xloggc** to **-Xverbosegclog** in **flume/conf/flume-env.sh**. For 32-bit JDK, the value of **-Xmx** must not exceed 3.25 GB.
- When installing a cross-platform client in a cluster, go to the **/opt/client/FusionInsight_Cluster_1_Flume_ClientConfig/Flume/FusionInsight-Flume-1.9.0.tar.gz** directory to install the Flume client.

----End

7.4 Viewing Flume Client Logs

Scenario

You can view logs to locate faults.

Prerequisites

The Flume client has been installed.

Procedure

Step 1 Go to the Flume client log directory (**/var/log/Bigdata** by default).

Step 2 Run the following command to view the log file:

```
ls -lR flume-client-*
```

A log file is shown as follows:

```
flume-client-1/flume:
total 7672
-rw-----, 1 root root    0 Sep  8 19:43 Flume-audit.log
-rw-----, 1 root root 1562037 Sep 11 06:05 FlumeClient.2017-09-11_04-05-09.[1].log.zip
-rw-----, 1 root root 6127274 Sep 11 14:47 FlumeClient.log
-rw-----, 1 root root  2935 Sep  8 22:20 flume-root-20170908202009-pid72456-gc.log.0.current
-rw-----, 1 root root  2935 Sep  8 22:27 flume-root-20170908202634-pid78789-gc.log.0.current
-rw-----, 1 root root  4382 Sep  8 22:47 flume-root-20170908203137-pid84925-gc.log.0.current
-rw-----, 1 root root  4390 Sep  8 23:46 flume-root-20170908204918-pid103920-gc.log.0.current
-rw-----, 1 root root  3196 Sep  9 10:12 flume-root-20170908215351-pid44372-gc.log.0.current
-rw-----, 1 root root  2935 Sep  9 10:13 flume-root-20170909101233-pid55119-gc.log.0.current
-rw-----, 1 root root  6441 Sep  9 11:10 flume-root-20170909101631-pid59301-gc.log.0.current
-rw-----, 1 root root    0 Sep  9 11:10 flume-root-20170909111009-pid119477-gc.log.0.current
-rw-----, 1 root root 92896 Sep 11 13:24 flume-root-20170909111126-pid120689-gc.log.0.current
-rw-----, 1 root root  5588 Sep 11 14:46 flume-root-20170911132445-pid42259-gc.log.0.current
-rw-----, 1 root root  2576 Sep 11 13:24 prestartDetail.log
-rw-----, 1 root root  3303 Sep 11 13:24 startDetail.log
-rw-----, 1 root root  1253 Sep 11 13:24 stopDetail.log

flume-client-1/monitor:
total 8
-rw-----, 1 root root  141 Sep  8 19:43 flumeMonitorChecker.log
-rw-----, 1 root root 2946 Sep 11 13:24 flumeMonitor.log
```

In the log file, **FlumeClient.log** is the run log of the Flume client.

----End

7.5 Stopping or Uninstalling the Flume Client

Scenario

You can stop and start the Flume client or uninstall the Flume client when the Flume data ingestion channel is not required.

Procedure

- Stop the Flume client of the Flume role.
Assume that the Flume client installation path is **/opt/FlumeClient**. Run the following command to stop the Flume client:

```
cd /opt/FlumeClient/fusioninsight-flume-Flume component version
number/bin
```

```
./flume-manage.sh stop
```

If the following information is displayed after the command execution, the Flume client is successfully stopped.

```
Stop Flume PID=120689 successful..
```

 NOTE

The Flume client will be automatically restarted after being stopped. If you do not need automatic restart, run the following command:

```
./flume-manage.sh stop force
```

If you want to restart the Flume client, run the following command:

```
./flume-manage.sh start force
```

- Uninstall the Flume client of the Flume role.

Assume that the Flume client installation path is **/opt/FlumeClient**. Run the following command to uninstall the Flume client:

```
cd /opt/FlumeClient/fusioninsight-flume-Flume component version  
number/inst  
./uninstall.sh
```

7.6 Using the Encryption Tool of the Flume Client

Scenario

You can use the encryption tool provided by the Flume client to encrypt some parameter values in the configuration file.

Prerequisites

The Flume client has been installed.

Procedure

Step 1 Log in to the Flume client node and go to the client installation directory, for example, **/opt/FlumeClient**.

Step 2 Run the following command to switch the directory:

```
cd fusioninsight-flume-Flume component version number/bin
```

Step 3 Run the following command to encrypt information:

```
./genPwFile.sh
```

Input the information that you want to encrypt twice.

Step 4 Run the following command to query the encrypted information:

```
cat password.property
```

 NOTE

If the encryption parameter is used for the Flume server, you need to perform encryption on the corresponding Flume server node. You need to run the encryption script as user **omm** for encryption.

- For versions earlier than MRS 3.x, the encryption path is `/opt/Bigdata/MRS_XXX/install/FusionInsight-Flume-Flume component version number/flume/bin/genPwFile.sh`.
- For MRS 3.x or later, the encryption path is `/opt/Bigdata/FusionInsight_Porter_XXX/install/FusionInsight-Flume-Flume component version number/flume/bin/genPwFile.sh`. *XXX* indicates the product version number.

----End

7.7 Flume Service Configuration Guide

This section applies to MRS 3.x and later versions.

This configuration guide describes how to configure common Flume services.

 NOTE

- Parameters in bold in the following tables are mandatory.
- The value of **BatchSize** of the Sink must be less than that of **transactionCapacity** of the Channel.
- Only some parameters of Source, Channel, and Sink are displayed on the Flume configuration tool page. For details, see the following configurations.
- The Customer Source, Customer Channel, and Customer Sink displayed on the Flume configuration tool page need to be configured based on self-developed code. The following common configurations are not displayed.

Common Source Configurations

- **Avro Source**

An Avro source listens to the Avro port, receives data from the external Avro client, and places data into configured channels. Common configurations are as follows:

Table 7-2 Common configurations of an Avro source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured.
type	avro	Specifies the type of the avro source, which must be avro .
bind	-	Specifies the listening host name/IP address.

Parameter	Default Value	Description
port	-	Specifies the bound listening port. Ensure that this port is not occupied.
threads	-	Specifies the maximum number of source threads.
compression-type	none	Specifies the message compression format, which can be set to none or deflate . none indicates that data is not compressed, while deflate indicates that data is compressed.
compression-level	6	Specifies the data compression level, which ranges from 1 to 9 . The larger the value is, the higher the compression rate is.
ssl	false	Specifies whether to use SSL encryption. If this parameter is set to true , the values of keystore and keystore-password must be specified.
truststore-type	JKS	Specifies the Java trust store type, which can be set to JKS or PKCS12 . NOTE Different passwords are used to protect the key store and private key of JKS , while the same password is used to protect the key store and private key of PKCS12 .
truststore	-	Specifies the Java trust store file.
truststore-password	-	Specifies the Java trust store password.

Parameter	Default Value	Description
keystore-type	JKS	Specifies the keystore type set after SSL is enabled, which can be set to JKS or PKCS12 . NOTE Different passwords are used to protect the key store and private key of JKS , while the same password is used to protect the key store and private key of PKCS12 .
keystore	-	Specifies the keystore file path set after SSL is enabled. This parameter is mandatory if SSL is enabled.
keystore-password	-	Specifies the keystore password set after SSL is enabled. This parameter is mandatory if SSL is enabled.
trust-all-certs	false	Specifies whether to disable the check for the SSL server certificate. If this parameter is set to true , the SSL server certificate of the remote source is not checked. You are not advised to perform this operation during the production.
exclude-protocols	SSLv3	Specifies the excluded protocols. The entered protocols must be separated by spaces. The default value is SSLv3 .
ipFilter	false	Specifies whether to enable the IP address filtering.

Parameter	Default Value	Description
ipFilter.rules	-	Specifies the rules of <i>N</i> network ipFilters . Host names or IP addresses must be separated by commas (.). If this parameter is set to true , there are two configuration rules: allow and forbidden. The configuration format is as follows: ipFilterRules=allow:ip:127.*, allow:name:localhost, deny:ip:*

- **SpoolDir Source**

SpoolDir Source monitors and transmits new files that have been added to directories in real-time mode. Common configurations are as follows:

Table 7-3 Common configurations of a Spooling Directory source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured.
type	spooldir	Specifies the type of the spooling source, which must be set to spooldir .
spoolDir	-	Specifies the monitoring directory of the Spooldir source. A Flume running user must have the read, write, and execution permissions on the directory.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the source is restarted. Unit: second
fileSuffix	.COMPLETED	Specifies the suffix added after file transmission is complete.

Parameter	Default Value	Description
deletePolicy	never	Specifies the source file deletion policy after file transmission is complete. The value can be either never or immediate . never indicates that the source file is not deleted after file transmission is complete, while immediate indicates that the source file is immediately deleted after file transmission is complete.
ignorePattern	^\$	Specifies the regular expression of a file to be ignored. The default value is ^\$, indicating that spaces are ignored.
includePattern	^.*\$	Specifies the regular expression that contains a file. This parameter can be used together with ignorePattern . If a file meets both ignorePattern and includePattern , the file is ignored. In addition, when a file starts with a period (.), the file will not be filtered.
trackerDir	.flumespool	Specifies the metadata storage path during data transmission.
batchSize	1000	Specifies the number of events written to the channel in batches.
decodeErrorPolicy	FAIL	Specifies the code error policy. NOTE If a code error occurs in the file, set decodeErrorPolicy to REPLACE or IGNORE . Flume will skip the code error and continue to collect subsequent logs.
deserializer	LINE	Specifies the file parser. The value can be either LINE or BufferedLine . <ul style="list-style-type: none"> When the value is set to LINE, characters read from the file are transcoded one by one. When the value is set to BufferedLine, one line or multiple lines of characters read from the file are transcoded in batches, which delivers better performance.
deserializer.maxLineLength	2048	Specifies the maximum length for resolution by line.

Parameter	Default Value	Description
deserializer.maxBatchLine	1	Specifies the maximum number of lines for resolution by line. If multiple lines are set, maxLineLength must be set to a corresponding multiplier. NOTE When configuring the Interceptor, take the multi-line combination into consideration to avoid data loss. If the Interceptor cannot process combined lines, set this parameter to 1.
selector.type	replicating	Specifies the selector type. The value can be either replicating or multiplexing . replicating indicates that data is replicated and then transferred to each channel so that each channel receives the same data, while multiplexing indicates that a channel is selected based on the value of the header in the event and each channel has different data.
interceptors	-	Specifies the interceptor. Multiple interceptors are separated by spaces.
inputCharset	UTF-8	Specifies the encoding format of a read file. The encoding format must be the same as that of the data source file that has been read. Otherwise, an error may occur during character parsing.
fileHeader	false	Specifies whether to add the file name (including the file path) to the event header.
fileHeaderKey	-	Specifies that the data storage structure in header is set in the <key,value> mode. Parameters fileHeaderKey and fileHeader must be used together. Following is an example if fileHeader is set to true: Define fileHeaderKey as file . When the /root/a.txt file is read, fileHeaderKey exists in the header in the file=/root/a.txt format.
basenameHeader	false	Specifies whether to add the file name (excluding the file path) to the event header.

Parameter	Default Value	Description
basenameHeaderKey	-	Specifies that the data storage structure in header is set in the <key,value> mode. Parameters fileHeaderKey and fileHeader must be used together. Following is an example if fileHeader is set to true: Define fileHeaderKey as file . When the a.txt file is read, fileHeaderKey exists in the header in the file=a.txt format.
pollDelay	500	Specifies the delay for polling new files in the monitoring directory. Unit: milliseconds
recursiveDirectorySearch	false	Specifies whether to monitor new files in the subdirectory of the configured directory.
consumeOrder	oldest	Specifies the consumption order of files in a directory. If this parameter is set to oldest or youngest , the sequence of files to be read is determined by the last modification time of files in the monitored directory. If there are a large number of files in the directory, it takes a long time to search for oldest or youngest files. If this parameter is set to random , an earlier created file may not be read for a long time. If this parameter is set to oldest or youngest , it takes a long time to find the latest and the earliest file. The options are as follows: random , youngest , and oldest .
maxBackoff	4000	Specifies the maximum time to wait between consecutive attempts to write to a channel if the channel is full. If the time exceeds the threshold, an exception is thrown. The corresponding source starts to write at a smaller time value. Each time the source attempts, the digital exponent increases until the current specified value is reached. If data cannot be written, the data write fails. Unit: second
emptyFileEvent	true	Specifies whether to collect empty file information and send it to the sink end. The default value is true , indicating that empty file information is sent to the sink end. This parameter is valid only for HDFS Sink. Taking HDFS Sink as an example, if this parameter is set to true and an empty file exists in the spoolDir directory, an empty file with the same name will be created in the hdfs.path directory of HDFS.

 **NOTE**

SpoolDir Source ignores the last line feed character of each event when data is reading by row. Therefore, Flume does not calculate the data volume counters used by the last line feed character.

- **Kafka Source**

A Kafka source consumes data from Kafka topics. Multiple sources can consume data of the same topic, and the sources consume different partitions of the topic. Common configurations are as follows:

Table 7-4 Common configurations of a Kafka source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured.
type	org.apache.flume.source.kafka.KafkaSource	Specifies the type of the Kafka source, which must be set to org.apache.flume.source.kafka.KafkaSource .
kafka.bootstrap.servers	-	Specifies the bootstrap address port list of Kafka. If Kafka has been installed in the cluster and the configuration has been synchronized to the server, you do not need to set this parameter on the server. The default value is the list of all brokers in the Kafka cluster. This parameter must be configured on the client. Use commas (,) to separate multiple values of <i>IP address:Port number</i> . The rules for matching ports and security protocols must be as follows: port 21007 matches the security mode (SASL_PLAINTEXT), and port 9092 matches the common mode (PLAINTEXT).
kafka.topics	-	Specifies the list of subscribed Kafka topics, which are separated by commas (,).
kafka.topics.regex	-	Specifies the subscribed topics that comply with regular expressions. kafka.topics.regex has a higher priority than kafka.topics and will overwrite kafka.topics .

Parameter	Default Value	Description
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the source is restarted. Unit: second
nodatotime	0 (Disabled)	Specifies the alarm threshold. An alarm is triggered when the duration that Kafka does not release data to subscribers exceeds the threshold. Unit: second This parameter can be configured in the properties.properties file.
batchSize	1000	Specifies the number of events written to the channel in batches.
batchDuration Millis	1000	Specifies the maximum duration of topic data consumption at a time, expressed in milliseconds.
keepTopicInHeader	false	Specifies whether to save topics in the event header. If the parameter value is true , topics configured in Kafka Sink become invalid.
setTopicHeader	true	If this parameter is set to true , the topic name defined in topicHeader is stored in the header.
topicHeader	topic	When setTopicHeader is set to true , this parameter specifies the name of the topic received by the storage device. If the property is used with that of Kafka Sink topicHeader , be careful not to send messages to the same topic cyclically.
useFlumeEventFormat	false	By default, an event is transferred from a Kafka topic to the body of the event in the form of bytes. If this parameter is set to true , the Avro binary format of Flume is used to read events. When used together with the parseAsFlumeEvent parameter with the same name in KafkaSink or KakfaChannel, any set header generated from the data source is retained.

Parameter	Default Value	Description
keepPartitionInHeader	false	Specifies whether to save partition IDs in the event header. If the parameter value is true , Kafka Sink writes data to the corresponding partition.
kafka.consumer.group.id	flume	Specifies the Kafka consumer group ID. Sources or proxies having the same ID are in the same consumer group.
kafka.security.protocol	SASL_PLAINTEXT	Specifies the Kafka security protocol. The parameter value must be set to PLAINTEXT in a common cluster. The rules for matching ports and security protocols must be as follows: port 21007 matches the security mode (SASL_PLAINTEXT), and port 9092 matches the common mode (PLAINTEXT).
Other Kafka Consumer Properties	-	Specifies other Kafka configurations. This parameter can be set to any consumption configuration supported by Kafka, and the .kafka prefix must be added to the configuration.

- **Taildir Source**

A Taildir source monitors file changes in a directory and automatically reads the file content. In addition, it can transmit data in real time. Common configurations are as follows:

Table 7-5 Common configurations of a Taildir source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured.
type	TAILDIR	Specifies the type of the taildir source, which must be set to TAILDIR.
filegroups	-	Specifies the group name of a collection file directory. Group names are separated by spaces.

Parameter	Default Value	Description
filegroups.<filegroupName>.parentDir	-	Specifies the parent directory. The value must be an absolute path.
filegroups.<filegroupName>.filePattern	-	Specifies the relative file path of the file group's parent directory. Directories can be included and regular expressions are supported. It must be used together with parentDir .
positionFile	-	Specifies the metadata storage path during data transmission.
headers.<filegroupName>.<headerKey>	-	Specifies the key-value of an event when data of a group is being collected.
byteOffsetHeader	false	Specifies whether each event header contains the event location information in the source file. If the parameter value is true, the location information is saved in the byteoffset variable.
maxBatchCount	Long.MAX_VALUE	Specifies the maximum number of batches that can be consecutively read from a file. If the monitored directory reads multiple files consecutively and one of the files is written at a rapid rate, other files may fail to be processed. This is because the file that is written at a high speed will be in an infinite read loop. In this case, set this parameter to a smaller value.
skipToEnd	false	Specifies whether Flume can locate the latest location of a file and read the latest data after restart. If the parameter value is true, Flume locates and reads the latest file data after restart.
idleTimeout	120000	Specifies the idle duration during file reading, expressed in milliseconds. If file content is not changed in the preset time duration, close the file. If data is written to this file after the file is closed, open the file and read data.
writePosInterval	3000	Specifies the interval for writing metadata to a file, expressed in milliseconds.

Parameter	Default Value	Description
batchSize	1000	Specifies the number of events written to the channel in batches.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the source is restarted. Unit: second
fileHeader	false	Specifies whether to add the file name (including the file path) to the event header.
fileHeaderKey	file	Specifies that the data storage structure in header is set in the <key,value> mode. Parameters fileHeaderKey and fileHeader must be used together. Following is an example if fileHeader is set to true: Define fileHeaderKey as file . When the /root/a.txt file is read, fileHeaderKey exists in the header in the file=/root/a.txt format.

- **Http Source**

An HTTP source receives data from an external HTTP client and sends the data to the configured channels. Common configurations are as follows:

Table 7-6 Common configurations of an HTTP source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured.
type	http	Specifies the type of the http source, which must be set to http.
bind	-	Specifies the listening host name/IP address.
port	-	Specifies the bound listening port. Ensure that this port is not occupied.

Parameter	Default Value	Description
handler	org.apache.flume.source.http.JSONHandler	Specifies the message parsing method of an HTTP request. Two formats are supported: JSON (org.apache.flume.source.http.JSONHandler) and BLOB (org.apache.flume.sink.solr.morphline.BlobHandler).
handler.*	-	Specifies handler parameters.
exclude-protocols	SSLv3	Specifies the excluded protocols. The entered protocols must be separated by spaces. The default value is SSLv3 .
include-cipher-suites	-	Specifies the included protocols. The entered protocols must be separated by spaces. If this parameter is left empty, all protocols are supported by default.
enableSSL	false	Specifies whether SSL is enabled in HTTP. If this parameter is set to true , the values of keystore and keystore-password must be specified.
keystore-type	JKS	Specifies the keystore type, which can be JKS or PKCS12 .
keystore	-	Specifies the keystore path set after SSL is enabled in HTTP.
keystorePassword	-	Specifies the keystore password set after SSL is enabled in HTTP.

- **Thrift Source**

Thrift Source monitors the thrift port, receives data from the external Thrift clients, and puts the data into the configured channel. Common configurations are as follows:

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured.
type	thrift	Specifies the type of the thrift source, which must be set to thrift .
bind	-	Specifies the listening host name/IP address.

Parameter	Default Value	Description
port	-	Specifies the bound listening port. Ensure that this port is not occupied.
threads	-	Specifies the maximum number of worker threads that can be run.
kerberos	false	Specifies whether Kerberos authentication is enabled.
agent-keytab	-	Specifies the address of the keytab file used by the server. The machine-machine account must be used. You are advised to use flume/conf/flume_server.keytab in the Flume service installation directory.
agent-principal	-	Specifies the principal of the security user used by the server. The principal must be a machine-machine account. You are advised to use the default user of Flume: <code>flume_server/hadoop.<system domain name>@<system domain name></code> NOTE <code>flume_server/hadoop.<system domain name></code> is the username. All letters in the system domain name contained in the username are lowercase letters. For example, Local Domain is set to 9427068F-6EFA-4833-B43E-60CB641E5B6C.COM , and the username is flume_server/hadoop.9427068f-6efa-4833-b43e-60cb641e5b6c.com .
compression-type	none	Specifies the message compression format, which can be set to none or deflate . none indicates that data is not compressed, while deflate indicates that data is compressed.
ssl	false	Specifies whether to use SSL encryption. If this parameter is set to true , the values of keystore and keystore-password must be specified.
keystore-type	JKS	Specifies the keystore type set after SSL is enabled.
keystore	-	Specifies the keystore file path set after SSL is enabled. This parameter is mandatory if SSL is enabled.
keystore-password	-	Specifies the keystore password set after SSL is enabled. This parameter is mandatory if SSL is enabled.

Common Channel Configurations

- **Memory Channel**

A memory channel uses memory as the cache. Events are stored in memory queues. Common configurations are as follows:

Table 7-7 Common configurations of a memory channel

Parameter	Default Value	Description
type	-	Specifies the type of the memory channel, which must be set to memory .
capacity	10000	Specifies the maximum number of events cached in a channel.
transactionCapacity	1000	Specifies the maximum number of events accessed each time. NOTE <ul style="list-style-type: none"> • The parameter value must be greater than the batchSize of the source and sink. • The value of transactionCapacity must be less than or equal to that of capacity.
channelFullcount	10	Specifies the channel full count. When the count reaches the threshold, an alarm is reported.
keep-alive	3	Specifies the waiting time of the Put and Take threads when the transaction or channel cache is full. Unit: second
byteCapacity	80% of the maximum JVM memory	Specifies the total bytes of all event bodies in a channel. The default value is the 80% of the maximum JVM memory (indicated by -Xmx). Unit: bytes

Parameter	Default Value	Description
byteCapacityBufferPercentage	20	Specifies the percentage of bytes in a channel (%).

- **File Channel**

A file channel uses local disks as the cache. Events are stored in the folder specified by **dataDirs**. Common configurations are as follows:

Table 7-8 Common configurations of a file channel

Parameter	Default Value	Description
type	-	Specifies the type of the file channel, which must be set to file .
checkpointDir	\${BIGDATA_DATA_HOME}/ hadoop/data1~N/flume/ checkpoint NOTE This path is changed with the custom data path.	Specifies the checkpoint storage directory.
dataDirs	\${BIGDATA_DATA_HOME}/ hadoop/data1~N/flume/data NOTE This path is changed with the custom data path.	Specifies the data cache directory. Multiple directories can be configured to improve performance. The directories are separated by commas (,).
maxFileSize	2146435071	Specifies the maximum size of a single cache file, expressed in bytes.
minimumRequiredSpace	524288000	Specifies the minimum idle space in the cache, expressed in bytes.
capacity	1000000	Specifies the maximum number of events cached in a channel.

Parameter	Default Value	Description
transactionCapacity	10000	Specifies the maximum number of events accessed each time. NOTE <ul style="list-style-type: none"> The parameter value must be greater than the batchSize of the source and sink. The value of transactionCapacity must be less than or equal to that of capacity.
channelFullCount	10	Specifies the channel full count. When the count reaches the threshold, an alarm is reported.
useDualCheckpoints	false	Specifies the backup checkpoint. If this parameter is set to true , the backupCheckpointDir parameter value must be set.
backupCheckpointDir	-	Specifies the path of the backup checkpoint.
checkpointInterval	30000	Specifies the check interval, expressed in seconds.
keep-alive	3	Specifies the waiting time of the Put and Take threads when the transaction or channel cache is full. Unit: second
use-log-replay-v1	false	Specifies whether to enable the old reply logic.
use-fast-replay	false	Specifies whether to enable the queue reply.
checkpointOnClose	true	Specifies that whether a checkpoint is created when a channel is disabled.

- **Memory File Channel**

A memory file channel uses both memory and local disks as its cache and supports message persistence. It provides similar performance as a memory channel and better performance than a file channel. This channel is currently experimental and not recommended for use in production. The following table describes common configuration items: Common configurations are as follows:

Table 7-9 Common configurations of a memory file channel

Parameter	Default Value	Description
type	org.apache.flume.channel.MemoryFileChannel	Specifies the type of the memory file channel, which must be set to org.apache.flume.channel.MemoryFileChannel .
capacity	50000	Specifies the maximum number of events cached in a channel.
transactionCapacity	5000	Specifies the maximum number of events processed by a transaction. NOTE <ul style="list-style-type: none"> • The parameter value must be greater than the batchSize of the source and sink. • The value of transactionCapacity must be less than or equal to that of capacity.

Parameter	Default Value	Description
subqueueByteCapacity	20971520	<p>Specifies the maximum size of events that can be stored in a subqueue, expressed in bytes.</p> <p>A memory file channel uses both queues and subqueues to cache data. Events are stored in a subqueue, and subqueues are stored in a queue.</p> <p>subqueueCapacity and subqueueInterval determine the size of events that can be stored in a subqueue. subqueueCapacity specifies the capacity of a subqueue, and subqueueInterval specifies the duration that a subqueue can store events. Events in a subqueue are sent to the destination only after the subqueue reaches the upper limit of subqueueCapacity or subqueueInterval.</p> <p>NOTE The value of subqueueByteCapacity must be greater than the number of events specified by batchSize.</p>
subqueueInterval	2000	Specifies the maximum duration that a subqueue can store events, expressed in milliseconds.
keep-alive	3	<p>Specifies the waiting time of the Put and Take threads when the transaction or channel cache is full.</p> <p>Unit: second</p>
dataDir	-	Specifies the cache directory for local files.
byteCapacity	80% of the maximum JVM memory	Specifies the channel cache capacity. Unit: bytes
compression-type	None	Specifies the message compression format, which can be set to none or deflate . none indicates that data is not compressed, while deflate indicates that data is compressed.
channelFullCount	10	Specifies the channel full count. When the count reaches the threshold, an alarm is reported.

The following is a configuration example of a memory file channel:

```
server.channels.c1.type = org.apache.flume.channel.MemoryFileChannel
server.channels.c1.dataDir = /opt/flume/mfdata
server.channels.c1.subqueueByteCapacity = 20971520
server.channels.c1.subqueueInterval=2000
server.channels.c1.capacity = 500000
server.channels.c1.transactionCapacity = 40000
```

- **Kafka Channel**

A Kafka channel uses a Kafka cluster as the cache. Kafka provides high availability and multiple copies to prevent data from being immediately consumed by sinks when Flume or Kafka Broker crashes.

Table 7-10 Common configurations of a Kafka channel

Parameter	Default Value	Description
type	-	Specifies the type of the Kafka channel, which must be set to org.apache.flume.channel.kafka.KafkaChannel .
kafka.bootstrap.servers	-	Specifies the bootstrap address port list of Kafka. If Kafka has been installed in the cluster and the configuration has been synchronized to the server, you do not need to set this parameter on the server. The default value is the list of all brokers in the Kafka cluster. This parameter must be configured on the client. Use commas (,) to separate multiple values of <i>IP address:Port number</i> . The rules for matching ports and security protocols must be as follows: port 21007 matches the security mode (SASL_PLAINTEXT), and port 9092 matches the common mode (PLAINTEXT).

Parameter	Default Value	Description
kafka.topic	flume-channel	Specifies the Kafka topic used by the channel to cache data.
kafka.consumer.group.id	flume	Specifies the data group ID obtained from Kafka. This parameter cannot be left blank.
parseAsFlumeEvent	true	Specifies whether data is parsed into Flume events.
migrateZookeeperOffsets	true	Specifies whether to search for offsets in ZooKeeper and submit them to Kafka when there is no offset in Kafka.
kafka.consumer.auto.offset.reset	latest	Specifies where to consume if there is no offset record, which can be set to earliest , latest , or none . earliest indicates that the offset is reset to the initial point, latest indicates that the offset is set to the latest position, and none indicates that an exception is thrown if there is no offset.

Parameter	Default Value	Description
kafka.producer.security.protocol	SASL_PLAINTEXT	Specifies the Kafka producer security protocol. The rules for matching ports and security protocols must be as follows: port 21007 matches the security mode (SASL_PLAINTEXT), and port 9092 matches the common mode (PLAINTEXT). NOTE If the parameter is not displayed, click + in the lower left corner of the dialog box to display all parameters.
kafka.consumer.security.protocol	SASL_PLAINTEXT	Specifies the Kafka consumer security protocol. The rules for matching ports and security protocols must be as follows: port 21007 matches the security mode (SASL_PLAINTEXT), and port 9092 matches the common mode (PLAINTEXT).
pollTimeout	500	Specifies the maximum timeout interval for the consumer to invoke the poll function. Unit: milliseconds
ignoreLongMessage	false	Specifies whether to discard oversized messages.
messageMaxLength	1000012	Specifies the maximum length of a message written by Flume to Kafka.

Common Sink Configurations

- **HDFS Sink**

An HDFS sink writes data into HDFS. Common configurations are as follows:

Table 7-11 Common configurations of an HDFS sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink.
type	hdfs	Specifies the type of the hdfs sink, which must be set to hdfs .
hdfs.path	-	Specifies the data storage path in HDFS. The value must start with hdfs://hacluster/ .
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the sink is restarted. Unit: second
hdfs.inUseSuffix	.tmp	Specifies the suffix of the HDFS file to which data is being written.
hdfs.rollInterval	30	Specifies the interval for file rolling, expressed in seconds.
hdfs.rollSize	1024	Specifies the size for file rolling, expressed in bytes.
hdfs.rollCount	10	Specifies the number of events for file rolling. NOTE Parameters rollInterval , rollSize , and rollCount can be configured at the same time. The parameter meeting the requirements takes precedence for compression.
hdfs.idleTimeout	0	Specifies the timeout interval for closing idle files automatically, expressed in seconds.
hdfs.batchSize	1000	Specifies the number of events written into HDFS in batches.
hdfs.kerberosPrincipal	-	Specifies the Kerberos principal of HDFS authentication. This parameter is mandatory in a secure mode, but not required in a common mode.
hdfs.kerberosKeytab	-	Specifies the Kerberos keytab of HDFS authentication. This parameter is not required in a common mode, but in a secure mode, the Flume running user must have the permission to access keyTab path in the jaas.conf file.

Parameter	Default Value	Description
hdfs.fileCloseByEvent	true	Specifies whether to close the HDFS file when the last event of the source file is received.
hdfs.batchCallTimeout	-	<p>Specifies the timeout control duration when events are written into HDFS in batches. Unit: milliseconds</p> <p>If this parameter is not specified, the timeout duration is controlled when each event is written into HDFS. When the value of hdfs.batchSize is greater than 0, configure this parameter to improve the performance of writing data into HDFS.</p> <p>NOTE The value of hdfs.batchCallTimeout depends on hdfs.batchSize. A greater hdfs.batchSize requires a larger hdfs.batchCallTimeout. If the value of hdfs.batchCallTimeout is too small, writing events to HDFS may fail.</p>
serializer.appendNewline	true	Specifies whether to add a line feed character (\n) after an event is written to HDFS. If a line feed character is added, the data volume counters used by the line feed character will not be calculated by HDFS sinks.
hdfs.filePrefix	over_ % {base name}	Specifies the file name prefix after data is written to HDFS.
hdfs.fileSuffix	-	Specifies the file name suffix after data is written to HDFS.
hdfs.inUsePrefix	-	Specifies the prefix of the HDFS file to which data is being written.
hdfs.fileType	DataStream	<p>Specifies the HDFS file format, which can be set to SequenceFile, DataStream, or CompressedStream.</p> <p>NOTE If the parameter is set to SequenceFile or DataStream, output files are not compressed, and the codeC parameter cannot be configured. However, if the parameter is set to CompressedStream, the output files are compressed, and the codeC parameter must be configured together.</p>

Parameter	Default Value	Description
hdfs.codeC	-	Specifies the file compression format, which can be set to gzip , bzip2 , lzo , lzop , or snappy .
hdfs.maxOpenFiles	5000	Specifies the maximum number of HDFS files that can be opened. If the number of opened files reaches this value, the earliest opened files are closed.
hdfs.writeFormat	Writable	Specifies the file write format, which can be set to Writable or Text .
hdfs.callTimeout	10000	Specifies the timeout control duration each time events are written into HDFS, expressed in milliseconds.
hdfs.threadsPoolSize	-	Specifies the number of threads used by each HDFS sink for HDFS I/O operations.
hdfs.rollTimerPoolSize	-	Specifies the number of threads used by each HDFS sink to schedule the scheduled file rolling.
hdfs.round	false	Specifies whether to round off the timestamp value. If this parameter is set to true, all time-based escape sequences (except %t) are affected.
hdfs.roundUnit	second	Specifies the unit of the timestamp value that has been rounded off, which can be set to second , minute , or hour .
hdfs.useLocalTimestamp	true	Specifies whether to enable the local timestamp. The recommended parameter value is true .
hdfs.closeTries	0	Specifies the maximum attempts for the hdfs sink to stop renaming a file. If the parameter is set to the default value 0 , the sink does not stop renaming the file until the file is successfully renamed.

Parameter	Default Value	Description
hdfs.retryInterval	180	Specifies the interval of request for closing the HDFS file, expressed in seconds. NOTE For each closing request, there are multiple RPCs working on the NameNode back and forth, which may make the NameNode overloaded if the parameter value is too small. Also, when the parameter is set to 0 , the Sink will not attempt to close the file, but opens the file or uses .tmp as the file name extension, if the first closing attempt fails.
hdfs.failcount	10	Specifies the number of times that data fails to be written to HDFS. If the number of times that the sink fails to write data to HDFS exceeds the parameter value, an alarm indicating abnormal data transmission is reported.

- **Avro Sink**

An Avro sink converts events into Avro events and sends them to the monitoring ports of the hosts. Common configurations are as follows:

Table 7-12 Common configurations of an Avro sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink.
type	-	Specifies the type of the avro sink, which must be set to avro .
hostname	-	Specifies the bound host name or IP address.
port	-	Specifies the bound listening port. Ensure that this port is not occupied.
batch-size	1000	Specifies the number of events sent in a batch.

Parameter	Default Value	Description
client.type	DEFAULT	<p>Specifies the client instance type. Set this parameter based on the communication protocol used by the configured model. The options are as follows:</p> <ul style="list-style-type: none"> • DEFAULT: The client instance of the AvroRPC type is returned. • OTHER: NULL is returned. • THRIFT: The client instance of the Thrift RPC type is returned. • DEFAULT_LOADBALANCING: The client instance of the LoadBalancing RPC type is returned. • DEFAULT_FAILOVER: The client instance of the Failover RPC type is returned.
ssl	false	<p>Specifies whether to use SSL encryption. If this parameter is set to true, the values of keystore and keystore-password must be specified.</p>
truststore-type	JKS	<p>Specifies the Java trust store type, which can be set to JKS or PKCS12.</p> <p>NOTE Different passwords are used to protect the key store and private key of JKS, while the same password is used to protect the key store and private key of PKCS12.</p>
truststore	-	<p>Specifies the Java trust store file.</p>

Parameter	Default Value	Description
truststore-password	-	Specifies the Java trust store password.
keystore-type	JKS	Specifies the keystore type set after SSL is enabled.
keystore	-	Specifies the keystore file path set after SSL is enabled. This parameter is mandatory if SSL is enabled.
keystore-password	-	Specifies the keystore password after SSL is enabled. This parameter is mandatory if SSL is enabled.
connect-timeout	20000	Specifies the timeout for the first connection, expressed in milliseconds.
request-timeout	20000	Specifies the maximum timeout for a request after the first request, expressed in milliseconds.
reset-connection-interval	0	Specifies the interval between a connection failure and a second connection, expressed in seconds. If the parameter is set to 0 , the system continuously attempts to perform a connection.

Parameter	Default Value	Description
compression-type	none	Specifies the compression type of the batch data, which can be set to none or deflate . none indicates that data is not compressed, while deflate indicates that data is compressed. This parameter value must be the same as that of the AvroSource compression-type.
compression-level	6	Specifies the compression level of batch data, which can be set to 1 to 9 . A larger value indicates a higher compression rate.
exclude-protocols	SSLv3	Specifies the excluded protocols. The entered protocols must be separated by spaces. The default value is SSLv3 .

- **HBase Sink**

An HBase sink writes data into HBase. Common configurations are as follows:

Table 7-13 Common configurations of an HBase sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink.
type	-	Specifies the type of the HBase sink, which must be set to hbase .
table	-	Specifies the HBase table name.
columnFamily	-	Specifies the HBase column family.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the sink is restarted. Unit: second

Parameter	Default Value	Description
batchSize	1000	Specifies the number of events written into HBase in batches.
kerberosPrincipal	-	Specifies the Kerberos principal of HBase authentication. This parameter is mandatory in a secure mode, but not required in a common mode.
kerberosKeytab	-	Specifies the Kerberos keytab of HBase authentication. This parameter is not required in a common mode, but in a secure mode, the Flume running user must have the permission to access keyTab path in the jaas.cof file.
coalesceIncrements	true	Specifies whether to perform multiple operations on the same hbase cell in a same processing batch. Setting this parameter to true improves performance.

- **Kafka Sink**

A Kafka sink writes data into Kafka. Common configurations are as follows:

Table 7-14 Common configurations of a Kafka sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink.
type	-	Specifies the type of the kafka sink, which must be set to org.apache.flume.sink.kafka.KafkaSink .

Parameter	Default Value	Description
kafka.bootstrap.servers	-	Specifies the bootstrap address port list of Kafka. If Kafka has been installed in the cluster and the configuration has been synchronized to the server, you do not need to set this parameter on the server. The default value is the list of all brokers in the Kafka cluster. The client must be configured with this parameter. If there are multiple values, use commas (,) to separate the values. The rules for matching ports and security protocols must be as follows: port 21007 matches the security mode (SASL_PLAINTEXT), and port 9092 matches the common mode (PLAINTEXT).
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the sink is restarted. Unit: second
kafka.producer.acks	1	Successful write is determined by the number of received acknowledgement messages about replicas. The value 0 indicates that no confirm message needs to be received, the value 1 indicates that the system is only waiting for only the acknowledgement information from a leader, and the value -1 indicates that the system is waiting for the acknowledgement messages of all replicas. If this parameter is set to -1 , data loss can be avoided in some leader failure scenarios.
kafka.topic	-	Specifies the topic to which data is written. This parameter is mandatory.
flumeBatchSize	1000	Specifies the number of events written into Kafka in batches.
kafka.security.protocol	SASL_PLAINTEXT	Specifies the Kafka security protocol. The parameter value must be set to PLAINTEXT in a common cluster. The rules for matching ports and security protocols must be as follows: port 21007 matches the security mode (SASL_PLAINTEXT), and port 9092 matches the common mode (PLAINTEXT).

Parameter	Default Value	Description
ignoreLongMessage	false	Specifies whether to discard oversized messages.
messageMaxLength	1000012	Specifies the maximum length of a message written by Flume to Kafka.
defaultPartitionId	-	Specifies the Kafka partition ID to which the events of a channel is transferred. The partitionIdHeader value overwrites this parameter value. By default, if this parameter is left blank, events will be distributed by the Kafka Producer's partitioner (by a specified key or a partitioner customized by kafka.partitionner.class).
partitionIdHeader	-	When you set this parameter, the sink will take the value of the field named using the value of this property from the event header and send the message to the specified partition of the topic. If the value does not have a valid partition, EventDeliveryException is thrown. If the header value already exists, this setting overwrites the defaultPartitionId parameter.
Other Kafka Producer Properties	-	Specifies other Kafka configurations. This parameter can be set to any production configuration supported by Kafka, and the .kafka prefix must be added to the configuration.

- **Thrift Sink**

A Thrift sink converts events to Thrift events and sends them to the monitoring port of the configured host. Common configurations are as follows:

Table 7-15 Common configurations of a Thrift sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink.
type	thrift	Specifies the type of the thrift sink, which must be set to thrift .

Parameter	Default Value	Description
hostname	-	Specifies the bound host name or IP address.
port	-	Specifies the bound listening port. Ensure that this port is not occupied.
batch-size	1000	Specifies the number of events sent in a batch.
connect-timeout	20000	Specifies the timeout for the first connection, expressed in milliseconds.
request-timeout	20000	Specifies the maximum timeout for a request after the first request, expressed in milliseconds.
kerberos	false	Specifies whether Kerberos authentication is enabled.
client-keytab	-	Specifies the path of the client keytab file. The Flume running user must have the access permission on the authentication file.
client-principal	-	Specifies the principal of the security user used by the client.
server-principal	-	Specifies the principal of the security user used by the server.
compression-type	none	Specifies the compression type of data sent by Flume, which can be set to none or deflate . none indicates that data is not compressed, while deflate indicates that data is compressed.

Parameter	Default Value	Description
maxConnections	5	Specifies the maximum size of the connection pool for Flume to send data.
ssl	false	Specifies whether to use SSL encryption.
truststore-type	JKS	Specifies the Java trust store type.
truststore	-	Specifies the Java trust store file.
truststore-password	-	Specifies the Java trust store password.
reset-connection-interval	0	Specifies the interval between a connection failure and a second connection, expressed in seconds. If the parameter is set to 0 , the system continuously attempts to perform a connection.

Precautions

- What are the reliability measures of Flume?
 - Use the transaction mechanisms between Source and Channel as well as between Channel and Sink.
 - Configure the failover and load_balance mechanisms for Sink Processor. The following shows a load balancing example.


```
server.sinkgroups=g1
server.sinkgroups.g1.sinks=k1 k2
server.sinkgroups.g1.processor.type=load_balance
server.sinkgroups.g1.processor.backoff=true
server.sinkgroups.g1.processor.selector=random
```
- What are the precautions for the aggregation and cascading of multiple Flume agents?
 - Avro or Thrift protocol can be used for cascading.
 - When the aggregation end contains multiple nodes, evenly distribute the agents and do not aggregate all agents on a single node.

7.8 Flume Configuration Parameter Description

For versions earlier than MRS 3.x, configure Flume parameters in the **properties.properties** file.

For MRS 3.x or later, some parameters can be configured on Manager.

Overview

This section describes how to configure the sources, channels, and sinks of Flume, and modify the configuration items of each module.

For MRS 3.x or later, log in to FusionInsight Manager and choose **Cluster > Services > Flume**. On the displayed page, click the **Configuration Tool** tab to configure the **source**, **channel**, and **sink** parameters. Parameters such as **channels** and **type** are configured only in the client configuration file **properties.properties**, the path of which is *Flume client installation directory/fusioninsight-flume-Flume version/conf/properties.properties*.

NOTE

You must input encrypted information for some configurations. For details on how to encrypt information, see [Using the Encryption Tool of the Flume Client](#).

Common Source Configurations

- **Avro Source**

An Avro source listens to the Avro port, receives data from the external Avro client, and places data into configured channels. [Table 7-16](#) lists common configurations.

Table 7-16 Common configurations of an Avro source

Parameter	Default Value	Description
channels	-	<p>Specifies the channel connected to the source. Multiple channels can be configured. Use spaces to separate them.</p> <p>In a single proxy process, sources and sinks are connected through channels. A source instance corresponds to multiple channels, but a sink instance corresponds only to one channel.</p> <p>The format is as follows:</p> <pre><Agent >.sources.<Source>.channels = <channel1> <channel2> <channel3>...</pre> <pre><Agent >.sinks.<Sink>.channels = <channel1></pre> <p>This parameter can be configured only in the properties.properties file.</p>

Parameter	Default Value	Description
type	avro	Specifies the type, which is set to avro . The type of each source is a fixed value. This parameter can be configured only in the properties.properties file.
bind	-	Specifies the host name or IP address associated with the source.
port	-	Specifies the bound port number.
ssl	false	Specifies whether to use SSL encryption. <ul style="list-style-type: none"> • true • false
truststore-type	JKS	Specifies the Java trust store type. Set this parameter to JKS or other truststore types supported by Java.
truststore	-	Specifies the Java trust store file.
truststore-password	-	Specifies the Java trust store password.
keystore-type	JKS	Specifies the key storage type. Set this parameter to JKS or other truststore types supported by Java.
keystore	-	Specifies the key storage file.
keystore-password	-	Specifies the key storage password.

- **SpoolDir Source**

A SpoolDir source monitors and transmits new files that have been added to directories in quasi-real-time mode. Common configurations are as follows:

Table 7-17 Common configurations of a SpoolDir source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured. This parameter can be configured only in the properties.properties file.

Parameter	Default Value	Description
type	spooldir	Type, which is set to spooldir . This parameter can be configured only in the properties.properties file.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the source is restarted. Unit: second
spoolDir	-	Specifies the monitoring directory.
fileSuffix	.COMPLETED	Specifies the suffix added after file transmission is complete.
deletePolicy	never	Specifies the source file deletion policy after file transmission is complete. The value can be either never or immediate .
ignorePattern	^\$	Specifies the regular expression of a file to be ignored.
trackerDir	.flumespool	Specifies the metadata storage path during data transmission.
batchSize	1000	Specifies the source transmission granularity.
decodeErrorPolicy	FAIL	Specifies the code error policy. This parameter can be configured only in the properties.properties file. The value can be FAIL , REPLACE , or IGNORE . FAIL : Generate an exception and fail the parsing. REPLACE : Replace the characters that cannot be identified with other characters, such as U+FFFD. IGNORE : Discard character strings that cannot be parsed. NOTE If a code error occurs in the file, set decodeErrorPolicy to REPLACE or IGNORE . Flume will skip the code error and continue to collect subsequent logs.

Parameter	Default Value	Description
deserializer	LINE	Specifies the file parser. The value can be either LINE or BufferedLine . <ul style="list-style-type: none"> When the value is set to LINE, characters read from the file are transcoded one by one. When the value is set to BufferedLine, one line or multiple lines of characters read from the file are transcoded in batches, which delivers better performance.
deserializer.maxLineLength	2048	Specifies the maximum length for resolution by line, ranging from 0 to 2,147,483,647.
deserializer.maxBatchLine	1	Specifies the maximum number of lines for resolution by line. If multiple lines are set, maxLineLength must be set to a corresponding multiplier. For example, if maxBatchLine is set to 2 , maxLineLength is set to 4096 (2048 x 2).
selector.type	replicating	Specifies the selector type. The value can be either replicating or multiplexing . <ul style="list-style-type: none"> replicating indicates that the same content is sent to each channel. multiplexing indicates that the content is sent only to certain channels according to the distribution rule.
interceptors	-	Specifies the interceptor. For details, see the Flume official document . This parameter can be configured only in the properties.properties file.

 **NOTE**

The Spooling source ignores the last line feed character of each event when data is read by line. Therefore, Flume does not calculate the data volume counters used by the last line feed character.

- **Kafka Source**

A Kafka source consumes data from Kafka topics. Multiple sources can consume data of the same topic, and the sources consume different partitions of the topic. Common configurations are as follows:

Table 7-18 Common configurations of a Kafka source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured. This parameter can be configured only in the properties.properties file.
type	org.apache.flume.source.kafka.KafkaSource	Specifies the type, which is set to org.apache.flume.source.kafka.KafkaSource . This parameter can be configured only in the properties.properties file.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the source is restarted. Unit: second
nodatotime	0 (Disabled)	Specifies the alarm threshold. An alarm is triggered when the duration that Kafka does not release data to subscribers exceeds the threshold. Unit: second
batchSize	1000	Specifies the number of events written into a channel at a time.
batchDurationMillis	1000	Specifies the maximum duration of topic data consumption at a time, expressed in milliseconds.
keepTopicInHeader	false	Specifies whether to save topics in the event header. If topics are saved, topics configured in Kafka sinks become invalid. <ul style="list-style-type: none"> • true • false This parameter can be configured only in the properties.properties file.

Parameter	Default Value	Description
keepPartitionIn-Header	false	Specifies whether to save partition IDs in the event header. If partition IDs are saved, Kafka sinks write data to the corresponding partitions. <ul style="list-style-type: none"> • true • false This parameter can be set only in the properties.properties file.
kafka.bootstrap.servers	-	Specifies the list of Broker addresses, which are separated by commas.
kafka.consumer.group.id	-	Specifies the Kafka consumer group ID.
kafka.topics	-	Specifies the list of subscribed Kafka topics, which are separated by commas (,).
kafka.topics.regex	-	Specifies the subscribed topics that comply with regular expressions. kafka.topics.regex has a higher priority than kafka.topics and will overwrite kafka.topics .
kafka.security.protocol	SASL_PLAINTEXT	Specifies the security protocol of Kafka. The value must be set to PLAINTEXT for clusters in which Kerberos authentication is disabled.
kafka.kerberos.domain.name	-	Specifies the value of default_realm of Kerberos in the Kafka cluster, which should be configured only for security clusters. This parameter can be set only in the properties.properties file.
Other Kafka Consumer Properties	-	Specifies other Kafka configurations. This parameter can be set to any consumption configuration supported by Kafka, and the .kafka prefix must be added to the configuration. This parameter can be set only in the properties.properties file.

- **Taildir Source**

A Taildir source monitors file changes in a directory and automatically reads the file content. In addition, it can transmit data in real time. [Table 7-19](#) lists common configurations.

Table 7-19 Common configurations of a Taildir source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured. This parameter can be set only in the properties.properties file.
type	taildir	Specifies the type, which is set to taildir . This parameter can be set only in the properties.properties file.
filegroups	-	Specifies the group name of a collection file directory. Group names are separated by spaces.
filegroups.<filegroup Name>.parentDir	-	Specifies the parent directory. The value must be an absolute path. This parameter can be set only in the properties.properties file.
filegroups.<filegroup Name>.filePattern	-	Specifies the relative file path of the file group's parent directory. Directories can be included and regular expressions are supported. It must be used together with parentDir . This parameter can be set only in the properties.properties file.
positionFile	-	Specifies the metadata storage path during data transmission.
headers.<filegroup Name>.<headerKey>	-	Specifies the key-value of an event when data of a group is being collected. This parameter can be set only in the properties.properties file.
byteOffsetHeader	false	Specifies whether each event header should contain the location information about the event in the source file. The location information is saved in the byteoffset variable.

Parameter	Default Value	Description
skipToEnd	false	Specifies whether Flume can locate the latest location of a file and read the latest data after restart.
idleTimeout	120000	Specifies the idle duration during file reading, expressed in milliseconds. If the file data is not changed in this idle period, the source closes the file. If data is written into this file after it is closed, the source opens the file and reads data.
writePosInterval	3000	Specifies the interval for writing metadata to a file, expressed in milliseconds.
batchSize	1000	Specifies the number of events written to the channel in batches.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the source is restarted. Unit: second

- **Http Source**

An HTTP source receives data from an external HTTP client and sends the data to the configured channels. [Table 7-20](#) lists common configurations.

Table 7-20 Common configurations of an HTTP source

Parameter	Default Value	Description
channels	-	Specifies the channel connected to the source. Multiple channels can be configured. This parameter can be set only in the properties.properties file.
type	http	Specifies the type, which is set to http . This parameter can be set only in the properties.properties file.
bind	-	Specifies the name or IP address of the bound host.
port	-	Specifies the bound port.

Parameter	Default Value	Description
handler	org.apache.flume.source.http.JSONHandler	Specifies the message parsing method of an HTTP request. The following methods are supported: <ul style="list-style-type: none"> org.apache.flume.source.http.JSONHandler: JSON org.apache.flume.sink.solr.morphline.BlobHandler: BLOB
handler.*	-	Specifies handler parameters.
enableSSL	false	Specifies whether SSL is enabled in HTTP.
keystore	-	Specifies the keystore path set after SSL is enabled in HTTP.
keystorePassword	-	Specifies the keystore password set after SSL is enabled in HTTP.

Common Channel Configurations

- **Memory Channel**

A memory channel uses memory as the cache. Events are stored in memory queues. [Table 7-21](#) lists common configurations.

Table 7-21 Common configurations of a memory channel

Parameter	Default Value	Description
type	-	Specifies the type, which is set to memory . This parameter can be set only in the properties.properties file.
capacity	10000	Specifies the maximum number of events cached in a channel.
transactionCapacity	1000	Specifies the maximum number of events accessed each time.
channelFullcount	10	Specifies the channel full count. When the count reaches the threshold, an alarm is reported.

- **File Channel**

A file channel uses local disks as the cache. Events are stored in the folder specified by **dataDirs**. [Table 7-22](#) lists common configurations.

Table 7-22 Common configurations of a file channel

Parameter	Default Value	Description
type	-	Specifies the type, which is set to file . This parameter can be set only in the properties.properties file.
checkpointDir	\${BIGDATA_DATA_HOME}/flume/checkpoint	Specifies the checkpoint storage directory.
dataDirs	\${BIGDATA_DATA_HOME}/flume/data	Specifies the data cache directory. Multiple directories can be configured to improve performance. The directories are separated by commas (,).
maxFileSize	2146435071	Specifies the maximum size of a single cache file, expressed in bytes.
minimumRequired-Space	524288000	Specifies the minimum idle space in the cache, expressed in bytes.
capacity	1000000	Specifies the maximum number of events cached in a channel.
transactionCapacity	10000	Specifies the maximum number of events accessed each time.
channelfullcount	10	Specifies the channel full count. When the count reaches the threshold, an alarm is reported.

- **Kafka Channel**

A Kafka channel uses a Kafka cluster as the cache. Kafka provides high availability and multiple copies to prevent data from being immediately consumed by sinks when Flume or Kafka Broker crashes. [Table 10 Common configurations of a Kafka channel](#) lists common configurations.

Table 7-23 Common configurations of a Kafka channel

Parameter	Default Value	Description
type	-	Specifies the type, which is set to org.apache.flume.channel.kafka.KafkaChannel . This parameter can be set only in the properties.properties file.

Parameter	Default Value	Description
kafka.bootstrap.servers	-	Specifies the list of Brokers in the Kafka cluster.
kafka.topic	flume-channel	Specifies the Kafka topic used by the channel to cache data.
kafka.consumer.group.id	flume	Specifies the Kafka consumer group ID.
parseAsFlumeEvent	true	Specifies whether data is parsed into Flume events.
migrateZookeeper-Offsets	true	Specifies whether to search for offsets in ZooKeeper and submit them to Kafka when there is no offset in Kafka.
kafka.consumer.auto.offset.reset	latest	Consumes data from the specified location when there is no offset.
kafka.producer.security.protocol	SASL_PLAINTEXT	Specifies the Kafka producer security protocol.
kafka.consumer.security.protocol	SASL_PLAINTEXT	Specifies the Kafka consumer security protocol.

Common Sink Configurations

- **HDFS Sink**

An HDFS sink writes data into HDFS. [Table 7-24](#) lists common configurations.

Table 7-24 Common configurations of an HDFS sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink. This parameter can be set only in the properties.properties file.
type	hdfs	Specifies the type, which is set to hdfs . This parameter can be set only in the properties.properties file.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the sink is restarted. Unit: second
hdfs.path	-	Specifies the HDFS path.

Parameter	Default Value	Description
hdfs.inUseSuffix	.tmp	Specifies the suffix of the HDFS file to which data is being written.
hdfs.rollInterval	30	Specifies the interval for file rolling, expressed in seconds.
hdfs.rollSize	1024	Specifies the size for file rolling, expressed in bytes.
hdfs.rollCount	10	Specifies the number of events for file rolling.
hdfs.idleTimeout	0	Specifies the timeout interval for closing idle files automatically, expressed in seconds.
hdfs.batchSize	1000	Specifies the number of events written into HDFS at a time.
hdfs.kerberosPrincipal	-	Specifies the Kerberos username for HDFS authentication. This parameter is not required for a cluster in which Kerberos authentication is disabled.
hdfs.kerberosKeytab	-	Specifies the Kerberos keytab of HDFS authentication. This parameter is not required for a cluster in which Kerberos authentication is disabled.
hdfs.fileCloseByEvent	true	Specifies whether to close the file when the last event is received.
hdfs.batchCallTimeout	-	<p>Specifies the timeout control duration each time events are written into HDFS, expressed in milliseconds.</p> <p>If this parameter is not specified, the timeout duration is controlled when each event is written into HDFS. When the value of hdfs.batchSize is greater than 0, configure this parameter to improve the performance of writing data into HDFS.</p> <p>NOTE The value of hdfs.batchCallTimeout depends on hdfs.batchSize. A greater hdfs.batchSize requires a larger hdfs.batchCallTimeout. If the value of hdfs.batchCallTimeout is too small, writing events to HDFS may fail.</p>

Parameter	Default Value	Description
serializer.appendNewLine	true	Specifies whether to add a line feed character (\n) after an event is written to HDFS. If a line feed character is added, the data volume counters used by the line feed character will not be calculated by HDFS sinks.

- **Avro Sink**

An Avro sink converts events into Avro events and sends them to the monitoring ports of the hosts. [Table 7-25](#) lists common configurations.

Table 7-25 Common configurations of an Avro sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink. This parameter can be set only in the properties.properties file.
type	-	Specifies the type, which is set to avro . This parameter can be set only in the properties.properties file.
hostname	-	Specifies the name or IP address of the bound host.
port	-	Specifies the monitoring port.
batch-size	1000	Specifies the number of events sent in a batch.
ssl	false	Specifies whether to use SSL encryption.
truststore-type	JKS	Specifies the Java trust store type.
truststore	-	Specifies the Java trust store file.
truststore-password	-	Specifies the Java trust store password.
keystore-type	JKS	Specifies the key storage type.
keystore	-	Specifies the key storage file.
keystore-password	-	Specifies the key storage password.

- **HBase Sink**

An HBase sink writes data into HBase. [Table 7-26](#) lists common configurations.

Table 7-26 Common configurations of an HBase sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink. This parameter can be set only in the properties.properties file.
type	-	Specifies the type, which is set to hbase . This parameter can be set only in the properties.properties file.
table	-	Specifies the HBase table name.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the sink is restarted. Unit: second
columnFamily	-	Specifies the HBase column family.
batchSize	1000	Specifies the number of events written into HBase at a time.
kerberosPrincipal	-	Specifies the Kerberos username for HBase authentication. This parameter is not required for a cluster in which Kerberos authentication is disabled.
kerberosKeytab	-	Specifies the Kerberos keytab of HBase authentication. This parameter is not required for a cluster in which Kerberos authentication is disabled.

- **Kafka Sink**

A Kafka sink writes data into Kafka. [Table 7-27](#) lists common configurations.

Table 7-27 Common configurations of a Kafka sink

Parameter	Default Value	Description
channel	-	Specifies the channel connected to the sink. This parameter can be set only in the properties.properties file.
type	-	Specifies the type, which is set to org.apache.flume.sink.kafka.Kafka Sink . This parameter can be set only in the properties.properties file.

Parameter	Default Value	Description
kafka.bootstrap.servers	-	Specifies the list of Kafka Brokers, which are separated by commas.
monTime	0 (Disabled)	Specifies the thread monitoring threshold. When the update time exceeds the threshold, the sink is restarted. Unit: second
kafka.topic	default-flume-topic	Specifies the topic where data is written.
flumeBatchSize	1000	Specifies the number of events written into Kafka at a time.
kafka.security.protocol	SASL_PLAINTEXT	Specifies the security protocol of Kafka. The value must be set to PLAINTEXT for clusters in which Kerberos authentication is disabled.
kafka.kerberos.domain.name	-	Specifies the Kafka domain name. This parameter is mandatory for a security cluster. This parameter can be set only in the properties.properties file.
Other Kafka Producer Properties	-	Specifies other Kafka configurations. This parameter can be set to any production configuration supported by Kafka, and the .kafka prefix must be added to the configuration. This parameter can be set only in the properties.properties file.

7.9 Using Environment Variables in the properties.properties File

Scenario

This section describes how to use environment variables in the **properties.properties** configuration file.

This section applies to MRS 3.x and later versions.

Prerequisites

You have installed Flume service or client.

Procedure

- Step 1** Add variables to the *Flume client installation directory/fusioninsight-flume-Flume Version/conf/flume-env.sh* file.

Add variables:

```
export Variable name = Variable value
```

Example:

```
JAVA_OPTS="-Xms2G -Xmx4G -XX:CMSFullGCsBeforeCompaction=1 -XX:  
+UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled -XX:  
+UseCMSCompactAtFullCollection -  
DpropertiesImplementation=org.apache.flume.node.EnvVarResolverProperties  
"
```

```
export TAILDIR_PATH=/tmp/flumetest/201907/20190703/1/*.log.*
```

- Step 2** Restart the Flume instance process.

1. Log in to FusionInsight Manager.
2. Choose **Cluster** > *Name of the target cluster* > **Services** > **Flume** > **Instance**, select all Flume instances, choose **More** > **Restart Instance**, enter the password, and click **OK**.

NOTICE

- Do not restart the Flume process by restarting the Flume service on the Manager page, after **flume-env.sh** takes effect in the flume server. Otherwise, user-defined environment variables are lost.
- Ensure that **flume-env.sh** takes effect before configuring the **properties.properties** as instructed in [Step 3](#) and uploading the file on the GUI. If the operation sequence is not standard, user-defined environment variables may be lost.

- Step 3** In the **properties.properties** configuration file, use the `${variable name}` format to reference the variable, taking the client as an example.

Example:

```
client.sources.s1.type = TAILDIR  
client.sources.s1.filegroups = f1  
client.sources.s1.filegroups.f1 = ${TAILDIR_PATH}  
client.sources.s1.positionFile = /tmp/flumetest/201907/20190703/1/taildir_position.json  
client.sources.s1.channels = c1
```

----End

7.10 Non-Encrypted Transmission

7.10.1 Configuring Non-encrypted Transmission

Scenario

This section describes how to configure Flume server and client parameters after the cluster and the Flume service are installed to ensure proper running of the service.

This section applies to MRS 3.x and later versions.

NOTE

By default, the cluster network environment is secure and the SSL authentication is not enabled during the data transmission process. For details about how to use the encryption mode, see [Configuring the Encrypted Transmission](#).

Prerequisites

- The cluster and Flume service have been installed.
- The network environment of the cluster is secure.

Procedure

Step 1 Configure the client parameters of the Flume role.

1. Use the Flume configuration tool on FusionInsight Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager. Choose **Cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **client**. Select and drag the source, channel, and sink to be used to the GUI on the right, and connect them.
For example, use SpoolDir Source, File Channel, and Avro Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by referring to [Table 7-28](#) based on the actual environment.

NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory/fusioninsight-flume-1.9.0/conf/properties.properties* to ensure that the configuration is in concordance with the previous. Log in to FusionInsight Manager, choose **Cluster > Services > Flume > Configuration > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
- d. Click **Export** to save the **properties.properties** configuration file to the local server.

Table 7-28 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
ssl	<p>Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.)</p> <p>Only Sources of the Avro type have this configuration item.</p> <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the function is not enabled. 	false

2. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.

Step 2 Configure the server parameters of the Flume role and upload the configuration file to the cluster.

1. Use the Flume configuration tool on the FusionInsight Manager portal to configure the server parameters and generate the configuration file.
 - a. Log in to FusionInsight Manager. Choose **Cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **server**. Select and drag the source, channel, and sink to be used to the GUI on the right, and connect them.
For example, use Avro Source, File Channel, and HDFS Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by referring to **Table 7-29** based on the actual environment.

 **NOTE**

- If the server parameters of the Flume role have been configured, you can choose **Cluster > Services > Flume > Instance** on FusionInsight Manager. Then select the corresponding Flume role instance and click the **Download** button behind the **flume.config.file** parameter on the **Instance Configurations** page to obtain the existing server parameter configuration file. Choose **Cluster > Service > Flume > Configurations > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
 - A unique checkpoint directory needs to be configured for each File Channel.
- d. Click **Export** to save the **properties.properties** configuration file to the local server.

Table 7-29 Parameters to be modified of the Flume role server

Parameter	Description	Example Value
ssl	<p>Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.)</p> <p>Only Sources of the Avro type have this configuration item.</p> <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the function is not enabled. 	false

2. Log in to FusionInsight Manager and choose **Cluster > Services > Flume**. On the **Instances** tab page, click **Flume**.
3. Select the Flume role of the node where the configuration file is to be uploaded, choose **Instance Configurations > Import** beside the **flume.config.file**, and select the **properties.properties** file.

 **NOTE**

- An independent server configuration file can be uploaded to each Flume instance.
 - This step is required for updating the configuration file. Modifying the configuration file on the background is an improper operation because the modification will be overwritten after configuration synchronization.
4. Click **Save**, and then click **OK**.
 5. Click **Finish**.

----End

7.10.2 Typical Scenario: Collecting Local Static Logs and Uploading Them to Kafka

Scenario

This section describes how to use Flume to collect static logs from a local host (service IP address: 192.168.108.11) and save them to the topic list (test1) of Kafka.

This section applies to MRS 3.x and later versions.

 **NOTE**

By default, the cluster network environment is secure and the SSL authentication is not enabled during the data transmission process. For details about how to use the encryption mode, see [Configuring the Encrypted Transmission](#). The configuration can apply to scenarios where only the server is configured, for example, Server:Spooldir Source+File Channel+Kafka Sink.

Prerequisites

- The cluster, Kafka, and Flume service have been installed.
- The network environment of the cluster is secure.
- You have understood service requirements and prepared Kafka administrator `flume_kafka`.

Procedure

Step 1 Configure the client parameters of the Flume role.

1. Use the Flume configuration tool on Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager. Choose **Cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **client**. Select and drag the source, channel, and sink to be used to the GUI on the right, and connect them.
Use SpoolDir Source, File Channel, and Avro Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by referring to [Table 7-30](#) based on the actual environment.

NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory/fusioninsight-flume-1.9.0/conf/properties.properties* to ensure that the configuration is in concordance with the previous. Log in to Manager, choose **Cluster > Services > Flume > Configurations > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
- d. Click **Export** to save the **properties.properties** configuration file to the local server.

Table 7-30 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test

Parameter	Description	Example Value
spoolDir	Specifies the directory where the file to be collected resides. This parameter cannot be left blank. The directory needs to exist and have the write, read, and execute permissions on the flume running user.	/srv/BigData/hadoop/data1/zb
trackerDir	Specifies the path for storing the metadata of files collected by Flume.	/srv/BigData/hadoop/data1/tracker
batchSize	Specifies the number of events that Flume sends in a batch (number of data pieces). A larger value indicates higher performance and lower timeliness.	61200
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/data

Parameter	Description	Example Value
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hostname	Specifies the name or IP address of the host whose data is to be sent. This parameter cannot be left blank. The parameter must be configured to be the name or IP address of the host where the connected Avro Source resides.	192.168.108.11

Parameter	Description	Example Value
port	Specifies the port that sends the data. This parameter cannot be left blank. It must be configured to be the port that is listened to by the connected Avro Source.	21154
ssl	Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) Only Sources of the Avro type have this configuration item. <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the function is not enabled. 	false

2. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.

Step 2 Configure the server parameters of the Flume role and upload the configuration file to the cluster.

1. Use the Flume configuration tool on the FusionInsight Manager portal to configure the server parameters and generate the configuration file.
 - a. Log in to FusionInsight Manager. Choose **Cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **server**. Select and drag the source, channel, and sink to be used to the GUI on the right, and connect them.
Avro Source, File Channel, and Kafka Sink are used.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by referring to **Table 7-31** based on the actual environment.

 NOTE

- If the server parameters of the Flume role have been configured, you can choose **Cluster > Services > Flume > Instance** on Manager. Then select the corresponding Flume role instance and click the **Download** button behind the **flume.config.file** parameter on the **Instance Configurations** page to obtain the existing server parameter configuration file. Choose **Cluster > Services > Flume > Configurations > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
 - A unique checkpoint directory needs to be configured for each File Channel.
- d. Click **Export** to save the **properties.properties** configuration file to the local server.

Table 7-31 Parameters to be modified of the Flume role server

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
bind	Specifies the IP address to which Avro Source is bound. This parameter cannot be left blank. It must be configured as the IP address that the server configuration file will upload.	192.168.108.11
port	Specifies the port that Avro Source listens to. This parameter cannot be left blank. It must be configured as an unused port.	21154
ssl	Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) Only Sources of the Avro type have this configuration item. <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the function is not enabled. 	false

Parameter	Description	Example Value
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/data
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
kafka.topics	Specifies the list of subscribed Kafka topics, which are separated by commas (,). This parameter cannot be left blank.	test1

Parameter	Description	Example Value
kafka.bootstrap.servers	Specifies the bootstrap IP address and port list of Kafka. The default value is all Kafka lists in a Kafka cluster. If Kafka has been installed in the cluster and its configurations have been synchronized, this parameter can be left blank.	192.168.101.10:21007

2. Log in to FusionInsight Manager and choose **Cluster > Services > Flume**. On the **Instance** tab page, click the **Flume** role.
3. Select the Flume role of the node where the configuration file is to be uploaded, choose **Instance Configurations > Import** beside the **flume.config.file**, and select the **properties.properties** file.

 **NOTE**

- An independent server configuration file can be uploaded to each Flume instance.
 - This step is required for updating the configuration file. Modifying the configuration file on the background is an improper operation because the modification will be overwritten after configuration synchronization.
4. Click **Save**, and then click **OK**.
 5. Click **Finish**.

Step 3 Verify log transmission.

1. Log in to the Kafka client.
cd /Client installation directory/Kafka/kafka
kinit flume_kafka (Enter the password.)
2. Read data from a Kafka topic.
bin/kafka-console-consumer.sh --topic topic name --bootstrap-server Kafka service IP address of the node where the role instance is located: 21007 --consumer.config config/consumer.properties --from-beginning

The system displays the contents of the file to be collected.

```
[root@host1 kafka]# bin/kafka-console-consumer.sh --topic test1 --bootstrap-server 192.168.101.10:21007 --consumer.config config/consumer.properties --from-beginning
Welcome to flume
```

----End

7.10.3 Typical Scenario: Collecting Local Static Logs and Uploading Them to HDFS

Scenario

This section describes how to use Flume to collect static logs from a local PC (for example, the service IP address is 192.168.108.11) and save them to the **/flume/test** directory on HDFS.

This section applies to MRS 3.x or later.

 NOTE

By default, the cluster network environment is secure and the SSL authentication is not enabled during the data transmission process. For details about how to use the encryption mode, see [Configuring the Encrypted Transmission](#). This configuration can use only one Flume scenario, for example, Server:SpoolDir Source+File Channel+HDFS Sink.

Prerequisites

- The cluster, HDFS, and Flume service have been installed.
- The network environment of the cluster is secure.
- User **flume_hdfs** has been created, and the HDFS directory and data used for log verification have been authorized to the user.

Procedure

Step 1 On FusionInsight Manager, choose **System > Permission > User**, select user **flume_hdfs**, and choose **More > Download Authentication Credential** to download the Kerberos certificate file of user **flume_hdfs** and save it to the local host.

Step 2 Configure the client parameters of the Flume role.

1. Use Flume on FusionInsight Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager. Choose **Cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **client**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
Use SpoolDir Source, File Channel, and Avro Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by referring to [Table 7-32](#) based on the actual environment.

 NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory/fusioninsight-flume-1.9.0/conf/properties.properties* to ensure that the configuration is in concordance with the previous. Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-32 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
spoolDir	Specifies the directory where the file to be collected resides. This parameter cannot be left blank. The directory needs to exist and have the write, read, and execute permissions on the flume running user.	/srv/BigData/hadoop/data1/zb
trackerDir	Specifies the path for storing the metadata of files collected by Flume.	/srv/BigData/hadoop/data1/tracker
batch-size	Specifies the number of events that Flume sends in a batch.	61200
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/data

Parameter	Description	Example Value
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hostname	Specifies the name or IP address of the host whose data is to be sent. This parameter cannot be left blank. Name or IP address must be configured to be the name or IP address that the Avro source associated with it.	192.168.108.11

Parameter	Description	Example Value
port	Specifies the IP address to which Avro Sink is bound. This parameter cannot be left blank. It must be consistent with the port that is monitored by the connected Avro Source.	21154
ssl	<p>Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.)</p> <p>Only Sources of the Avro type have this configuration item.</p> <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	false

2. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.

Step 3 Configure the server parameters of the Flume role and upload the configuration file to the cluster.

1. Use the Flume configuration tool on the FusionInsight Manager portal to configure the server parameters and generate the configuration file.
 - a. Log in to FusionInsight Manager. Choose **Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **server**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Avro Source, File Channel, and HDFS Sink.
 - c. Double-click the source, channel, and sink. Refer to **Table 7-33** to set corresponding configuration parameters based on the actual environment.

 NOTE

- If the server parameters of the Flume role have been configured, you can choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Instance** on FusionInsight Manager. Then select the corresponding Flume role instance and click the **Download** button behind the **flume.config.file** parameter on the **Instance Configurations** page to obtain the existing server parameter configuration file. Choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Configuration Tool** > **Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
 - A unique checkpoint directory needs to be configured for each File Channel.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-33 Parameters to be modified of the Flume role server

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
bind	Specifies the IP address to which Avro Source is bound. This parameter cannot be left blank. It must be configured as the IP address that the server configuration file will upload.	192.168.108.11
port	Specifies the ID of the port that the Avro Source monitors. This parameter cannot be left blank. It must be configured as an unused port.	21154
ssl	Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) Only Sources of the Avro type have this configuration item. <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	false

Parameter	Description	Example Value
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/data
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hdfs.path	Specifies the HDFS data write directory. This parameter cannot be left blank.	hdfs://hacluster/flume/test
hdfs.inUsePrefix	Specifies the prefix of the file that is being written to HDFS.	TMP_
hdfs.batchSize	Specifies the maximum number of events that can be written to HDFS once.	61200

Parameter	Description	Example Value
hdfs.kerberosPrincipal	Specifies the Kerberos authentication user, which is mandatory in security versions. This configuration is required only in security clusters.	flume_hdfs
hdfs.kerberosKeytab	Specifies the keytab file path for Kerberos authentication, which is mandatory in security versions. This configuration is required only in security clusters.	/opt/test/conf/user.keytab NOTE Obtain the user.keytab file from the Kerberos certificate file of the user flume_hdfs . In addition, ensure that the user who installs and runs the Flume client has the read and write permissions on the user.keytab file.
hdfs.useLocalTimeStamp	Specifies whether to use the local time. Possible values are true and false .	true

2. Log in to FusionInsight Manager and choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume**. On the displayed page, click the **Flume** role under **Role**.
3. Select the Flume role of the node where the configuration file is to be uploaded, choose **Instance Configurations** > **Import** beside the **flume.config.file**, and select the **properties.properties** file.

 **NOTE**

- An independent server configuration file can be uploaded to each Flume instance.
 - This step is required for updating the configuration file. Modifying the configuration file on the background is an improper operation because the modification will be overwritten after configuration synchronization.
4. Click **Save**, and then click **OK**.
 5. Click **Finish**.

Step 4 Verify log transmission.

1. Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS**, click the HDFS WebUI link next to **NameNode (Active)** to go to the HDFS WebUI, and choose **Utilities** > **Browse the file system**.
2. Check whether the data is generated in the **/flume/test** directory on the HDFS.

----End

7.10.4 Typical Scenario: Collecting Local Dynamic Logs and Uploading Them to HDFS

Scenario

This section describes how to use Flume to collect dynamic logs from a local PC (for example, the service IP address is 192.168.108.11) and save them to the `/flume/test` directory on HDFS.

This section applies to MRS 3.x or later.

NOTE

By default, the cluster network environment is secure and the SSL authentication is not enabled during the data transmission process. For details about how to use the encryption mode, see [Configuring the Encrypted Transmission](#). This configuration can use only one Flume scenario, for example, Server:Taildir Source+File Channel+HDFS Sink.

Prerequisites

- The cluster, HDFS, and Flume service have been installed.
- The network environment of the cluster is secure.
- You have created user `flume_hdfs` and authorized the HDFS directory and data to be operated during log verification.

Procedure

Step 1 On FusionInsight Manager, choose **System > User** and choose **More > Download Authentication Credential** to download the Kerberos certificate file of user `flume_hdfs` and save it to the local host.

Step 2 Configure the client parameters of the Flume role.

1. Use the Flume configuration tool on FusionInsight Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to `client`. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Taildir Source, File Channel, and Avro Sink.
 - c. Double-click the source, channel, and sink. Refer to [Table 7-34](#) to set corresponding configuration parameters based on the actual environment.

 NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory/fusioninsight-flume-1.9.0/conf/properties.properties* to ensure that the configuration is in concordance with the previous. Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-34 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
filegroups	Specifies the file group list name. This parameter cannot be left blank. Values are separated by spaces	epgtest
positionFile	Specifies the location where the collected file information (file name and location from which the file collected) is saved. This parameter cannot be left blank. The file does not need to be created manually, but the Flume running user needs to have the write permission on its upper-level directory.	/home/omm/flume/ positionfile
batch-size	Specifies the number of events that Flume sends in a batch.	61200

Parameter	Description	Example Value
dataDirs	<p>Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.</p>	/srv/BigData/hadoop/data1/flume/data
checkpointDir	<p>Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.</p>	/srv/BigData/hadoop/data1/flume/checkpoint

Parameter	Description	Example Value
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hostname	Specifies the name or IP address of the host whose data is to be sent. This parameter cannot be left blank. Name or IP address must be configured to be the name or IP address that the Avro source associated with it.	192.168.108.11
port	Specifies the IP address to which Avro Sink is bound. This parameter cannot be left blank. It must be consistent with the port that is monitored by the connected Avro Source.	21154

Parameter	Description	Example Value
ssl	<p>Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.)</p> <p>Only Sources of the Avro type have this configuration item.</p> <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	false

2. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.

Step 3 Configure the server parameters of the Flume role and upload the configuration file to the cluster.

1. Use the Flume configuration tool on the FusionInsight Manager portal to configure the server parameters and generate the configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **server**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Avro Source, File Channel, and HDFS Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing **Table 7-35** based on the actual environment.

 NOTE

- If the server parameters of the Flume role have been configured, you can choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Instance** on FusionInsight Manager. Then select the corresponding Flume role instance and click the **Download** button behind the **flume.config.file** parameter on the **Instance Configurations** page to obtain the existing server parameter configuration file. Choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Configuration Tool** > **Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
 - A unique checkpoint directory needs to be configured for each File Channel.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-35 Parameters to be modified of the Flume role server

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
bind	Specifies the IP address to which Avro Source is bound. This parameter cannot be left blank. It must be configured as the IP address that the server configuration file will upload.	192.168.108.11
port	Specifies the ID of the port that the Avro Source monitors. This parameter cannot be left blank. It must be configured as an unused port.	21154
ssl	Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) Only Sources of the Avro type have this configuration item. <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	false

Parameter	Description	Example Value
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/data
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hdfs.path	Specifies the HDFS data write directory. This parameter cannot be left blank.	hdfs://hacluster/flume/test
hdfs.inUsePrefix	Specifies the prefix of the file that is being written to HDFS.	TMP_
hdfs.batchSize	Specifies the maximum number of events that can be written to HDFS once.	61200

Parameter	Description	Example Value
hdfs.kerberosPrincipal	Specifies the Kerberos authentication user, which is mandatory in security versions. This configuration is required only in security clusters.	flume_hdfs
hdfs.kerberosKeytab	Specifies the keytab file path for Kerberos authentication, which is mandatory in security versions. This configuration is required only in security clusters.	/opt/test/conf/user.keytab NOTE Obtain the user.keytab file from the Kerberos certificate file of the user flume_hdfs . In addition, ensure that the user who installs and runs the Flume client has the read and write permissions on the user.keytab file.
hdfs.useLocalTimestamp	Specifies whether to use the local time. Possible values are true and false .	true

2. Log in to FusionInsight Manager and choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume**. On the displayed page, click the **Flume** role in the **Role** column.
3. Select the Flume role of the node where the configuration file is to be uploaded, choose **Instance Configurations** > **Import** beside the **flume.config.file**, and select the **properties.properties** file.

 **NOTE**

- An independent server configuration file can be uploaded to each Flume instance.
 - This step is required for updating the configuration file. Modifying the configuration file on the background is an improper operation because the modification will be overwritten after configuration synchronization.
4. Click **Save**, and then click **OK**.
 5. Click **Finish**.

Step 4 Verify log transmission.

1. Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS**, click the HDFS web UI link of **NameNode**(*Node name, Active*) to go to the HDFS web UI, and choose **Utilities** > **Browse the file system**.
2. Check whether the data is generated in the **/flume/test** directory on the HDFS.

----End

7.10.5 Typical Scenario: Collecting Logs from Kafka and Uploading Them to HDFS

Scenario

This section describes how to use Flume to collect logs from the Topic list (test1) of Kafka and save them to the `/flume/test` directory on HDFS.

This section applies to MRS 3.x or later.

NOTE

By default, the cluster network environment is secure and the SSL authentication is not enabled during the data transmission process. For details about how to use the encryption mode, see [Configuring the Encrypted Transmission](#). This configuration can use only one Flume scenario, for example, Server:Kafka Source+File Channel+HDFS Sink.

Prerequisites

- The cluster, HDFS, Kafka, and Flume service have been installed.
- The network environment of the cluster is secure.
- You have created user `flume_hdfs` and authorized the HDFS directory and data to be operated during log verification.

Procedure

Step 1 On FusionInsight Manager, choose **System > User** and choose **More > Download Authentication Credential** to download the Kerberos certificate file of user `flume_hdfs` and save it to the local host.

Step 2 Configure the client parameters of the Flume role.

1. Use the Flume configuration tool on FusionInsight Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to `client`. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Kafka Source, File Channel, and Avro Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing [Table 7-36](#) based on the actual environment.

 NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory/fusioninsight-flume-1.9.0/conf/properties.properties* to ensure that the configuration is in concordance with the previous. Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-36 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
kafka.topics	Specifies the subscribed Kafka topic list, in which topics are separated by commas (.). This parameter cannot be left blank.	test1
kafka.consumer.group.id	Specifies the data group ID obtained from Kafka. This parameter cannot be left blank.	flume
kafka.bootstrap.servers	Specifies the bootstrap IP address and port list of Kafka. The default value is all Kafka lists in a Kafka cluster. If Kafka has been installed in the cluster and its configurations have been synchronized, this parameter can be left blank.	192.168.101.10:9092

Parameter	Description	Example Value
batchSize	Specifies the number of events that Flume sends in a batch (number of data pieces).	61200
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/data
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/checkpoint

Parameter	Description	Example Value
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hostname	Specifies the name or IP address of the host whose data is to be sent. This parameter cannot be left blank. Name or IP address must be configured to be the name or IP address that the Avro source associated with it.	192.168.108.11
port	Specifies the IP address to which Avro Sink is bound. This parameter cannot be left blank. It must be consistent with the port that is monitored by the connected Avro Source.	21154

Parameter	Description	Example Value
ssl	<p>Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.)</p> <p>Only Sources of the Avro type have this configuration item.</p> <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	false

2. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.

Step 3 Configure the server parameters of the Flume role and upload the configuration file to the cluster.

1. Use the Flume configuration tool on the FusionInsight Manager portal to configure the server parameters and generate the configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **server**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Avro Source, File Channel, and HDFS Sink.
 - c. Double-click the source, channel, and sink. Refer to **Table 7-37** to set corresponding configuration parameters based on the actual environment.

 NOTE

- If the server parameters of the Flume role have been configured, you can choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Instance** on FusionInsight Manager. Then select the corresponding Flume role instance and click the **Download** button behind the **flume.config.file** parameter on the **Instance Configurations** page to obtain the existing server parameter configuration file. Choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Configurations** > **Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
 - A unique checkpoint directory needs to be configured for each File Channel.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-37 Parameters to be modified of the Flume role server

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
bind	Specifies the IP address to which Avro Source is bound. This parameter cannot be left blank. It must be configured as the IP address that the server configuration file will upload.	192.168.108.11
port	Specifies the ID of the port that the Avro Source monitors. This parameter cannot be left blank. It must be configured as an unused port.	21154
ssl	Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) Only Sources of the Avro type have this configuration item. <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	false

Parameter	Description	Example Value
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/data
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hdfs.path	Specifies the HDFS data write directory. This parameter cannot be left blank.	hdfs://hacluster/flume/test
hdfs.inUsePrefix	Specifies the prefix of the file that is being written to HDFS.	TMP_
hdfs.batchSize	Specifies the maximum number of events that can be written to HDFS once.	61200

Parameter	Description	Example Value
hdfs.kerberosPrincipal	Specifies the Kerberos authentication user, which is mandatory in security versions. This configuration is required only in security clusters.	flume_hdfs
hdfs.kerberosKeytab	Specifies the keytab file path for Kerberos authentication, which is mandatory in security versions. This configuration is required only in security clusters.	/opt/test/conf/user.keytab NOTE Obtain the user.keytab file from the Kerberos certificate file of the user flume_hdfs . In addition, ensure that the user who installs and runs the Flume client has the read and write permissions on the user.keytab file.
hdfs.useLocalTimeStamp	Specifies whether to use the local time. Possible values are true and false .	true

2. Log in to FusionInsight Manager and choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume**. On the displayed page, click the **Flume** role under **Role**.
3. Select the Flume role of the node where the configuration file is to be uploaded, choose **Instance Configurations** > **Import** beside the **flume.config.file**, and select the **properties.properties** file.

 **NOTE**

- An independent server configuration file can be uploaded to each Flume instance.
 - This step is required for updating the configuration file. Modifying the configuration file on the background is an improper operation because the modification will be overwritten after configuration synchronization.
4. Click **Save**, and then click **OK**.
 5. Click **Finish**.

Step 4 Verify log transmission.

1. Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS**, click the HDFS web UI link of **NameNode**(*Node name, Active*) to go to the HDFS web UI, and choose **Utilities** > **Browse the file system**.
2. Check whether the data is generated in the **/flume/test** directory on the HDFS.

----End

7.10.6 Typical Scenario: Collecting Logs from Kafka and Uploading Them to HDFS Through the Flume Client

Scenario

This section describes how to use Flume to collect logs from the Topic list (test1) of Kafka client and save them to the `/flume/test` directory on HDFS.

This section applies to MRS 3.x or later.

NOTE

By default, the cluster network environment is secure and the SSL authentication is not enabled during the data transmission process. For details about how to use the encryption mode, see [Configuring the Encrypted Transmission](#).

Prerequisites

- The cluster, HDFS, Kafka, and Flume service have been installed.
- You have created user `flume_hdfs` and authorized the HDFS directory and data to be operated during log verification.
- The network environment of the cluster is secure.

Procedure

Step 1 On FusionInsight Manager, choose **System > User** and choose **More > Download Authentication Credential** to download the Kerberos certificate file of user `flume_hdfs` and save it to the local host.

Step 2 Configure the client parameters of the Flume role.

1. Use the Flume configuration tool on FusionInsight Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to `client`. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Kafka Source, File Channel, and HDFS Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing [Table 7-38](#) based on the actual environment.

 NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory/fusioninsight-flume-1.9.0/conf/properties.properties* to ensure that the configuration is in concordance with the previous. Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-38 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
kafka.topics	Specifies the subscribed Kafka topic list, in which topics are separated by commas (.). This parameter cannot be left blank.	test1
kafka.consumer.group.id	Specifies the data group ID obtained from Kafka. This parameter cannot be left blank.	flume
kafka.bootstrap.servers	Specifies the bootstrap IP address and port list of Kafka. The default value is all Kafka lists in a Kafka cluster. If Kafka has been installed in the cluster and its configurations have been synchronized, this parameter can be left blank.	192.168.101.10:21007

Parameter	Description	Example Value
batchSize	Specifies the number of events that Flume sends in a batch (number of data pieces).	61200
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/data
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/checkpoint

Parameter	Description	Example Value
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hdfs.path	Specifies the HDFS data write directory. This parameter cannot be left blank.	hdfs://hacluster/flume/test
hdfs.inUsePrefix	Specifies the prefix of the file that is being written to HDFS.	TMP_
hdfs.batchSize	Specifies the maximum number of events that can be written to HDFS once.	61200
hdfs.kerberosPrincipal	Specifies the Kerberos authentication user, which is mandatory in security versions. This configuration is required only in security clusters.	flume_hdfs
hdfs.kerberosKeytab	Specifies the keytab file path for Kerberos authentication, which is mandatory in security versions. This configuration is required only in security clusters.	/opt/test/conf/user.keytab NOTE Obtain the user.keytab file from the Kerberos certificate file of the user flume_hdfs . In addition, ensure that the user who installs and runs the Flume client has the read and write permissions on the user.keytab file.
hdfs.useLocalTimeStamp	Specifies whether to use the local time. Possible values are true and false .	true

2. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.
3. To connect the Flume client to the HDFS, you need to add the following configuration:
 - a. Download the Kerberos certificate of account **flume_hdfs** and obtain the **krb5.conf** configuration file. Upload the configuration file to the **fusioninsight-flume-1.9.0/conf/** directory on the node where the client is installed.
 - b. In **fusioninsight-flume-1.9.0/conf/**, create the **jaas.conf** configuration file.
vi jaas.conf

```
KafkaClient {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/opt/test/conf/user.keytab"
principal="flume_hdfs@<System domain name>"
useTicketCache=false
storeKey=true
debug=true;
};
```

Values of **keyTab** and **principal** vary depending on the actual situation.
 - c. Obtain configuration files **core-site.xml** and **hdfs-site.xml** from **/opt/FusionInsight_Cluster_<Cluster ID>_Flume_ClientConfig/Flume/config** and upload them to **fusioninsight-flume-1.9.0/conf/**.
4. Restart the Flume service.

Step 3 Verify log transmission.

1. Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS**, click the HDFS WebUI link of **NameNode** (*Node name, Active*) to go to the HDFS WebUI, and choose **Utilities** > **Browse the file system**.
2. Check whether the data is generated in the **/flume/test** directory on the HDFS.

----End

7.10.7 Typical Scenario: Collecting Local Static Logs and Uploading Them to HBase

Scenario

This section describes how to use Flume to collect static logs from a local computer (service IP address: 192.168.108.11) and upload them to the **flume_test** table of HBase.

This section applies to MRS 3.x or later.

 NOTE

By default, the cluster network environment is secure and the SSL authentication is not enabled during the data transmission process. For details about how to use the encryption mode, see [Configuring the Encrypted Transmission](#). The configuration can apply to scenarios where only the client or the server is configured, for example, Client/Server:SpoolDir Source+File Channel+HBase Sink.

Prerequisites

- The cluster, HBase, and Flume service have been installed.
- The network environment of the cluster is secure.
- An HBase table has been created by running the **create 'flume_test', 'cf'** command.
- You have understood service requirements and prepared HBase administrator **flume_hbase**.

Procedure

Step 1 On FusionInsight Manager, choose **System > User** and choose **More > Download Authentication Credential** to download the Kerberos certificate file of user **flume_hbase** and save it to the local host.

Step 2 Configure the client parameters of the Flume role.

1. Use the Flume configuration tool on FusionInsight Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **client**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
Use SpoolDir Source, File Channel, and Avro Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing [Table 7-39](#) based on the actual environment.

 NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory*/fusioninsight-flume-1.9.0/conf/properties.properties to ensure that the configuration is in concordance with the previous. Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool > Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-39 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
spoolDir	Specifies the directory where the file to be collected resides. This parameter cannot be left blank. The directory needs to exist and have the write, read, and execute permissions on the flume running user.	/srv/BigData/hadoop/data1/zb
trackerDir	Specifies the path for storing the metadata of files collected by Flume.	/srv/BigData/hadoop/data1/tracker
batchSize	Specifies the number of events that Flume sends in a batch (number of data pieces). A larger value indicates higher performance and lower timeliness.	61200

Parameter	Description	Example Value
dataDirs	<p>Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.</p>	/srv/BigData/hadoop/data1/flume/data
checkpointDir	<p>Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.</p>	/srv/BigData/hadoop/data1/flume/checkpoint

Parameter	Description	Example Value
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hostname	Specifies the name or IP address of the host whose data is to be sent. This parameter cannot be left blank. Name or IP address must be configured to be the name or IP address that the Avro source associated with it.	192.168.108.11
port	Specifies the port that sends the data. This parameter cannot be left blank. It must be consistent with the port that is monitored by the connected Avro Source.	21154

Parameter	Description	Example Value
ssl	<p>Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.)</p> <p>Only Sources of the Avro type have this configuration item.</p> <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	false

2. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.

Step 3 Configure the server parameters of the Flume role and upload the configuration file to the cluster.

1. Use the Flume configuration tool on the FusionInsight Manager portal to configure the server parameters and generate the configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **server**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Avro Source, File Channel, and HBase Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing **Table 7-40** based on the actual environment.

 NOTE

- If the server parameters of the Flume role have been configured, you can choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Instance** on FusionInsight Manager. Then select the corresponding Flume role instance and click the **Download** button behind the **flume.config.file** parameter on the **Instance Configurations** page to obtain the existing server parameter configuration file. Choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Configuration Tool** > **Import**, import the file, and modify the configuration items related to non-encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
 - A unique checkpoint directory needs to be configured for each File Channel.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-40 Parameters to be modified of the Flume role server

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
bind	Specifies the IP address to which Avro Source is bound. This parameter cannot be left blank. It must be configured as the IP address that the server configuration file will upload.	192.168.108.11
port	Specifies the ID of the port that the Avro Source monitors. This parameter cannot be left blank. It must be configured as an unused port.	21154
ssl	Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) Only Sources of the Avro type have this configuration item. <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	false

Parameter	Description	Example Value
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/data
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
table	Specifies the HBase table name. This parameter cannot be left blank.	flume_test
columnFamily	Specifies the HBase column family name. This parameter cannot be left blank.	cf
batchSize	Specifies the maximum number of events written to HBase by Flume in a batch.	61200

Parameter	Description	Example Value
kerberosPrincipal	Specifies the Kerberos authentication user, which is mandatory in security versions. This configuration is required only in security clusters.	flume_hbase
kerberosKeytab	Specifies the file path for Kerberos authentication, which is mandatory in security versions. This configuration is required only in security clusters.	/opt/test/conf/user.keytab NOTE Obtain the user.keytab file from the Kerberos certificate file of the user flume_hbase . In addition, ensure that the user who installs and runs the Flume client has the read and write permissions on the user.keytab file.

2. Log in to FusionInsight Manager and choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume**. On the displayed page, click the **Flume** role on the **Instance** tab page.
3. Select the Flume role of the node where the configuration file is to be uploaded, choose **Instance Configurations** > **Import** beside the **flume.config.file**, and select the **properties.properties** file.

 **NOTE**

- An independent server configuration file can be uploaded to each Flume instance.
 - This step is required for updating the configuration file. Modifying the configuration file on the background is an improper operation because the modification will be overwritten after configuration synchronization.
4. Click **Save**, and then click **OK**.
 5. Click **Finish**.

Step 4 Verify log transmission.

1. Go to the directory where the HBase client is installed.
cd /Client installation directory/ HBase/hbase
kinit flume_hbase (Enter the password.)
2. Run the **hbase shell** command to access the HBase client.
3. Run the **scan 'flume_test'** statement. Logs are written in the HBase column family by line.

```
hbase(main):001:0> scan 'flume_test'
ROW                                COLUMN
+CELL
```

```
2017-09-18 16:05:36,394 INFO [hconnection-0x415a3f6a-shared--pool2-t1] ipc.AbstractRpcClient:
RPC Server Kerberos principal name for service=ClientService is hbase/hadoop.<system domain
name>@<system domain name>
```

```
default4021ff4a-9339-4151-a4d0-00f20807e76d      column=cf:pCol,  
timestamp=1505721909388, value=Welcome to  
flume  
incRow                                           column=cf:iCol, timestamp=1505721909461, value=  
\x00\x00\x00\x00\x00\x00\x00\x00\x01  
2 row(s) in 0.3660 seconds
```

----End

7.11 Encrypted Transmission

7.11.1 Configuring the Encrypted Transmission

Scenario

This section describes how to configure the server and client parameters of the Flume service (including the Flume and MonitorServer roles) after the cluster is installed to ensure proper running of the service.

This section applies to MRS 3.x or later.

Prerequisites

The cluster and Flume service have been installed.

Procedure

Step 1 Generate the certificate trust lists of the server and client of the Flume role respectively.

1. Remotely log in to the node using ECM where the Flume server is to be installed as user **omm**. Go to the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin** directory.

```
cd `${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin
```

NOTE

The version 8.0.2.1 is used as an example. Replace it with the actual version number.

2. Run the following command to generate and export the server and client certificates of the Flume role:

```
sh geneJKS.sh -f sNetty12@ -g cNetty12@
```

The generated certificate is saved in the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf** path .

- **flume_sChat.jks** is the certificate library of the Flume role server.
flume_sChat.crt is the exported file of the **flume_sChat.jks** certificate. **-f** indicates the password of the certificate and certificate library.
- **flume_cChat.jks** is the certificate library of the Flume role client.
flume_cChat.crt is the exported file of the **flume_cChat.jks** certificate. **-g** indicates the password of the certificate and certificate library.

- **flume_sChatt.jks** and **flume_cChatt.jks** are the SSL certificate trust lists of the Flume server and client, respectively.

 **NOTE**

All user-defined passwords involved in this section (such as *sNetty12@*) must meet the following requirements:

- The password must contain at least four types of uppercase letters, lowercase letters, digits, and special characters.
- The password must contain 8 to 64 characters.
- It is recommended that the user-defined passwords be changed periodically (for example, every three months), and certificates and trust lists be generated again to ensure security.

Step 2 Configure the server parameters of the Flume role and upload the configuration file to the cluster.

1. Remotely log in to any node where the Flume role is located as user **omm** using ECM. Run the following command to go to the `${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin` directory:

```
cd ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin
```

2. Run the following command to generate and obtain Flume server keystore password, trust list password, and keystore-password encrypted private key information. Enter the password twice and confirm the password. It is the password of the **flume_sChat.jks** certificate library, for example, *sNetty12@*.

```
./genPwFile.sh
```

```
cat password.property
```

```
password=D03C2D03D97CBA3F4FD2491A40CAA5E0
```

3. Use the Flume configuration tool on the FusionInsight Manager portal to configure the server parameters and generate the configuration file.
 - a. Log in to FusionInsight Manager. Choose **Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **server**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Avro Source, File Channel, and HDFS Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing [Table 7-41](#) based on the actual environment.

 **NOTE**

- If the server parameters of the Flume role have been configured, you can choose **Services > Flume > Instance** on FusionInsight Manager. Then select the corresponding Flume role instance and click the **Download** button behind the **flume.config.file** parameter on the **Instance Configurations** page to obtain the existing server parameter configuration file. Choose **Services > Flume > Import** to change the relevant configuration items of encrypted transmission after the file is imported.
- It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.

- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-41 Parameters to be modified of the Flume role server

Parameter	Description	Example Value
ssl	Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	true
keystore	Indicates the server certificate.	\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/flume_sChat.jks
keystore-password	Specifies the password of the key library, which is the password required to obtain the keystore information. Enter the value of password obtained in Step 2.2 .	D03C2D03D97CBA3F4FD2491A40CAA5E0
truststore	Indicates the SSL certificate trust list of the server.	\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/flume_sChatt.jks
truststore-password	Specifies the trust list password, which is the password required to obtain the truststore information. Enter the value of password obtained in Step 2.2 .	D03C2D03D97CBA3F4FD2491A40CAA5E0

4. Log in to FusionInsight Manager and choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume**. On the displayed page, click the **Flume** role under **Role**.
5. Select the Flume role of the node where the configuration file is to be uploaded, choose **Instance Configurations** > **Import** beside the **flume.config.file**, and select the **properties.properties** file.

 **NOTE**

- An independent server configuration file can be uploaded to each Flume instance.
- This step is required for updating the configuration file. Modifying the configuration file on the background is an improper operation because the modification will be overwritten after configuration synchronization.

6. Click **Save**, and then click **OK**. Click **Finish**.

Step 3 Set the client parameters of the Flume role.

1. Run the following commands to copy the generated client certificate (**flume_cChat.jks**) and client trust list (**flume_cChatt.jks**) to the client directory, for example, **/opt/flume-client/fusionInsight-flume-1.9.0/conf/**. (The Flume client must have been installed.) **10.196.26.1** is the service plane IP address of the node where the client resides.

```
scp ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/flume_cChat.jks user@10.196.26.1:/opt/flume-client/fusionInsight-flume-1.9.0/conf/
```

```
scp ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/flume_cChatt.jks user@10.196.26.1:/opt/flume-client/fusionInsight-flume-1.9.0/conf/
```

 **NOTE**

When copying the client certificate, you need to enter the password of user **user** of the host (for example, **10.196.26.1**) where the client resides.

2. Log in to the node where the Flume client is decompressed as user **user**. Run the following command to go to the client directory **opt/flume-client/fusionInsight-flume-1.9.0/bin**.

```
cd opt/flume-client/fusionInsight-flume-1.9.0/bin
```

3. Run the following command to generate and obtain Flume client keystore password, trust list password, and keystore-password encrypted private key information. Enter the password twice and confirm the password. The password is the same as the password of the certificate whose alias is *flumechatclient* and the password of the *flume_cChat.jks* certificate library, for example *cNetty12@*.

```
./genPwFile.sh
```

```
cat password.property
```

```
password=4FD2491A40CAA5E0D03C2D03D97CBA3F
```

 **NOTE**

If the following error message is displayed, run the export **JAVA_HOME=JDK path** command.

```
JAVA_HOME is null in current user,please install the JDK and set the JAVA_HOME
```

4. Use the Flume configuration tool on FusionInsight Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **client**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use SpoolDir Source, File Channel, and Avro Sink.

- c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing [Table 7-42](#) based on the actual environment.

 NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory/fusioninsight-flume-1.9.0/conf/properties.properties* to ensure that the configuration is in concordance with the previous. Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool > Import**, import the file, and modify the configuration items related to encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
 - A unique checkpoint directory needs to be configured for each File Channel.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-42 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
ssl	Indicates whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	true
keystore	Specified the client certificate.	/opt/flume-client/fusionInsight-flume-1.9.0/conf/flume_cChat.jks

Parameter	Description	Example Value
keystore-password	Specifies the password of the key library, which is the password required to obtain the keystore information. Enter the value of password obtained in Step 3.3 .	4FD2491A40CAA5E0D 03C2D03D97CBA3F
truststore	Indicates the SSL certificate trust list of the client.	/opt/flume-client/ fusionInsight- flume-1.9.0/conf/ flume_cChat.jks
truststore-password	Specifies the trust list password, which is the password required to obtain the truststore information. Enter the value of password obtained in Step 3.3 .	4FD2491A40CAA5E0D 03C2D03D97CBA3F

5. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.

Step 4 Generate the certificate and trust list of the server and client of the MonitorServer role respectively.

1. Log in to the host using ECM with the MonitorServer role assigned as user **omm**.

Go to the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin** directory.

```
cd `${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin
```

2. Run the following command to generate and export the server and client certificates of the MonitorServer role:

```
sh geneJKS.sh -m sKitty12@ -n cKitty12@
```

The generated certificate is saved in the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf** path. Where:

- **ms_sChat.jks** is the certificate library of the MonitorServer role server. **ms_sChat.crt** is the exported file of the **ms_sChat.jks** certificate. **-m** indicates the password of the certificate and certificate library.
- **ms_cChat.jks** is the certificate library of the MonitorServer role client. **ms_cChat.crt** is the exported file of the **ms_cChat.jks** certificate. **-n** indicates the password of the certificate and certificate library.

- **ms_sChatt.jks** and **ms_cChatt.jks** are the SSL certificate trust lists of the MonitorServer server and client, respectively.

Step 5 Set the server parameters of the MonitorServer role.

1. Run the following command to generate and obtain MonitorServer server keystore password, trust list password, and keystore-password encrypted private key information. Enter the password twice and confirm the password. The password is the same as the password of the certificate whose alias is *mschatserver* and the password of the *ms_sChat.jks* certificate library, for example *sKitty12@*.

```
./genPwFile.sh
```

```
cat password.property
```

```
password=AA5E0D03C2D4FD24CBA3F91A40C03D97
```

2. Run the following command to open the `${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/service/application.properties` file: Modify related parameters based on the description in [Table 7-43](#), save the modification, and exit.

```
vi ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/service/application.properties
```

Table 7-43 Parameters to be modified of the MonitorServer role server

Parameter	Description	Example Value
ssl_need_kspas swd_decrypt_k ey	Specifies whether to enable the user-defined key encryption and decryption function. (You are advised to enable this function to ensure security.) - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled.	true
ssl_server_enab le	Indicates whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled.	true

Parameter	Description	Example Value
ssl_server_key_store	Set this parameter based on the specific storage location.	\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_sChat.jks
ssl_server_trust_key_store	Set this parameter based on the specific storage location.	\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_sChatt.jks
ssl_server_key_store_password	Indicates the client certificate password. Set this parameter based on the actual situation of certificate creation (the plaintext key used to generate the certificate). Enter the value of password obtained in Step 5.1 .	AA5E0D03C2D4FD24CBA3F91A40C03D97
ssl_server_trust_key_store_password	Specifies the trustkeystore password. Set this parameter based on the actual situation of certificate creation (the plaintext key used to generate the trust list). Enter the value of password obtained in Step 5.1 .	AA5E0D03C2D4FD24CBA3F91A40C03D97
ssl_need_client_auth	Indicates whether to enable the client authentication. (You are advised to enable this function to ensure security.) <ul style="list-style-type: none"> - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled. 	true

- Restart the MonitorServer instance. Choose **Services > Flume > Instance > MonitorServer**, select the MonitorServer instance, and choose **More > Restart Instance**. Enter password and click **OK**. After the restart is complete, click **Finish**.

Step 6 Set the client parameters of the MonitorServer role.

1. Run the following commands to copy the generated client certificate (**ms_cChat.jks**) and client trust list (**ms_cChatt.jks**) to the **/opt/flume-client/fusionInsight-flume-1.9.0/conf/** client directory. **10.196.26.1** is the service plane IP address of the node where the client resides.

```
scp ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_cChat.jks user@10.196.26.1:/opt/flume-client/fusionInsight-flume-1.9.0/conf/
```

```
scp ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_cChatt.jks user@10.196.26.1:/opt/flume-client/fusionInsight-flume-1.9.0/conf/
```

2. Log in to the node where the Flume client is located as **user**. Run the following command to go to the client directory **/opt/flume-client/fusionInsight-flume-1.9.0/bin**.

```
cd /opt/flume-client/fusionInsight-flume-1.9.0/bin
```

3. Run the following command to generate and obtain MonitorServer client keystore password, trust list password, and keystore-password encrypted private key information. Enter the password twice and confirm the password. The password is the same as the password of the certificate whose alias is *mschatclient* and the password of the *ms_cChat.jks* certificate library, for example *cKitty12@*.

```
./genPwFile.sh
```

```
cat password.property
```

```
password=BA3F91A40C03D97AA5E0D03C2D4FD24C
```

4. Run the following command to open the **/opt/flume-client/fusionInsight-flume-1.9.0/conf/service/application.properties** file. (**/opt/flume-client/fusionInsight-flume-1.9.0** is the directory where the client software is installed.) Modify related parameters based on the description in [Table 7-44](#), save the modification, and exit.

```
vi /opt/flume-client/fusionInsight-flume-1.9.0/flume/conf/service/application.properties
```

Table 7-44 Parameters to be modified of the MonitorServer role client

Parameter	Description	Example Value
ssl_need_kspas swd_decrypt_key	Indicates whether to enable the user-defined key encryption and decryption function. (You are advised to enable this function to ensure security.) <ul style="list-style-type: none"> - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled. 	true

Parameter	Description	Example Value
ssl_client_enable	<p>Indicates whether to enable the SSL authentication. (You are advised to enable this function to ensure security.)</p> <ul style="list-style-type: none"> - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled. 	true
ssl_client_key_store	Set this parameter based on the specific storage location.	<pre> \${BIGDATA_HOME}/ FusionInsight_Porter_8.0.2.1 /install/FusionInsight- Flume-1.9.0/flume/conf/ ms_cChat.jks </pre>
ssl_client_trust_key_store	Set this parameter based on the specific storage location.	<pre> \${BIGDATA_HOME}/ FusionInsight_Porter_8.0.2.1 /install/FusionInsight- Flume-1.9.0/flume/conf/ ms_cChat.jks </pre>
ssl_client_key_store_password	<p>Specifies the keystore password. Set this parameter based on the actual situation of certificate creation (the plaintext key used to generate the certificate).</p> <p>Enter the value of password obtained in Step 6.3.</p>	BA3F91A40C03D97AA5E0D03C2D4FD24C
ssl_client_trust_key_store_password	<p>Specifies the trustkeystore password. Set this parameter based on the actual situation of certificate creation (the plaintext key used to generate the trust list).</p> <p>Enter the value of password obtained in Step 6.3.</p>	BA3F91A40C03D97AA5E0D03C2D4FD24C

Parameter	Description	Example Value
ssl_need_client_auth	<p>Indicates whether to enable the client authentication. (You are advised to enable this function to ensure security.)</p> <ul style="list-style-type: none"> - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled. 	true

----End

7.11.2 Typical Scenario: Collecting Local Static Logs and Uploading Them to HDFS

Scenario

This section describes how to use Flume to collect static logs from a local PC (service IP address: **192.168.108.11**) and save them to the **/flume/test** directory on HDFS.

This section applies to MRS 3.x or later.

Prerequisites

- The cluster, HDFS and Flume services, and Flume client have been installed.
- User **flume_hdfs** has been created, and the HDFS directory and data used for log verification have been authorized to the user.

Procedure

Step 1 Generate the certificate trust lists of the server and client of the Flume role respectively.

1. Log in to the node where the Flume server is located as user **omm**. Go to the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin** directory.

```
cd `${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin
```

2. Run the following command to generate and export the server and client certificates of the Flume role:

```
sh geneJKS.sh -f sNetty12@ -g cNetty12@
```

The generated certificate is saved in the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf** path .

- **flume_sChat.jks** is the certificate library of the Flume role server. **flume_sChat.crt** is the exported file of the **flume_sChat.jks** certificate. **-f** indicates the password of the certificate and certificate library.
- **flume_cChat.jks** is the certificate library of the Flume role client. **flume_cChat.crt** is the exported file of the **flume_cChat.jks** certificate. **-g** indicates the password of the certificate and certificate library.
- **flume_sChatt.jks** and **flume_cChatt.jks** are the SSL certificate trust lists of the Flume server and client, respectively.

 **NOTE**

All user-defined passwords involved in this section (such as *sNetty12@*) must meet the following requirements:

- Contain at least four types of the following: uppercase letters, lowercase letters, digits, and special characters.
- Contain at least eight characters and a maximum of 64 characters.
- It is recommended that the user-defined passwords be changed periodically (for example, every three months), and certificates and trust lists be generated again to ensure security.

Step 2 On FusionInsight Manager, choose **System > User** and choose **More > Download Authentication Credential** to download the Kerberos certificate file of user **flume_hdfs** and save it to the local host.

Step 3 Configure the server parameters of the Flume role and upload the configuration file to the cluster.

1. Log in to any node where the Flume role is located as user **omm**. Run the following command to go to the `/${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin` directory:

```
cd ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin
```

2. Run the following command to generate and obtain Flume server keystore password, trust list password, and keystore-password encrypted private key information. Enter the password twice and confirm the password. It is the password of the **flume_sChat.jks** certificate library, for example, *sNetty12@*.

```
./genPwFile.sh
```

```
cat password.property
```

```
password=D03C2D03D97CBA3F4FD2491A40CAA5E0
```

3. Use the Flume configuration tool on the FusionInsight Manager portal to configure the server parameters and generate the configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **server**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
For example, use Avro Source, File Channel, and HDFS Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing [Table 7-45](#) based on the actual environment.

 NOTE

- If the server parameters of the Flume role have been configured, you can choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Instance** on FusionInsight Manager. Then select the corresponding Flume role instance and click the **Download** button behind the **flume.config.file** parameter on the **Instance Configurations** page to obtain the existing server parameter configuration file. Choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Configuration Tool** > **Import**, import the file, and modify the configuration items related to encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
 - A unique checkpoint directory needs to be configured for each File Channel.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-45 Parameters to be modified of the Flume role server

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
bind	Specifies the IP address to which Avro Source is bound. This parameter cannot be left blank. It must be configured as the IP address that the server configuration file will upload.	192.168.108.11
port	Specifies the IP address to which Avro Source is bound. This parameter cannot be left blank. It must be configured as an unused port.	21154
ssl	Indicates whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) Only Sources of the Avro type have this configuration item. <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	true

Parameter	Description	Example Value
keystore	Indicates the server certificate.	<code>\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/flume_sChat.jks</code>
keystore-password	Specifies the password of the key library, which is the password required to obtain the keystore information. Enter the value of password obtained in Step 3.2 .	D03C2D03D97CBA3 F4FD2491A40CAA5E 0
truststore	Indicates the SSL certificate trust list of the server.	<code>\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/flume_sChatt.jks</code>
truststore-password	Specifies the trust list password, which is the password required to obtain the truststore information. Enter the value of password obtained in Step 3.2 .	D03C2D03D97CBA3 F4FD2491A40CAA5E 0
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	<code>/srv/BigData/hadoop/data1/flumeserver/data</code>

Parameter	Description	Example Value
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flumeserver/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hdfs.path	Specifies the HDFS data write directory. This parameter cannot be left blank.	hdfs://hacluster/flume/test
hdfs.inUsePrefix	Specifies the prefix of the file that is being written to HDFS.	TMP_
hdfs.batchSize	Specifies the maximum number of events that can be written to HDFS once.	61200
hdfs.kerberosPrincipal	Specifies the Kerberos authentication user, which is mandatory in security versions. This configuration is required only in security clusters.	flume_hdfs

Parameter	Description	Example Value
hdfs.kerberosKeytab	Specifies the keytab file path for Kerberos authentication, which is mandatory in security versions. This configuration is required only in security clusters.	/opt/test/conf/user.keytab NOTE Obtain the user.keytab file from the Kerberos certificate file of the user flume_hdfs . In addition, ensure that the user who installs and runs the Flume client has the read and write permissions on the user.keytab file.
hdfs.useLocalTimeStamp	Specifies whether to use the local time. Possible values are true and false .	true

- Log in to FusionInsight Manager and choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume**. On the displayed page, click the **Flume** role under **Role**.
- Select the Flume role of the node where the configuration file is to be uploaded, choose **Instance Configurations** > **Import** beside the **flume.config.file**, and select the **properties.properties** file.

 **NOTE**

- An independent server configuration file can be uploaded to each Flume instance.
 - This step is required for updating the configuration file. Modifying the configuration file on the background is an improper operation because the modification will be overwritten after configuration synchronization.
- Click **Save**, and then click **OK**.
 - Click **Finish**.

Step 4 Configure the client parameters of the Flume role.

- Run the following commands to copy the generated client certificate (**flume_cChat.jks**) and client trust list (**flume_cChatt.jks**) to the client directory, for example, **/opt/flume-client/fusionInsight-flume-1.9.0/conf/**. (The Flume client must have been installed.) **10.196.26.1** is the service plane IP address of the node where the client resides.

```
scp ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/flume_cChat.jks user@10.196.26.1:/opt/flume-client/fusionInsight-flume-1.9.0/conf/
scp ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/flume_cChatt.jks user@10.196.26.1:/opt/flume-client/fusionInsight-flume-1.9.0/conf/
```

 NOTE

When copying the client certificate, you need to enter the password of user **user** of the host (for example, **10.196.26.1**) where the client resides.

2. Log in to the node where the Flume client is decompressed as user **user**. Run the following command to go to the client directory **/opt/flume-client/fusionInsight-flume-1.9.0/bin**.

```
cd opt/flume-client/fusionInsight-flume-1.9.0/bin
```

3. Run the following command to generate and obtain Flume client keystore password, trust list password, and keystore-password encrypted private key information. Enter the password twice and confirm the password. The password is the same as the password of the certificate whose alias is *flumechatclient* and the password of the *flume_cChat.jks* certificate library, for example *cNetty12@*.

```
./genPwFile.sh
```

```
cat password.property
```

```
password=4FD2491A40CAA5E0D03C2D03D97CBA3F
```

 NOTE

If the following error message is displayed, run the export **JAVA_HOME=JDKpath** command.

```
JAVA_HOME is null in current user,please install the JDK and set the JAVA_HOME
```

4. Use the Flume configuration tool on FusionInsight Manager to configure the Flume role client parameters and generate a configuration file.
 - a. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool**.
 - b. Set **Agent Name** to **client**. Select the source, channel, and sink to be used, drag them to the GUI on the right, and connect them.
Use SpoolDir Source, File Channel, and Avro Sink.
 - c. Double-click the source, channel, and sink. Set corresponding configuration parameters by seeing [Table 7-46](#) based on the actual environment.

 NOTE

- If the client parameters of the Flume role have been configured, you can obtain the existing client parameter configuration file from *client installation directory/fusioninsight-flume-1.9.0/conf/properties.properties* to ensure that the configuration is in concordance with the previous. Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Flume > Configuration Tool > Import**, import the file, and modify the configuration items related to encrypted transmission.
 - It is recommended that the numbers of Sources, Channels, and Sinks do not exceed 40 during configuration file import. Otherwise, the response time may be very long.
- d. Click **Export** to save the **properties.properties** configuration file to the local.

Table 7-46 Parameters to be modified of the Flume role client

Parameter	Description	Example Value
Name	The value must be unique and cannot be left blank.	test
spoolDir	Specifies the directory where the file to be collected resides. This parameter cannot be left blank. The directory needs to exist and have the write, read, and execute permissions on the flume running user.	/srv/BigData/hadoop/data1/zb
trackerDir	Specifies the path for storing the metadata of files collected by Flume.	/srv/BigData/hadoop/data1/tracker
batch-size	Specifies the number of events that Flume sends in a batch.	61200
dataDirs	Specifies the directory for storing buffer data. The run directory is used by default. Configuring multiple directories on disks can improve transmission efficiency. Use commas (,) to separate multiple directories. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/data directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/data

Parameter	Description	Example Value
checkpointDir	Specifies the directory for storing the checkpoint information, which is under the run directory by default. If the directory is inside the cluster, the /srv/BigData/hadoop/dataX/flume/checkpoint directory can be used. dataX ranges from data1 to dataN. If the directory is outside the cluster, it needs to be independently planned.	/srv/BigData/hadoop/data1/flume/checkpoint
transactionCapacity	Specifies the transaction size, that is, the number of events in a transaction that can be processed by the current Channel. The size cannot be smaller than the batchSize of Source. Setting the same size as batchSize is recommended.	61200
hostname	Specifies the name or IP address of the host whose data is to be sent. This parameter cannot be left blank. Name or IP address must be configured to be the name or IP address that the Avro source associated with it.	192.168.108.11

Parameter	Description	Example Value
port	Specifies the IP address to which Avro Sink is bound. This parameter cannot be left blank. It must be consistent with the port that is monitored by the connected Avro Source.	21154
ssl	Specifies whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) Only Sources of the Avro type have this configuration item. <ul style="list-style-type: none"> ▪ true indicates that the function is enabled. ▪ false indicates that the client authentication function is not enabled. 	true
keystore	Specifies the flume_cChat.jks certificate generated on the server.	/opt/flume-client/fusionInsight-flume-1.9.0/conf/flume_cChat.jks
keystore-password	Specifies the password of the key library, which is the password required to obtain the keystore information. Enter the value of password obtained in Step 4.3 .	D03C2D03D97CBA3F4 FD2491A40CAA5E0
truststore	Indicates the SSL certificate trust list of the server.	/opt/flume-client/fusionInsight-flume-1.9.0/conf/flume_cChat.jks

Parameter	Description	Example Value
truststore-password	Specifies the trust list password, which is the password required to obtain the truststore information. Enter the value of password obtained in Step 4.3 .	D03C2D03D97CBA3F4 FD2491A40CAA5E0

5. Upload the **properties.properties** file to **flume/conf/** under the installation directory of the Flume client.

Step 5 Generate the certificate and trust list of the server and client of the MonitorServer role respectively.

1. Log in to the host with the MonitorServer role assigned as user **omm**.

Go to the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin** directory.

```
cd `${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/bin
```

2. Run the following command to generate and export the server and client certificates of the MonitorServer role:

```
sh geneJKS.sh -m sKitty12@ -n cKitty12@
```

The generated certificate is saved in the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf** path. Where:

- **ms_sChat.jks** is the certificate library of the MonitorServer role server. **ms_sChat.crt** is the exported file of the **ms_sChat.jks** certificate. **-m** indicates the password of the certificate and certificate library.
- **ms_cChat.jks** is the certificate library of the MonitorServer role client. **ms_cChat.crt** is the exported file of the **ms_cChat.jks** certificate. **-n** indicates the password of the certificate and certificate library.
- **ms_sChatt.jks** and **ms_cChatt.jks** are the SSL certificate trust lists of the MonitorServer server and client, respectively.

Step 6 Set the server parameters of the MonitorServer role.

1. Run the following command to generate and obtain MonitorServer server keystore password, trust list password, and keystore-password encrypted private key information. Enter the password twice and confirm the password. The password is the same as the password of the certificate whose alias is *mschatserver* and the password of the *ms_sChat.jks* certificate library, for example *sKitty12@*.

```
./genPwFile.sh
```

```
cat password.property
```

```
password=AA5E0D03C2D4FD24CBA3F91A40C03D97
```

2. Run the following command to open the **`\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/**

service/application.properties file: Modify related parameters based on the description in [Table 7-47](#), save the modification, and exit.

vi `${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/service/application.properties`

Table 7-47 Parameters to be modified of the MonitorServer role server

Parameter	Description	Example Value
ssl_need_kspas swd_decrypt_k ey	Indicates whether to enable the user-defined key encryption and decryption function. (You are advised to enable this function to ensure security.) <ul style="list-style-type: none"> - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled. 	true
ssl_server_enab le	Indicates whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) <ul style="list-style-type: none"> - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled. 	true
ssl_server_key_ store	Set this parameter based on the specific storage location.	<code>\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_sChat.jks</code>
ssl_server_trust_ _key_store	Set this parameter based on the specific storage location.	<code>\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_sChatt.jks</code>

Parameter	Description	Example Value
ssl_server_key_store_password	Indicates the client certificate password. Set this parameter based on the actual situation of certificate creation (the plaintext key used to generate the certificate). Enter the value of password obtained in Step 6.1 .	AA5E0D03C2D4FD24CBA3F91A40C03D97
ssl_server_trust_key_store_password	Indicates the client trust list password. Set this parameter based on the actual situation of certificate creation (the plaintext key used to generate the trust list). Enter the value of password obtained in Step 6.1 .	AA5E0D03C2D4FD24CBA3F91A40C03D97
ssl_need_client_auth	Indicates whether to enable the client authentication. (You are advised to enable this function to ensure security.) <ul style="list-style-type: none"> - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled. 	true

- Restart the MonitorServer instance. Choose **Cluster > Name of the desired cluster > Services > Flume > Instance > MonitorServer**, select the configured MonitorServer instance, and choose **More > Restart Instance**. Enter password and click **OK**. After the restart is complete, click **Finish**.

Step 7 Set the client parameters of the MonitorServer role.

- Run the following commands to copy the generated client certificate (**ms_cChat.jks**) and client trust list (**ms_cChatt.jks**) to the **/opt/flume-client/fusionInsight-flume-1.9.0/conf/** client directory. **10.196.26.1** is the service plane IP address of the node where the client resides.

```

scp ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_cChat.jks user@10.196.26.1:/opt/flume-client/fusionInsight-flume-1.9.0/conf/
scp ${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_cChatt.jks user@10.196.26.1:/opt/flume-client/fusionInsight-flume-1.9.0/conf/

```

- Log in to the node where the Flume client is located as user **user**. Run the following command to go to the client directory **/opt/flume-client/fusionInsight-flume-1.9.0/bin**.
- Run the following command to generate and obtain MonitorServer client keystore password, trust list password, and keystore-password encrypted private key information. Enter the password twice and confirm the password. The password is the same as the password of the certificate whose alias is *mschatclient* and the password of the *ms_cChat.jks* certificate library, for example *cKitty12@*.

```
./genPwFile.sh
```

```
cat password.property
```

```
password=BA3F91A40C03D97AA5E0D03C2D4FD24C
```

- Run the following command to open the **/opt/flume-client/fusionInsight-flume-1.9.0/conf/service/application.properties** file. (**/opt/flume-client/fusionInsight-flume-1.9.0** is the directory where the client is installed.) Modify related parameters based on the description in [Table 7-48](#), save the modification, and exit.

```
vi /opt/flume-client/fusionInsight-flume-1.9.0/conf/service/application.properties
```

Table 7-48 Parameters to be modified of the MonitorServer role client

Parameter	Description	Example Value
ssl_need_kspas swd_decrypt_k ey	Indicates whether to enable the user-defined key encryption and decryption function. (You are advised to enable this function to ensure security.) - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled.	true
ssl_client_enab le	Indicates whether to enable the SSL authentication. (You are advised to enable this function to ensure security.) - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled.	true

Parameter	Description	Example Value
ssl_client_key_store	Set this parameter based on the specific storage location.	\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_cChat.jks
ssl_client_trust_key_store	Set this parameter based on the specific storage location.	\${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Flume-1.9.0/flume/conf/ms_cChatt.jks
ssl_client_key_store_password	Specifies the keystore password. Set this parameter based on the actual situation of certificate creation (the plaintext key used to generate the certificate). Enter the value of password obtained in Step 7.3 .	BA3F91A40C03D97AA5E0D03C2D4FD24C
ssl_client_trust_key_store_password	Specifies the trustkeystore password. Set this parameter based on the actual situation of certificate creation (the plaintext key used to generate the trust list). Enter the value of password obtained in Step 7.3 .	BA3F91A40C03D97AA5E0D03C2D4FD24C
ssl_need_client_auth	Indicates whether to enable the client authentication. (You are advised to enable this function to ensure security.) <ul style="list-style-type: none"> - true indicates that the function is enabled. - false indicates that the client authentication function is not enabled. 	true

Step 8 Verify log transmission.

1. Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster*

- > **Services** > **HDFS**, click the HDFS WebUI link to go to the HDFS WebUI, and choose **Utilities** > **Browse the file system**.
 2. Check whether the data is generated in the **/flume/test** directory on the HDFS.
- End

7.12 Viewing Flume Client Monitoring Information

Scenario

The Flume client outside the FusionInsight cluster is a part of the end-to-end data collection. Both the Flume client outside the cluster and the Flume server in the cluster need to be monitored. Users can use FusionInsight Manager to monitor the Flume client and view the monitoring indicators of the Source, Sink, and Channel of the client as well as the client process status.

This section applies to MRS 3.x or later.

Procedure

- Step 1** Log in to FusionInsight Manager.
 - Step 2** Choose **Cluster** > *Name of the desired cluster* > **Services** > **Flume** > **Flume Management** to view the current Flume client list and process status.
 - Step 3** Click the **Instance ID**, and view client monitoring metrics in the **Current** area.
 - Step 4** Click **History**. The page for querying historical monitoring data is displayed. Select a time range and click **View** to view the monitoring data within the time range.
- End

7.13 Connecting Flume to Kafka in Security Mode

Scenario

This section describes how to connect to Kafka using the Flume client in security mode.

This section applies to MRS 3.x or later.

Procedure

- Step 1** Create a **jaas.conf** file and save it to **`\${Flume client installation directory}`/conf**. The content of the **jaas.conf** file is as follows:

```
KafkaClient {
  com.sun.security.auth.module.Krb5LoginModule required
  useKeyTab=true
  keyTab="/opt/test/conf/user.keytab"
  principal="flume_hdfs@<System domain name>"
  useTicketCache=false
  storeKey=true
  debug=true;
};
```

Set **keyTab** and **principal** based on site requirements. The configured **principal** must have certain kafka permissions.

Step 2 Configure services. Set the port number of **kafka.bootstrap.servers** to **21007**, and set **kafka.security.protocol** to **SASL_PLAINTEXT**.

Step 3 If the domain name of the cluster where Kafka is located is changed, change the value of **-Dkerberos.domain.name** in the **flume-env.sh** file in **`\${Flume client installation directory}`/conf/** based on the site requirements.

Step 4 Upload the configured **properties.properties** file to **`\${Flume client installation directory}`/conf**.

----End

7.14 Connecting Flume with Hive in Security Mode

Scenario

This section describes how to use Flume to connect to Hive (version 3.1.0) in the cluster.

This section applies to MRS 3.x or later.

Prerequisites

Flume and Hive have been correctly installed in the cluster. The services are running properly, and no alarm is reported.

Procedure

Step 1 Import the following JAR packages to the lib directory (client/server) of the Flume instance to be tested as user **omm**:

- antlr-2.7.7.jar
- antlr-runtime-3.4.jar
- calcite-core-1.16.0.jar
- hadoop-mapreduce-client-core-3.1.1.jar
- hive-beeline-3.1.0.jar
- hive-cli-3.1.0.jar
- hive-common-3.1.0.jar
- hive-exec-3.1.0.jar
- hive-hcatalog-core-3.1.0.jar
- hive-hcatalog-pig-adapter-3.1.0.jar
- hive-hcatalog-server-extensions-3.1.0.jar
- hive-hcatalog-streaming-3.1.0.jar
- hive-metastore-3.1.0.jar
- hive-service-3.1.0.jar
- libfb303-0.9.3.jar

- hadoop-plugins-1.0.jar

You can obtain the JAR package from the Hive installation directory and restart the Flume process to ensure that the JAR package is loaded to the running environment.

Step 2 Set Hive configuration items.

On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive** > **Configurations** > **All Configurations** > **HiveServer** > **Customization** > **hive.server.customized.configs**.

Example configurations:

Name	Value
hive.support.concurrency	true
hive.exec.dynamic.partition.mode	nonstrict
hive.txn.manager	org.apache.hadoop.hive.ql.lockmgr.DbTxnManager
hive.compactor.initiator.on	true
hive.compactor.worker.threads	1

Step 3 Prepare the system user **flume_hive** who has the supergroup and Hive permissions, install the client, and create the required Hive table.

Example:

1. The cluster client has been correctly installed. For example, the installation directory is **/opt/client**.
2. Run the following command to authenticate the user:

```
cd /opt/client
source bigdata_env
kinit flume_hive
```
3. Run the **beeline** command and run the following table creation statement:

```
create table flume_multi_type_part(id string, msg string)
partitioned by (country string, year_month string, day string)
clustered by (id) into 5 buckets
stored as orc TBLPROPERTIES('transactional'='true');
```
4. Run the **select * from Table name;** command to query data in the table.
 In this case, the number of data records in the table is **0**.

Step 4 Prepare related configuration files. Assume that the client installation package is stored in **/opt/FusionInsight_Cluster_1_Services_ClientConfig**.

1. Obtain the following files from the **\$Client decompression directory/Hive/config** directory:
 - hivemetastore-site.xml
 - hive-site.xml
2. Obtain the following files from the **\$Client decompression directory/HDFS/config** directory:

core-site.xml

3. Create a directory on the host where the Flume instance is started and save the prepared files to the created directory.

Example: **/opt/hivesink-conf/hive-site.xml**.

4. Copy all property configurations in the **hivemetastore-site.xml** file to the **hive-site.xml** file and ensure that the configurations are placed before the original configurations.

Data is loaded in sequence in Hive.

 **NOTE**

Ensure that the Flume running user **omm** has the read and write permissions on the directory where the configuration file is stored.

Step 5 Observe the result.

On the Hive client, run the **select * from *Table name*;** command. Check whether the corresponding data has been written to the Hive table.

----End

Examples

Flume configuration example (SpoolDir--Mem--Hive):

```
server.sources = spool_source
server.channels = mem_channel
server.sinks = Hive_Sink

#config the source
server.sources.spool_source.type = spooldir
server.sources.spool_source.spoolDir = /tmp/testflume
server.sources.spool_source.montime =
server.sources.spool_source.fileSuffix = .COMPLETED
server.sources.spool_source.deletePolicy = never
server.sources.spool_source.trackerDir = flumespool
server.sources.spool_source.ignorePattern = ^$
server.sources.spool_source.batchSize = 20
server.sources.spool_source.inputCharset = UTF-8
server.sources.spool_source.selector.type = replicating
server.sources.spool_source.fileHeader = false
server.sources.spool_source.fileHeaderKey = file
server.sources.spool_source.basenameHeaderKey = basename
server.sources.spool_source.deserializer = LINE
server.sources.spool_source.deserializer.maxBatchLine = 1
server.sources.spool_source.deserializer.maxLineLength = 2048
server.sources.spool_source.channels = mem_channel

#config the channel
server.channels.mem_channel.type = memory
server.channels.mem_channel.capacity = 10000
server.channels.mem_channel.transactionCapacity = 2000
server.channels.mem_channel.channelFullCount = 10
server.channels.mem_channel.keep-alive = 3
server.channels.mem_channel.byteCapacity =
server.channels.mem_channel.byteCapacityBufferPercentage = 20

#config the sink
server.sinks.Hive_Sink.type = hive
server.sinks.Hive_Sink.channel = mem_channel
server.sinks.Hive_Sink.hive.metastore = thrift://${any MetaStore service IP address}:21088
server.sinks.Hive_Sink.hive.hiveSite = /opt/hivesink-conf/hive-site.xml
server.sinks.Hive_Sink.hive.coreSite = /opt/hivesink-conf/core-site.xml
server.sinks.Hive_Sink.hive.metastoreSite = /opt/hivesink-conf/hivemeatastore-site.xml
```



```
server.sinks.Hive_Sink.hive.database = default
server.sinks.Hive_Sink.hive.table = flume_multi_type_part
server.sinks.Hive_Sink.hive.partition = Tag,%Y-%m,%d
server.sinks.Hive_Sink.hive.txnsPerBatchAsk= 100
server.sinks.Hive_Sink.hive.autoCreatePartitions= true
server.sinks.Hive_Sink.useLocalTimeStamp = true
server.sinks.Hive_Sink.batchSize = 1000
server.sinks.Hive_Sink.hive.kerberosPrincipal= super1
server.sinks.Hive_Sink.hive.kerberosKeytab= /opt/mykeytab/user.keytab
server.sinks.Hive_Sink.round = true
server.sinks.Hive_Sink.roundValue = 10
server.sinks.Hive_Sink.roundUnit = minute
server.sinks.Hive_Sink.serializer = DELIMITED
server.sinks.Hive_Sink.serializer.delimiter= ";"
server.sinks.Hive_Sink.serializer.serdeSeparator= ','
server.sinks.Hive_Sink.serializer.fieldnames= id,msg
```

7.15 Configuring the Flume Service Model

7.15.1 Overview

This section applies to MRS 3.x or later.

Guide a reasonable Flume service configuration by providing performance differences between Flume common modules, to avoid a nonstandard overall service performance caused when a frontend Source and a backend Sink do not match in performance.

Only single channels are compared for description.

7.15.2 Service Model Configuration Guide

This section applies to MRS 3.x or later.

During Flume service configuration and module selection, the ultimate throughput of a sink must be greater than the maximum throughput of a source. Otherwise, in extreme load scenarios, the write speed of the source to a channel is greater than the read speed of sink from channel. Therefore, the channel is fully occupied due to frequent usage, and the performance is affected.

Avro Source and Avro Sink are usually used in pairs to transfer data between multiple Flume Agents. Therefore, Avro Source and Avro Sink do not become a performance bottleneck in general scenarios.

Inter-Module Performance

Based on comparison between the limit performances of modules, Kafka Sink and HDFS Sink can meet the throughput requirements when the front-end is SpoolDir Source. However, HBase Sink could become performance bottlenecks due to the low write performances thereof. As a result, data is stacked in Channel. If you have to use HBase Sink or other sinks that are prone to become performance bottlenecks, you can use **Channel Selector** or **Sink Group** to meet performance requirements.

Channel Selector

A channel selector allows a source to connect to multiple channels. Data of the source can be distributed or copied by selecting different types of selectors.

Currently, a channel selector provided by Flume can be a replicating channel selector or a multiplexing channel selector.

Replicating: indicates that the data of the source is synchronized to all channels.

Multiplexing: indicates that based on the value of a specific field of the header of an event, a channel is selected to send the data. In this way, the data is distributed based on a service type.

- Replicating configuration example:

```
client.sources = kafkasource
client.channels = channel1 channel2
client.sources.kafkasource.type = org.apache.flume.source.kafka.KafkaSource
client.sources.kafkasource.kafka.topics = topic1,topic2
client.sources.kafkasource.kafka.consumer.group.id = flume
client.sources.kafkasource.kafka.bootstrap.servers = 10.69.112.108:21007
client.sources.kafkasource.kafka.security.protocol = SASL_PLAINTEXT
client.sources.kafkasource.batchDurationMillis = 1000
client.sources.kafkasource.batchSize = 800
client.sources.kafkasource.channels = channel1 c el2

client.sources.kafkasource.selector.type = replicating
client.sources.kafkasource.selector.optional = channel2
```

Table 7-49 Parameters in the Replicating configuration example

Parameter	Default Value	Description
Selector.type	replicating	Selector type. Set this parameter to replicating .
Selector.optional	-	Optional channel. Configure this parameter as a list.

- Multiplexing configuration example:

```
client.sources = kafkasource
client.channels = channel1 channel2
client.sources.kafkasource.type = org.apache.flume.source.kafka.KafkaSource
client.sources.kafkasource.kafka.topics = topic1,topic2
client.sources.kafkasource.kafka.consumer.group.id = flume
client.sources.kafkasource.kafka.bootstrap.servers = 10.69.112.108:21007
client.sources.kafkasource.kafka.security.protocol = SASL_PLAINTEXT
client.sources.kafkasource.batchDurationMillis = 1000
client.sources.kafkasource.batchSize = 800
client.sources.kafkasource.channels = channel1 channel2

client.sources.kafkasource.selector.type = multiplexing
client.sources.kafkasource.selector.header = myheader
client.sources.kafkasource.selector.mapping.topic1 = channel1
client.sources.kafkasource.selector.mapping.topic2 = channel2
client.sources.kafkasource.selector.default = channel1
```

Table 7-50 Parameters in the Multiplexing configuration example

Parameter	Default Value	Description
Selector.type	replicating	Selector type. Set this parameter to multiplexing .

Parameter	Default Value	Description
Selector.header	Flume.selector.header	-
Selector.default	-	-
Selector.mapping.*	-	-

In a multiplexing selector example, select a field whose name is topic from the header of the event. When the value of the topic field in the header is topic1, send the event to a channel 1; or when the value of the topic field in the header is topic2, send the event to a channel 2.

Selectors need to use a specific header of an event in a source to select a channel, and need to select a proper header based on a service scenario to distribute data.

SinkGroup

When the performance of a backend single sink is insufficient, and high reliability or heterogeneous output is required, you can use a sink group to connect a specified channel to multiple sinks, thereby meeting use requirements. Currently, Flume provides two types of sink processors to manage sinks in a sink group. The types are load balancing and failover.

Failover: Indicates that there is only one active sink in the sink group each time, and the other sinks are on standby and inactive. When the active sink becomes faulty, one of the inactive sinks is selected based on priorities to take over services, so as to ensure that data is not lost. This is used in high-reliability scenarios.

Load balancing: Indicates that all sinks in the sink group are active. Each sink obtains data from the channel and processes the data. In addition, during running, loads of all sinks in the sink group are balanced. This is used in performance improvement scenarios.

- Load balancing configuration examples:

```
client.sources = source1
client.sinks = sink1 sink2
client.channels = channel1

client.sinkgroups = g1
client.sinkgroups.g1.sinks = sink1 sink2
client.sinkgroups.g1.processor.type = load_balance
client.sinkgroups.g1.processor.backoff = true
client.sinkgroups.g1.processor.selector = random

client.sinks.sink1.type = logger
client.sinks.sink1.channel = channel1

client.sinks.sink2.type = logger
client.sinks.sink2.channel = channel1
```

Table 7-51 Parameters of Load Balancing configuration examples

Parameter	Default Value	Description
sinks	-	Specifies the sink list of the sink group. Multiple sinks are separated by spaces.
processor.type	default	Specifies the type of a processor. Set this parameter to load_balance .
processor.backoff	false	Indicates whether to back off failed sinks exponentially.
processor.selector	round_robin	Specifies the selection mechanism. It must be round_robin, random, or a customized class that inherits AbstractSinkSelector.
processor.selector.maxTimeOut	30000	Specifies the time for masking a faulty sink. The default value is 30,000 ms.

- Failover configuration examples:

```

client.sources = source1
client.sinks = sink1 sink2
client.channels = channel1

client.sinkgroups = g1
client.sinkgroups.g1.sinks = sink1 sink2
client.sinkgroups.g1.processor.type = failover
client.sinkgroups.g1.processor.priority.sink1 = 10
client.sinkgroups.g1.processor.priority.sink2 = 5
client.sinkgroups.g1.processor.maxpenalty = 10000

client.sinks.sink1.type = logger
client.sinks.sink1.channel = channel1

client.sinks.sink2.type = logger
client.sinks.sink2.channel = channel1
    
```

Table 7-52 Parameters in the **failover** configuration example

Parameter	Default Value	Description
sinks	-	Specifies the sink list of the sink group. Multiple sinks are separated by spaces.

Parameter	Default Value	Description
processor.type	default	Specifies the type of a processor. Set this parameter to failover .
processor.priority.<sink Name>	-	Priority. <sinkName> must be defined in description of sinks. A sink having a higher priority is activated earlier. A larger value indicates a higher priority. Note: If there are multiple sinks, their priorities must be different. Otherwise, only one of them takes effect.
processor.maxpenalty	30000	Specifies the maximum backoff time of failed sinks (unit: ms).

Interceptors

The Flume interceptor supports modification or discarding of basic unit events during data transmission. You can specify the class name list of built-in interceptors in Flume or develop customized interceptors to modify or discard events. The following table lists the built-in interceptors in Flume. A complex example is used in this section. Other users can configure and use interceptions as required.

NOTE

1. The interceptor is used between the sources and channels of Flume. Most sources provide parameters for configuring interceptors. You can set the parameters as required.
2. Flume allows multiple interceptors to be configured for a source. The interceptor names are separated by spaces.
3. The specified interceptor sequence is the order in which they are called.
4. The contents inserted by the interceptor in the header can be read and used in sink.

Table 7-53 Types of built-in interceptors in Flume

Interceptor Type	Description
Timestamp Interceptor	The interceptor inserts a timestamp into the header of an event.
Host Interceptor	The interceptor inserts the IP address or host name of the node where the agent is located into the Header of an event.

Interceptor Type	Description
Remove Header Interceptor	The interceptor discards the corresponding event based on the strings that matches the regular expression contained in the event header.
UUID Interceptor	The interceptor generates a UUID string for the header of each event.
Search and Replace Interceptor	The interceptor provides a simple string-based search and replacement function based on Java regular expressions. The rule is the same as that of Java <code>Matcher.replaceAll()</code> .
Regex Filtering Interceptor	The interceptor uses the body of an event as a text file and matches the configured regular expression to filter events. The provided regular expression can be used to exclude or include events.
Regex Extractor Interceptor	The interceptor extracts content from the original events using a regular expression and adds the content to the header of events.

Regex Filtering Interceptor is used as an example to describe how to use the interceptor. (For other types of interceptions, see the configuration provided on the official website.)

Table 7-54 Parameter configuration for **Regex Filtering Interceptor**

Parameter	Default Value	Description
type	-	Specifies the component type name. The value must be regex_filter .
regex	-	Specifies the regular expression used to match events.
excludeEvents	false	By default, the matched events are collected. If this parameter is set to true , the matched events are deleted and the unmatched events are retained.

Configuration example (netcat tcp is used as the source, and logger is used as the sink). After configuring the preceding parameters, run the **telnet Host name or IP address 4444** command on the host where the Linux operating system is run, and enter a string that complies with the regular expression and another does not

comply with the regular expression. The log shows that only the matched string is transmitted.

```
#define the source, channel, sink
server.sources = r1

server.channels = c1
server.sinks = k1

#config the source
server.sources.r1.type = netcat
server.sources.r1.bind = ${Host IP address}
server.sources.r1.port = 44444
server.sources.r1.interceptors= i1
server.sources.r1.interceptors.i1.type= regex_filter
server.sources.r1.interceptors.i1.regex= (flume)|(myflume)
server.sources.r1.interceptors.i1.excludeEvents= false
server.sources.r1.channels = c1

#config the channel
server.channels.c1.type = memory
server.channels.c1.capacity = 1000
server.channels.c1.transactionCapacity = 100
#config the sink
server.sinks.k1.type = logger
server.sinks.k1.channel = c1
```

7.16 Introduction to Flume Logs

Log Description

Log path: The default path of Flume log files is `/var/log/Bigdata/Role name`.

- FlumeServer: `/var/log/Bigdata/flume/flume`
- FlumeClient: `/var/log/Bigdata/flume-client-n/flume`
- MonitorServer: `/var/log/Bigdata/flume/monitor`

Log archive rule: The automatic Flume log compression function is enabled. By default, when the size of logs exceeds 50 MB , logs are automatically compressed into a log file named in the following format: `<Original log file name>-<yyyy-mm-dd_hh-mm-ss>.[ID].log.zip`. A maximum of 20 latest compressed files are reserved. The number of compressed files can be configured on the Manager portal.

Table 7-55 Flume log list

Type	Name	Description
Run logs	<code>/flume/flumeServer.log</code>	Log file that records FlumeServer running environment information.
	<code>/flume/install.log</code>	FlumeServer installation log file
	<code>/flume/flumeServer-gc.log.<No.></code>	GC log file of the FlumeServer process
	<code>/flume/prestartDvietail.log</code>	Work log file before the FlumeServer startup

Type	Name	Description
	/flume/startDetail.log	Startup log file of the Flume process
	/flume/stopDetail.log	Shutdown log file of the Flume process
	/monitor/monitorServer.log	Log file that records MonitorServer running environment information
	/monitor/startDetail.log	Startup log file of the MonitorServer process
	/monitor/stopDetail.log	Shutdown log file of the MonitorServer process
	function.log	External function invoking log file
	threadDump-<DATE>.log	The jstack log file to be printed when the NodeAgent delivers a service stop command

Log Level

Table 7-56 describes the log levels supported by Flume.

Levels of run logs are FATAL, ERROR, WARN, INFO, and DEBUG from the highest to the lowest priority. Run logs of equal or higher levels are recorded. The higher the specified log level, the fewer the logs recorded.

Table 7-56 Log level

Type	Level	Description
Run log	FATAL	Logs of this level record critical error information about system running.
	ERROR	Logs of this level record error information about system running.
	WARN	Logs of this level record exception information about the current event processing.

Type	Level	Description
	INFO	Logs of this level record normal running status information about the system and events.
	DEBUG	Logs of this level record the system information and system debugging information.

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page of Flume by referring to [Modifying Cluster Service Configuration Parameters](#).
- Step 2** On the menu bar on the left, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.

----End

 **NOTE**

The configurations take effect immediately without the need to restart the service.

Log Format

The following table lists the Flume log formats.

Table 7-57 Log format

Type	Format	Example
Run logs	<i><yyyy-MM-dd HH:mm:ss,SSS> <Log level> <Name of the thread that generates the log> <Message in the log> <Location where the log event occurs></i>	2014-12-12 11:54:57,316 INFO [main] log4j dynamic load is start. org.apache.flume.tools.LogDynamicLoad.start(LogDynamicLoad.java:59)
	<i><yyyy-MM-dd HH:mm:ss,SSS><Username><User IP><Time><Operation><Resource><Result><Detail></i>	2014-12-12 23:04:16,572 INFO [SinkRunner-PollingRunner-DefaultSinkProcessor] SRCIP=null OPERATION=close

7.17 Flume Client Cgroup Usage Guide

Scenario

This section describes how to join and log out of a cgroup, query the cgroup status, and change the cgroup CPU threshold.

This section applies to MRS 3.x or later.

Procedure

- **Join Cgroup**

Assume that the Flume client installation path is `/opt/FlumeClient`, and the cgroup CPU threshold is 50%. Run the following command to join a cgroup:

```
cd /opt/FlumeClient/fusioninsight-flume-1.9.0/bin
./flume-manage.sh cgroup join 50
```

 NOTE

- This command can be used to join a cgroup and change the cgroup CPU threshold.
 - The value of the CPU threshold of a cgroup ranges from 1 to 100 x *N*. *N* indicates the number of CPU cores.
- **Check Cgroup status**

Assume that the Flume client installation path is `/opt/FlumeClient`. Run the following commands to query the cgroup status:

```
cd /opt/FlumeClient/fusioninsight-flume-1.9.0/bin
./flume-manage.sh cgroup status
```

- **Exit Cgroup**

Assume that the Flume client installation path is `/opt/FlumeClient`. Run the following commands to exit cgroup:

```
cd /opt/FlumeClient/fusioninsight-flume-1.9.0/bin
./flume-manage.sh cgroup exit
```

 NOTE

- After the client is installed, the default cgroup is automatically created. If the `-s` parameter is not configured during client installation, the default value `-1` is used. The default value indicates that the agent process is not restricted by the CPU usage.
- Joining or exiting a cgroup does not affect the agent process. Even if the agent process is not started, the joining or exiting operation can be performed successfully, and the operation will take effect after the next startup of the agent process.
- After the client is uninstalled, the cgroups created during the client installation are automatically deleted.

7.18 Secondary Development Guide for Flume Third-Party Plug-ins

Scenario

This section describes how to perform secondary development for third-party plug-ins.

This section applies to MRS 3.x or later.

Prerequisites

- You have obtained the third-party JAR package.
- You have installed Flume server or client.

Procedure

Step 1 Compress the self-developed code into a JAR package.

Step 2 Create a directory for the plug-in.

1. Access the `$FLUME_HOME/plugins.d` path and run the following command to create a directory:

```
mkdir thirdPlugin
cd thirdPlugin
mkdir lib libext native
```

The command output is displayed as follows:

```
[root@██████████ plugins.d]#mkdir thirdPlugin
[root@██████████ plugins.d]#ll
total 8
drwxr-x--- 3 root root 4096 ██████████ native
drwxr-xr-x 2 root root 4096 ██████████ thirdPlugin
[root@██████████ plugins.d]#cd thirdPlugin/
[root@██████████ thirdPlugin]#mkdir lib libext native
[root@██████████ thirdPlugin]#ll
total 12
drwxr-xr-x 2 root root 4096 ██████████ lib
drwxr-xr-x 2 root root 4096 ██████████ libext
drwxr-xr-x 2 root root 4096 ██████████ native
[root@██████████ thirdPlugin]#
```

2. Place the third-party JAR package in the `$FLUME_HOME/plugins.d/thirdPlugin/lib` directory. If the JAR package depends on other JAR packages, place the depended JAR packages to the `$FLUME_HOME/plugins.d/thirdPlugin/libext` directory, and place the local library files in `$FLUME_HOME/plugins.d/thirdPlugin/native`.

Step 3 Configure the `properties.properties` file in `$FLUME_HOME/conf/`.

For details about how to set parameters in the `properties.properties` file, see the parameter list in the `properties.properties` file in the corresponding typical scenario [Non-Encrypted Transmission](#) and [Encrypted Transmission](#).

 NOTE

- **\$FLUME_HOME** indicates the Flume installation path. Set this parameter based on the site requirements (server or client) when configuring third-party plug-ins.
- **thirdPlugin** is the name of the third-party plugin.

----End

7.19 Common Issues About Flume

Flume logs are stored in `/var/log/Bigdata/flume/flume/flumeServer.log`. Most data transmission exceptions and data transmission failures are recorded in logs. You can run the following command:

tailf /var/log/Bigdata/flume/flume/flumeServer.log

- Problem: After the configuration file is uploaded, an exception occurs. After the configuration file is uploaded again, the scenario requirements are still not met, but no exception is recorded in the log.

Solution: Restart the Flume process, run the **kill -9 *Process code*** to kill the process code, and view the logs.

- Issue: "**java.lang.IllegalArgumentException: Keytab is not a readable file: /opt/test/conf/user.keytab**" is displayed when HDFS is connected.

Solution: Grant the read and write permissions to the Flume running user.

- Problem: The following error is reported when the Flume client is connected to Kafka:

Caused by: java.io.IOException: /opt/FlumeClient/fusioninsight-flume-1.9.0/cof//jaas.conf (No such file or directory)

Solution: Add the **jaas.conf** configuration file and save it to the **conf** directory of the Flume client.

vi jaas.conf

```
KafkaClient {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/opt/test/conf/user.keytab"
principal="flume_hdfs@<System domain name>"
useTicketCache=false
storeKey=true
debug=true;
};
```

Values of **keyTab** and **principal** vary depending on the actual situation.

- Problem: The following error is reported when the Flume client is connected to HBase:

Caused by: java.io.IOException: /opt/FlumeClient/fusioninsight-flume-1.9.0/cof//jaas.conf (No such file or directory)

Solution: Add the **jaas.conf** configuration file and save it to the **conf** directory of the Flume client.

vi jaas.conf

```
Client {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
keyTab="/opt/test/conf/user.keytab"
principal="flume_hbase@<System domain name>"
useTicketCache=false
```

```
storeKey=true  
debug=true;  
};
```

Values of **keyTab** and **principal** vary depending on the actual situation.

- Question: After the configuration file is submitted, the Flume Agent occupies resources. How do I restore the Flume Agent to the state when the configuration file is not uploaded?

Solution: Submit an empty **properties.properties** file.

8 Using HBase

8.1 Using HBase from Scratch

HBase is a column-based distributed storage system that features high reliability, performance, and scalability. This section describes how to use HBase from scratch, including how to update the client on the Master node in the cluster, create a table using the client, insert data in the table, modify the table, read data from the table, delete table data, and delete the table.

Background

Suppose a user develops an application to manage users who use service A in an enterprise. The procedure of operating service A on the HBase client is as follows:

- Create the **user_info** table.
- Add users' educational backgrounds and titles to the table.
- Query user names and addresses by user ID.
- Query information by user name.
- Deregister users and delete user data from the user information table.
- Delete the user information table after service A ends.

Table 8-1 User information

ID	Name	Gender	Age	Address
12005000201	A	Male	19	City A
12005000202	B	Female	23	City B
12005000203	C	Male	26	City C
12005000204	D	Male	18	City D
12005000205	E	Female	21	City E
12005000206	F	Male	32	City F

ID	Name	Gender	Age	Address
12005000207	G	Female	29	City G
12005000208	H	Female	30	City H
12005000209	I	Male	26	City I
12005000210	J	Male	25	City J

Prerequisites

The client has been installed. For example, the client is installed in the `/opt/client` directory. The client directory in the following operations is only an example. Change it to the actual installation directory. Before using the client, download and update the client configuration file, and ensure that the active management node of Manager is available.

Procedure

For versions earlier than MRS 3.x, perform the following operations:

Step 1 Download the client configuration file.

1. Log in to MRS Manager. For details, see [Accessing Manager](#). Then, choose **Services**.
2. Click **Download Client**.
Set **Client Type** to **Only configuration files**, **Download To** to **Server**, and click **OK** to generate the client configuration file. The generated file is saved in the `/tmp/MRS-client` directory on the active management node by default. You can customize the file path.

Step 2 Log in to the active management node of MRS Manager.

1. On the **Node** tab page, view the **Name** parameter. The node that contains **master1** in its name is the Master1 node. The node that contains **master2** in its name is the Master2 node.
The active and standby management nodes of MRS Manager are installed on Master nodes by default. Because Master1 and Master2 are switched over in active and standby mode, Master1 is not always the active management node of MRS Manager. Run a command in Master1 to check whether Master1 is active management node of MRS Manager. For details about the command, see [Step 2.4](#).
2. Log in to the Master1 node using the password as user **root**. For details, see [Logging In to an ECS](#).
3. Run the following commands to switch to user **omm**:
sudo su - root
su - omm
4. Run the following command to check the active management node of MRS Manager:
sh \${BIGDATA_HOME}/om-0.0.1/sbin/status-oms.sh

In the command output, the node whose **HAActive** is **active** is the active management node, and the node whose **HAActive** is **standby** is the standby management node. In the following example, **mgtomsdat-sh-3-01-1** is the active management node, and **mgtomsdat-sh-3-01-2** is the standby management node.

```
Ha mode
double
NodeName      HostName      HAVersion      StartTime      HAActive
HAAllResOK    HARunPhase
192-168-0-30  mgtomsdat-sh-3-01-1  V100R001C01    2021-11-18 23:43:02
active       normal        Activated
192-168-0-24  mgtomsdat-sh-3-01-2  V100R001C01    2021-11-21 07:14:02
standby     normal        Deactivated
```

5. Log in to the active management node, for example, **192-168-0-30** of MRS Manager as user **root**, and run the following command to switch to user **omm**:

```
sudo su - omm
```

- Step 3** Run the following command to switch to the client installation directory, for example, **/opt/client**:

```
cd /opt/client
```

- Step 4** Run the following command to update the client configuration for the active management node.

```
sh refreshConfig.sh /opt/client Full path of the client configuration file package
```

For example, run the following command:

```
sh refreshConfig.sh /opt/client /tmp/MRS-client/MRS_Services_Client.tar
```

If the following information is displayed, the configurations have been updated successfully.

```
ReFresh components client config is complete.
Succeed to refresh components client config.
```

 **NOTE**

You can refer to steps [Step 1](#) to [Step 4](#) or Method 2 in [Updating a Client](#).

- Step 5** Use the client on a Master node.

1. On the active management node where the client is updated, for example, node **192-168-0-30**, run the following command to go to the client directory:

```
cd /opt/client
```

2. Run the following command to configure environment variables:

```
source bigdata_env
```

3. If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the current user. The current user must have the permission to create HBase tables. For details about how to configure a role with the corresponding permissions, see [Creating a Role](#). To bind a role to a user, see [Creating a User](#). If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit MRS cluster user
```

For example, **kinit hbaseuser**.

4. Run the following HBase client command:

```
hbase shell
```

Step 6 Run the following commands on the HBase client to implement service A.

1. Create the **user_info** user information table according to [Table 8-1](#) and add data to it.

```
create 'user_info',{NAME => 'i'}
```

For example, to add information about the user whose ID is 12005000201, run the following commands:

```
put 'user_info','12005000201','i:name','A'
```

```
put 'user_info','12005000201','i:gender','Male'
```

```
put 'user_info','12005000201','i:age','19'
```

```
put 'user_info','12005000201','i:address','City A'
```

2. Add users' educational backgrounds and titles to the **user_info** table.

For example, to add educational background and title information about user 12005000201, run the following commands:

```
put 'user_info','12005000201','i:degree','master'
```

```
put 'user_info','12005000201','i:pose','manager'
```

3. Query user names and addresses by user ID.

For example, to query the name and address of user 12005000201, run the following command:

```
scan 'user_info',  
{STARTROW=>'12005000201',STOPROW=>'12005000201',COLUMNS=>['i:name','i:address']}
```

4. Query information by user name.

For example, to query information about user A, run the following command:

```
scan 'user_info',{FILTER=>"SingleColumnValueFilter('i','name',=,'binary:A')"
```

5. Delete user data from the user information table.

All user data needs to be deleted. For example, to delete data of user 12005000201, run the following command:

```
delete 'user_info','12005000201','i'
```

6. Delete the user information table.

```
disable 'user_info'
```

```
drop 'user_info'
```

----End

For MRS 3.x or later, perform the following operations:

Step 1 Use the client on the active management node.

1. Log in to the node where the client is installed as the client installation user and run the following command to switch to the client directory:

```
cd /opt/client
```

2. Run the following command to configure environment variables:

```
source bigdata_env
```

3. If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the current user. The current user must have the permission to create HBase tables. For details about how to configure a role with corresponding permissions, see [Creating a Role](#). To bind a role to a user, see [Creating a User](#). If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit MRS cluster user
```

For example, **kinit hbaseuser**.

4. Run the following HBase client command:

```
hbase shell
```

Step 2 Run the following commands on the HBase client to implement service A.

1. Create the **user_info** user information table according to [Table 8-1](#) and add data to it.

```
create 'user_info',{NAME => 'i'}
```

For example, to add information about the user whose ID is **12005000201**, run the following commands:

```
put 'user_info','12005000201','i:name','A'
```

```
put 'user_info','12005000201','i:gender','Male'
```

```
put 'user_info','12005000201','i:age','19'
```

```
put 'user_info','12005000201','i:address','City A'
```

2. Add users' educational backgrounds and titles to the **user_info** table.

For example, to add educational background and title information about user 12005000201, run the following commands:

```
put 'user_info','12005000201','i:degree','master'
```

```
put 'user_info','12005000201','i:pose','manager'
```

3. Query user names and addresses by user ID.

For example, to query the name and address of user 12005000201, run the following command:

```
scan'user_info',  
{STARTROW=>'12005000201',STOPROW=>'12005000201',COLUMNS=>['i:na  
me','i:address']}
```

4. Query information by user name.

For example, to query information about user A, run the following command:

```
scan'user_info',{FILTER=>"SingleColumnValueFilter('i','name',=,'binary:A')"
```

5. Delete user data from the user information table.

All user data needs to be deleted. For example, to delete data of user 12005000201, run the following command:

```
delete'user_info','12005000201','i'
```

6. Delete the user information table.

```
disable'user_info'
```

```
drop 'user_info'
```

----End

8.2 Using an HBase Client

Scenario

This section describes how to use the HBase client in an O&M scenario or a service scenario.

Prerequisites

- The client has been installed. For example, the installation directory is **/opt/hadoopclient**. The client directory in the following operations is only an example. Change it to the actual installation directory.
- Service component users are created by the MRS cluster administrator as required.
A machine-machine user needs to download the **keytab** file and a human-machine user needs to change the password upon the first login.
- If a non-**root** user uses the HBase client, ensure that the owner of the HBase client directory is this user. Otherwise, run the following command to change the owner.

```
chown user:group -R Client installation directory/HBase
```

Using the HBase Client (Versions Earlier Than MRS 3.x)

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client directory:

```
cd /opt/hadoopclient
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the current user. The current user must have the permission to create HBase tables. For details about how to configure a role with corresponding permissions, see [Creating a Role](#). To bind a role to a user, see [Creating a User](#). If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit Component service user
```

For example, **kinit hbaseuser**.

Step 5 Run the following HBase client command:

```
hbase shell
```

```
----End
```

Using the HBase Client (MRS 3.x or Later)

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client directory:

```
cd /opt/hadoopclient
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 If you use the client to connect to a specific HBase instance in a scenario where multiple HBase instances are installed, run the following command to load the environment variables of the instance. Otherwise, skip this step. For example, to load the environment variables of the HBase2 instance, run the following command:

```
source HBase2/component_env
```

Step 5 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the current user. The current user must have the permission to create HBase tables. For details about how to configure a role with corresponding permissions, see [Creating a Role](#). To bind a role to a user, see [Creating a User](#). If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit Component service user
```

For example, **kinit hbaseuser**.

Step 6 Run the following HBase client command:

```
hbase shell
```

```
----End
```

Common HBase client commands

The following table lists common HBase client commands. For more commands, see <http://hbase.apache.org/2.2/book.html>.

Table 8-2 HBase client commands

Command	Description
create	Used to create a table, for example, create 'test', 'f1', 'f2', 'f3' .
disable	Used to disable a specified table, for example, disable 'test' .
enable	Used to enable a specified table, for example, enable 'test' .
alter	Used to alter the table structure. You can run the alter command to add, modify, or delete column family information and table-related parameter values, for example, alter 'test', {NAME => 'f3', METHOD => 'delete'} .
describe	Used to obtain the table description, for example, describe 'test' .
drop	Used to delete a specified table, for example, drop 'test' . Before deleting a table, you must stop it.

Command	Description
put	Used to write the value of a specified cell, for example, put 'test','r1','f1:c1','myvalue1' . The cell location is unique and determined by the table, row, and column.
get	Used to get the value of a row or the value of a specified cell in a row, for example, get 'test','r1' .
scan	Used to query table data, for example, scan 'test' . The table name and scanner must be specified in the command.

8.3 Creating HBase Roles

Scenario

This section guides the you to create and configure an HBase role on Manager. The HBase role can set HBase administrator permissions and read (R), write (W), create (C), execute (X), or manage (A) permissions for HBase tables and column families.

Users can create a table, query/delete/insert/update data, and authorize others to access HBase tables after they set the corresponding permissions for the specified databases or tables on HDFS.

NOTE

- This section applies to MRS 3.x or later.
- HBase roles can be created in security mode, but cannot be created in normal mode.
- If the current component uses Ranger for permission control, you need to configure related policies based on Ranger for permission management. For details, see [Adding a Ranger Access Permission Policy for HBase](#).

Prerequisites

- You have understood the service requirements.
- You have logged in to Manager.

Procedure

Step 1 On Manager, choose **System > Permission > Role**.

Step 2 On the displayed page, click **Create Role** and enter a **Role Name** and **Description**.

Step 3 Set **Permission**. For details, see [Table 8-3](#).

HBase permissions:

- HBase Scope: Authorizes HBase tables. The minimum permission is read (R) and write (W) for columns.
- HBase administrator permission: HBase administrator permissions.

 **NOTE**

Users have the read (R), write (W), create (C), execute (X), and administrate (A) permissions for the tables created by themselves.

Table 8-3 Setting a role

Task	Role Authorization
Setting the HBase administrator permission	In Configure Resource Permission , choose <i>Name of the desired cluster</i> > HBase and select HBase Administrator Permission .
Setting the permission for users to create tables	<ol style="list-style-type: none"> 1. In Configure Resource Permission, choose <i>Name of the desired cluster</i> > HBase > HBase Scope. 2. Click global. 3. In the Permission column of the specified namespace, select Create and Execute. For example, select Create and Execute for the default namespace default.
Setting the permission for users to write data to tables	<ol style="list-style-type: none"> 1. In Configure Resource Permission, choose <i>Name of the desired cluster</i> > HBase > HBase Scope > global. 2. In the Permission column of the specified namespace, select Write. For example, select Write for the default namespace default. By default, HBase sub-objects inherit the permission from the parent object.
Setting the permission for users to read data from tables	<ol style="list-style-type: none"> 1. In Configure Resource Permission, choose <i>Name of the desired cluster</i> > HBase > HBase Scope > global. 2. In the Permission column of the specified namespace, select Read. For example, select Read for the default namespace default. By default, HBase sub-objects inherit the permission from the parent object.
Setting the permission for users to manage namespaces or tables	<ol style="list-style-type: none"> 1. In Configure Resource Permission, choose <i>Name of the desired cluster</i> > HBase > HBase Scope > global. 2. In the Permission column of the specified namespace, select Manage. For example, select Manage for the default namespace default.

Task	Role Authorization
<p>Setting the permission for reading data from or writing data to columns</p>	<ol style="list-style-type: none"> 1. In Configure Resource Permission, select <i>Name of the desired cluster</i> > HBase > HBase Scope > global and click the specified namespace to display the tables in the namespace. 2. Click a table. 3. Click a column family. 4. Confirm whether you want to create a role? <ul style="list-style-type: none"> - If yes, enter the column name in the Resource Name text box. Use commas (,) to separate multiple columns. Select Read or Write. If there are no columns with the same name in the HBase table, a newly created column with the same name as the existing column has the same permission as the existing one. The column permission is set successfully. - If no, modify the column permission of the existing HBase role. The columns for which the permission has been separately set are displayed in the table. Go to Step 3.5. 5. To add column permissions for a role, enter the column name in the Resource Name text box and set the column permissions. To modify column permissions for a role, enter the column name in the Resource Name text box and set the column permissions. Alternatively, you can directly modify the column permissions in the table. If the column permissions are modified in the table and column permissions with the same name are added, the settings cannot be saved. You are advised to modify the column permission of a role directly in the table. The search function is supported.

Step 4 Click **OK**, and return to the **Role** page.

----End

8.4 Configuring HBase Replication

Scenario

As a key feature to ensure high availability of the HBase cluster system, HBase cluster replication provides HBase with remote data replication in real time. It provides basic O&M tools, including tools for maintaining and re-establishing active/standby relationships, verifying data, and querying data synchronization progress. To achieve real-time data replication, you can replicate data from the HBase cluster to another one.

Prerequisites

- The active and standby clusters have been successfully installed and started (the cluster status is **Running** on the **Active Clusters** page), and you have the administrator rights of the clusters.
- The network between the active and standby clusters is normal and ports can be used properly.
- Cross-cluster mutual trust has been configured. For details, see [Configuring Cross-Cluster Mutual Trust Relationships](#).
- If historical data exists in the active cluster and needs to be synchronized to the standby cluster, cross-cluster replication must be configured for the active and standby clusters. For details, see [Enabling Cross-Cluster Copy](#).
- Time is consistent between the active and standby clusters and the Network Time Protocol (NTP) service on the active and standby clusters uses the same time source.
- Mapping relationships between the names of all hosts in the active and standby clusters and service IP addresses have been configured in the `/etc/hosts` file by appending `192.***.***.*** host1` to the `hosts` file.
- The network bandwidth between the active and standby clusters is determined based on service volume, which cannot be less than the possible maximum service volume.

Constraints

- Despite that HBase cluster replication provides the real-time data replication function, the data synchronization progress is determined by several factors, such as the service loads in the active cluster and the health status of processes in the standby cluster. In normal cases, the standby cluster should not take over services. In extreme cases, system maintenance personnel and other decision makers determine whether the standby cluster takes over services according to the current data synchronization indicators.
- Currently, the replication function supports only one active cluster and one standby cluster in HBase.
- Typically, do not perform operations on data synchronization tables in the standby cluster, such as modifying table properties or deleting tables. If any misoperation on the standby cluster occurs, data synchronization between the active and standby clusters will fail and data of the corresponding table in the standby cluster will be lost.
- If the replication function of HBase tables in the active cluster is enabled for data synchronization, after modifying the structure of a table in the active cluster, you need to manually modify the structure of the corresponding table in the standby cluster to ensure table structure consistency.

Procedure

Enable the replication function for the active cluster to synchronize data written by Put.

- Step 1** Log in to the MRS console, click a cluster name and choose **Components**.
- Step 2** Go to the **All Configurations** page of the HBase service. For details, see [Modifying Cluster Service Configuration Parameters](#).

 NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

Step 3 (Optional) Set configuration items listed in [Table 8-4](#). You can set the parameters based on the description or use the default values.

Table 8-4 Optional configuration items

Navigation Path	Parameter	Default Value	Description
HMaster > Performance	hbase.master.logcleaner.ttl	600000	Time to live (TTL) of HLog files. If the value is set to 604800000 (unit: millisecond), the retention period of HLog is 7 days.
	hbase.master.cleaner.interval	60000	Interval for the HMaster to delete historical HLog files. The HLog that exceeds the configured period will be automatically deleted. You are advised to set it to the maximum value to save more HLogs.
RegionServer > Replication	replication.source.size.capacity	16777216	Maximum size of edits, in bytes. If the edit size exceeds the value, HLog edits will be sent to the standby cluster.
	replication.source.nb.capacity	25000	Maximum number of edits, which is another condition for triggering HLog edits to be sent to the standby cluster. After data in the active cluster is synchronized to the standby cluster, the active cluster reads and sends data in HLog according to this parameter value. This parameter is used together with replication.source.size.capacity .
	replication.source.maxretriesmultiplier	10	Maximum number of retries when an exception occurs during replication.
	replication.source.sleepforretries	1000	Retry interval (unit: ms)

Navigation Path	Parameter	Default Value	Description
	hbase.regionserver.replication.handler.count	6	Number of replication RPC server instances on RegionServer

Enable the replication function for the active cluster to synchronize data written by bulkload.

Step 4 Determine whether to enable bulkload replication.

 **NOTE**

If bulkload import is used and data needs to be synchronized, you need to enable Bulkload replication.

If yes, go to [Step 5](#).

If no, go to [Step 9](#).

Step 5 Go to the **All Configurations** page of the HBase service parameters by referring to [Modifying Cluster Service Configuration Parameters](#).

Step 6 On the HBase configuration interface of the active and standby clusters, search for **hbase.replication.cluster.id** and modify it. It specifies the HBase ID of the active and standby clusters. For example, the HBase ID of the active cluster is set to **replication1** and the HBase ID of the standby cluster is set to **replication2** for connecting the active cluster to the standby cluster. To save data overhead, the parameter value length is not recommended to exceed 30.

Step 7 On the HBase configuration interface of the standby cluster, search for **hbase.replication.conf.dir** and modify it. It specifies the HBase configurations of the active cluster client used by the standby cluster and is used for data replication when the bulkload data replication function is enabled. The parameter value is a path name, for example, **/home**.

 **NOTE**

- In versions earlier than MRS 3.x, you do not need to set this parameter. Skip [Step 7](#).
- When bulkload replication is enabled, you need to manually place the HBase client configuration files (**core-site.xml**, **hdfs-site.xml**, and **hbase-site.xml**) in the active cluster on all RegionServer nodes in the standby cluster. The actual path for placing the configuration file is **`\${hbase.replication.conf.dir}/\${hbase.replication.cluster.id}**. For example, if **hbase.replication.conf.dir** of the standby cluster is set to **/home** and **hbase.replication.cluster.id** of the active cluster is set to **replication1**, the actual path for placing the configuration files in the standby cluster is **/home/replication1**. You also need to change the corresponding directory and file permissions by running the **chown -R omm:wheel /home/replication1** command.
- You can obtain the client configuration files from the client in the active cluster, for example, the **/opt/client/HBase/hbase/conf** path. For details about how to update the configuration file, see [Updating a Client](#).

Step 8 On the HBase configuration page of the active cluster, search for and change the value of **hbase.replication.bulkload.enabled** to **true** to enable bulkload replication.

Restarting the HBase service and install the client

Step 9 Save the configurations and restart HBase.

Step 10 In the active and standby clusters. For details about how to update the client configuration file, see [Updating a Client](#).

Synchronize table data of the active cluster. (Skip this step if the active cluster has no data.)

Step 11 Access the HBase shell of the active cluster as user **hbase**.

1. On the active management node where the client has been updated, run the following command to go to the client directory:

```
cd /opt/client
```

2. Run the following command to configure environment variables:

```
source bigdata_env
```

3. If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the current user. If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit hbase
```

 **NOTE**

The system prompts you to enter the password after you run **kinit hbase**. The default password of user **hbase** is **Hbase@123**.

4. Run the following HBase client command:

```
hbase shell
```

Step 12 Check whether historical data exists in the standby cluster. If historical data exists and data in the active and standby clusters must be consistent, delete data from the standby cluster first.

1. On the HBase shell of the standby cluster, run the **list** command to view the existing tables in the standby cluster.
2. Delete data tables from the standby cluster based on the output list.

```
disable 'tableName'
```

```
drop 'tableName'
```

Step 13 After HBase replication is configured and data synchronization is enabled, check whether tables and data exist in the active cluster and whether the historical data needs to be synchronized to the standby cluster.

Run the **list** command to check the existing tables in the active cluster and run the **scan 'tableName'** command to check whether the tables contain historical data.

- If tables exist and data needs to be synchronized, go to [Step 14](#).
- If no, no further action is required.

Step 14 The HBase replication configuration does not support automatic synchronization of historical data in tables. You need to back up the historical data of the active cluster and then manually synchronize the historical data to the standby cluster.

Manual synchronization refers to the synchronization of a single table that is implemented by Export, distcp, and Import.

The process for manually synchronizing data of a single table is as follows:

1. Export table data from the active cluster.
hbase org.apache.hadoop.hbase.mapreduce.Export - Dhbase.mapreduce.include.deleted.rows=true *Table name Directory where the source data is stored*
Example: **hbase org.apache.hadoop.hbase.mapreduce.Export - Dhbase.mapreduce.include.deleted.rows=true t1 /user/hbase/t1**
2. Copy the data that has been exported to the standby cluster.
hadoop distcp *Directory for storing source data in the active cluster* **hdfs://** *ActiveNameNodeIP:9820/* *Directory for storing source data in the standby cluster*
ActiveNameNodeIP indicates the IP address of the active NameNode in the standby cluster.
Example: **hadoop distcp /user/hbase/t1 hdfs://192.168.40.2:9820/user/hbase/t1**
3. Import data to the standby cluster as the HBase table user of the standby cluster.
hbase org.apache.hadoop.hbase.mapreduce.Import - Dimport.bulk.output=Directory where the output data is stored in the standby cluster Table name Directory where the source data is stored in the standby cluster
hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles *Directory where the output data is stored in the standby cluster Table name*
For example, **hbase org.apache.hadoop.hbase.mapreduce.Import - Dimport.bulk.output=/user/hbase/output_t1 t1 /user/hbase/t1** and **hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles /user/hbase/output_t1 t1**

Add the replication relationship between the active and standby clusters.

Step 15 Run the following command on the HBase Shell to create the replication synchronization relationship between the active cluster and the standby cluster:

```
add_peer 'Standby cluster ID', CLUSTER_KEY => 'ZooKeeper address of the standby cluster',{HDFS_CONFS => true}
```

- *Standby cluster ID* indicates an ID for the active cluster to recognize the standby cluster. It is recommended that the ID contain letters and digits.
- The ZooKeeper address of the standby cluster includes the service IP address of ZooKeeper, the port for listening to client connections, and the HBase root directory of the standby cluster on ZooKeeper.
- **{HDFS_CONFS => true}** indicates that the default HDFS configuration of the active cluster will be synchronized to the standby cluster. This parameter is used for HBase of the standby cluster to access HDFS of the active cluster. If bulkload replication is disabled, you do not need to use this parameter.

Suppose the standby cluster ID is replication2 and the ZooKeeper address of the standby cluster is **192.168.40.2,192.168.40.3,192.168.40.4:2181:/hbase**.

- Run the **add_peer 'replication2',CLUSTER_KEY => '192.168.40.2,192.168.40.3,192.168.40.4:2181:/hbase',CONFIG => { "hbase.regionserver.kerberos.principal" => "<val>", "hbase.master.kerberos.principal" => "<val2>" }** command for a

security cluster and the `add_peer 'replication2',CLUSTER_KEY => '192.168.40.2,192.168.40.3,192.168.40.4:2181:/hbase'` command for a common cluster.

The `hbase.master.kerberos.principal` and `hbase.regionserver.kerberos.principal` parameters are the Kerberos users of HBase in the security cluster. You can search the `hbase-site.xml` file on the client for the parameter values. For example, if the client is installed in the `/opt/client` directory of the Master node, you can run the `grep "kerberos.principal" /opt/client/HBase/hbase/conf/hbase-site.xml -A1` command to obtain the principal of HBase. See the following figure.

Figure 8-1 Obtaining the principal of HBase

```
[root@hadoop102 ~]# grep "kerberos.principal" /opt/client/HBase/hbase/conf/hbase-site.xml -A1
<name>hbase.regionserver.kerberos.principal</name>
<value>hbase/hadoop.hadoop.com@HADOOP.COM</value>
--
<name>hbase.master.kerberos.principal</name>
<value>hbase/hadoop.hadoop.com@HADOOP.COM</value>
--
```

 **NOTE**

1. Obtain the ZooKeeper service IP address.
Log in to the MRS console, click the cluster name, and choose **Components > ZooKeeper > Instances** to obtain the ZooKeeper service IP address.
2. On the ZooKeeper service parameter configuration page, search for `clientPort`, which is the port for the client to connect to the server.
3. Run the `list_peers` command to check whether the replication relationship between the active and standby clusters is added. If the following information is displayed, the relationship is successfully added.

```
hbase(main):003:0> list_peers
PEER_ID CLUSTER_KEY ENDPOINT_CLASSNAME STATE REPLICATE_ALL NAMESPACES
TABLE_CFS BANDWIDTH SERIAL
replication2 192.168.0.13,192.168.0.177,192.168.0.25:2181:/hbase ENABLED true 0 false
```

Specify the data writing status for the active and standby clusters.

Step 16 On the HBase shell of the active cluster, run the following command to retain the data writing status:

set_clusterState_active

The command is run successfully if the following information is displayed:

```
hbase(main):001:0> set_clusterState_active
=> true
```

Step 17 On the HBase shell of the standby cluster, run the following command to retain the data read-only status:

set_clusterState_standby

The command is run successfully if the following information is displayed:

```
hbase(main):001:0> set_clusterState_standby
=> true
```

Enable the HBase replication function to synchronize data.

Step 18 Check whether a namespace exists in the HBase service instance of the standby cluster and the namespace has the same name as the namespace of the HBase table for which the replication function is to be enabled.

On the HBase shell of the standby cluster, run the **list_namespace** command to query the namespace.

- If the same namespace exists, go to [Step 19](#).
- If the same namespace does not exist, on the HBase shell of the standby cluster, run the following command to create a namespace with the same name and go to [Step 19](#):

```
create_namespace'ns1
```

Step 19 On the HBase shell of the active cluster, run the following command to enable real-time replication for tables in the active cluster. This ensures that modified data in the active cluster can be synchronized to the standby cluster in real time.

You can only synchronize data of one HTable at one time.

```
enable_table_replication 'Table name'
```

 **NOTE**

- If the standby cluster does not contain a table with the same name as the table for which real-time synchronization is to be enabled, the table is automatically created.
- If a table with the same name as the table for which real-time synchronization is to be enabled exists in the standby cluster, the structures of the two tables must be the same.
- If the encryption algorithm SMS4 or AES is configured for '*Table name*', the function for synchronizing data from the active cluster to the standby cluster cannot be enabled for the HBase table.
- If the standby cluster is offline or has tables with the same name but different structures, the replication function cannot be enabled.

If the standby cluster is offline, start it.

If the standby cluster has a table with the same name but different structure, modify the table structure to make it as the same as the table structure of the active cluster. On the HBase shell of the standby cluster, run the **alter** command to change the password by referring to the example.

Step 20 On the HBase shell of the active cluster, run the following command to enable the real-time replication function for the active cluster to synchronize the HBase permission table:

```
enable_table_replication 'hbase:acl'
```

 **NOTE**

After the permission of the active HBase source data table is modified, to ensure that the standby cluster can properly read data, modify the role permission for the standby cluster.

Check the data synchronization status for the active and standby clusters.

Step 21 Run the following command on the HBase client to check the synchronized data of the active and standby clusters. After the replication function is enabled, you can run this command to check whether the newly synchronized data is consistent.

```
hbase org.apache.hadoop.hbase.mapreduce.replication.VerifyReplication --  
starttime=Start time --endtime=End time Column family name ID of the standby  
cluster Table name
```

 NOTE

- The start time must be earlier than the end time.
- The value of **starttime** and **endtime** must be in the timestamp format. You need to run **date -d "2015-09-30 00:00:00" +%s** to change a common time format to a timestamp format. The command output is a 10-digit number (accurate to second), but HBase identifies a 13-digit number (accurate to millisecond). Therefore, you need to add three zeros (000) to the end of the command output.

Switch over active and standby clusters.

 NOTE

1. If the standby cluster needs to be switched over to the active cluster, reconfigure the active/standby relationship by referring to [Step 10](#) and [Step 15](#) to [Step 20](#).
2. Do not perform [Step 11](#) to [Step 14](#).

----End

Related Commands

Table 8-5 HBase replication

Operation	Command	Description
Set up the active/standby relationship.	add_peer <i>'Standby cluster ID', 'Standby cluster address'</i> Examples: add_peer '1', 'zk1,zk2,zk3:2181:/hbase' add_peer '1', 'zk1,zk2,zk3:2181:/hbase1'	Set up the relationship between the active cluster and the standby cluster. To enable bulkload replication, run the add_peer <i>'Standby cluster ID', CLUSTER_KEY => 'Standby cluster address'</i> command, configure hbase.replication.conf.dir , and manually copy the HBase client configuration file in the active cluster to all RegionServer nodes in the standby cluster. For details, see Step 4 to 11 .
Remove the active/standby relationship.	remove_peer <i>'Standby cluster ID'</i> Example: remove_peer '1'	Remove standby cluster information from the active cluster.
Query the active/standby relationship.	list_peers	Query standby cluster information (mainly Zookeeper information) in the active cluster.

Operation	Command	Description
Enable the real-time user table synchronization function.	enable_table_replication <i>'Table name'</i> Example: enable_table_replication 't1'	Synchronize user tables from the active cluster to the standby cluster.
Disable the real-time user table synchronization function.	disable_table_replication <i>'Table name'</i> Example: disable_table_replication 't1'	Do not synchronize user tables from the active cluster to the standby cluster.
Verify data of the active and standby clusters.	bin/hbase org.apache.hadoop.hbase.mapreduce.replication.VerifyReplication --starttime --endtime Column family name Standby cluster ID Table name	Verify whether data of the specified table is the same between the active cluster and the standby cluster. The description of the parameters in this command is as follows: <ul style="list-style-type: none"> • Start time: If start time is not specified, the default value 0 will be used. • End time: If end time is not specified, the time when the current operation is submitted will be used by default. • Table name: If a table name is not entered, all user tables for which the real-time synchronization function is enabled will be verified by default.
Switch the data writing status.	set_clusterState_active set_clusterState_standby	Specifies whether data can be written to the cluster HBase tables.

Operation	Command	Description
Add or update the active cluster HDFS configurations saved in the peer cluster.	<code>set_replication_hdfs_confs 'PeerId', {'key1' => 'value1', 'key2' => 'value2'}</code>	<p>Enable replication for data including bulkload data. When HDFS parameters are modified in the active cluster, the modification cannot be automatically synchronized to the standby cluster. You need to manually run the command to synchronize the changes. The affected parameters are as follows:</p> <ul style="list-style-type: none"> • fs.defaultFS • dfs.client.failover.proxy.provider.hacluster • dfs.client.failover.connection.retries.on.timeouts • dfs.client.failover.connection.retries <p>For example, if the value of fs.defaultFS is changed to hdfs://hacluster_sale, run the <code>set_replication_hdfs_confs '1', {'fs.defaultFS' => 'hdfs://hacluster_sale'}</code> command to synchronize the HDFS configuration to the standby cluster whose ID is 1.</p>

8.5 Configuring HBase Parameters

NOTE

The operations described in this section apply only to clusters of versions earlier than MRS 3.x.

If the default parameter settings of the MRS service cannot meet your requirements, you can modify the parameter settings as required.

Step 1 Go to the cluster details page and choose **Components**.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

Step 2 Choose **HBase > Service Configuration** and switch **Basic** to **All**. On the displayed HBase configuration page, modify parameter settings.

Table 8-6 HBase parameters

Parameter	Description	Value
hbase.regionserver.hfile.durable.sync	Whether to enable the HFile durability to make data persistence on disks. If this parameter is set to true , HBase performance is affected because each HFile is synchronized to disks by hadoop fsync when being written to HBase. This parameter exists only in MRS 1.9.2 or earlier.	Possible values are as follows: <ul style="list-style-type: none"> • true • false The default value is true .
hbase.regionserver.wal.durable.sync	Specifies whether to enable WAL file durability to make the WAL data persistence on disks. If this parameter is set to true , HBase performance is affected because each edited WAL file is synchronized to disks by hadoop fsync when being written to HBase. This parameter exists only in MRS 1.9.2 or earlier.	Possible values are as follows: <ul style="list-style-type: none"> • true • false The default value is true .

----End

8.6 Enabling Cross-Cluster Copy

Scenario

DistCp is used to copy the data stored on HDFS from a cluster to another cluster. DistCp depends on the cross-cluster copy function, which is disabled by default. This function needs to be enabled in both clusters.

This section describes how to enable cross-cluster copy.

Impact on the System

Yarn needs to be restarted to enable the cross-cluster copy function and cannot be accessed during the restart.

Prerequisites

The **hadoop.rpc.protection** parameter of the two HDFS clusters must be set to the same data transmission mode, which can be **privacy** (encryption enabled) or **authentication** (encryption disabled).

 NOTE

Go to the **All Configurations** page by referring to [Modifying Cluster Service Configuration Parameters](#) and search for **hadoop.rpc.protection**.

For versions earlier than MRS 3.x, choose **Components > HDFS > Service Configuration** on the cluster details page. Switch **Basic** to **All**, and search for **hadoop.rpc.protection**.

Procedure

- Step 1** Go to the **All Configurations** page of the Yarn service. For details, see [Modifying Cluster Service Configuration Parameters](#).

 NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- Step 2** In the navigation pane, choose **Yarn > Distcp**.

- Step 3** Set **haclusterX.remotenn1** of **dfs.namenode.rpc-address** to the service IP address and RPC port number of one NameNode instance of the peer cluster, and set **haclusterX.remotenn2** to the service IP address and RPC port number of the other NameNode instance of the peer cluster. Enter a value in the *IP address:port* format.

 NOTE

You can log in to FusionInsight Manager in MRS 3.x clusters, and choose **Cluster > Name of the desired cluster > Services > HDFS > Instance** to obtain the service IP address of the NameNode instance.

For versions earlier than MRS 3.x, on the cluster details page, choose **Components > HDFS > Instances** to obtain the service IP address of the NameNode instance.

dfs.namenode.rpc-address.haclusterX.remotenn1 and **dfs.namenode.rpc-address.haclusterX.remotenn2** do not distinguish active and standby NameNode instances. The default NameNode RPC port is 9820 and cannot be modified on MRS Manager.

For example, **10.1.1.1:9820** and **10.1.1.2:9820**.

- Step 4** Save the configuration. On the **Dashboard** tab page, and choose **More > Restart Service** to restart the Yarn service.

Operation succeeded is displayed. Click **Finish**. The Yarn service is started successfully.

- Step 5** Log in to the other cluster and repeat the preceding operations.

----End

8.7 Using the ReplicationSyncUp Tool

Prerequisites

1. Active and standby clusters have been installed and started.

2. Time is consistent between the active and standby clusters and the NTP service on the active and standby clusters uses the same time source.
3. When the HBase service of the active cluster is stopped, the ZooKeeper and HDFS services must be started and run.
4. ReplicationSyncUp must be run by the system user who starts the HBase process.
5. In security mode, ensure that the HBase system user of the standby cluster has the read permission on HDFS of the active cluster. This is because that it will update the ZooKeeper nodes and HDFS files of the HBase system.
6. When HBase of the active cluster is faulty, the ZooKeeper, file system, and network of the active cluster are still available.

Scenarios

The replication mechanism can use WAL to synchronize the state of a cluster with the state of another cluster. After HBase replication is enabled, if the active cluster is faulty, ReplicationSyncUp synchronizes incremental data from the active cluster to the standby cluster using the information from the ZooKeeper node. After data synchronization is complete, the standby cluster can be used as an active cluster.

Parameter Configuration

Parameter	Description	Default Value
hbase.replication.bulkload.enabled	Whether to enable the bulkload data replication function. The parameter value type is Boolean. To enable the bulkload data replication function, set this parameter to true for the active cluster.	false
hbase.replication.cluster.id	ID of the source HBase cluster. After the bulkload data replication is enabled, this parameter is mandatory and must be defined in the source cluster. The parameter value type is String.	-

Tool Usage

Run the following command on the client of the active cluster:

```
hbase org.apache.hadoop.hbase.replication.regionserver.ReplicationSyncUp -Dreplication.sleep.before.failover=1
```

NOTE

replication.sleep.before.failover indicates sleep time required for replication of the remaining data when RegionServer fails to start. You are advised to set this parameter to 1 second to quickly trigger replication.

Precautions

1. When the active cluster is stopped, this tool obtains the WAL processing progress and WAL processing queue from the ZooKeeper Node (RS znode) and copies the queues that are not copied to the standby cluster.
2. RegionServer of each active cluster has its own znode under the replication node of ZooKeeper in the standby cluster. It contains one znode of each peer cluster.
3. If RegionServer is faulty, each RegionServer in the active cluster receives a notification through the watcher and attempts to lock the znode of the faulty RegionServer, including its queues. The successfully created RegionServer transfers all queues to the znode of its own queue. After queues are transferred, they are deleted from the old location.
4. When the active cluster is stopped, ReplicationSyncUp synchronizes data between active and standby clusters using the information from the ZooKeeper node. In addition, WALs of the RegionServer znode will be moved to the standby cluster.

Restrictions and Limitations

If the standby cluster is stopped or the peer relationship is closed, the tool runs normally but the peer relationship cannot be replicated.

8.8 Using HIndex

8.8.1 Introduction to HIndex

Scenarios

HBase is a distributed storage database of the Key-Value type. Data in tables is sorted by dictionary based on row keys. If you query data by specifying a row key or scan data in a specific row key range, HBase can help you quickly locate the data to be read. In most cases, you need to query data whose column value is *XXX*. HBase provides the filter function to enable you to query data with a specific column value. All data is scanned in the sequence of row keys and is matched with the specific column value until the required data is found. To obtain the required data, the filter will scan some unnecessary data. As a result, the filter function cannot meet the requirements for high-performance, frequent queries.

HBase HIndex is designed to address these issues. HBase HIndex provides HBase with the capability of indexing based on specific column values, making queries faster.

 **NOTE**

- Rolling upgrade is not supported for index data.
- Composite index: You must add or delete all columns that participate in composite indexes. Otherwise, the data may be inconsistent.
- You should not explicitly configure any split policy to a data table where an index has been created.
- The mutation operations are not supported, such as increment and append.
- Index of the column with **maxVersions** greater than 1 is not supported.
- The value size of a column for which an index is added cannot exceed 32 KB.
- When the user data is deleted because TTL of the column family is invalid, the corresponding index data will not be deleted immediately. The index data will be deleted during major compaction.
- After an index is created, the TTL of the user column family must not be changed.
 - If the TTL of the column family is changed to a larger value after an index is created, delete the index and create one again. Otherwise, some generated index data may be deleted before the deletion of user data.
 - If the TTL of the column family is changed to a smaller value after an index is created, the index may be deleted after the deletion of user data.
- After disaster recovery is enabled for HBase tables, a secondary index is created in the active cluster and index table changes are not automatically synchronized to the standby cluster. To implement disaster recovery in this case, perform the following operations:
 1. After the secondary index is created in the active table, create a secondary index with the same schema and name using the same method in the standby cluster.
 2. In the active cluster, manually set **REPLICATION_SCOPE** of the index column family (default value: **d**) to **1**.

Parameter Configuration

1. Log in to the MRS console, click a cluster name and choose **Components**.
2. Go to the **All Configurations** page of the HBase service. For details, see [Modifying Cluster Service Configuration Parameters](#).
3. View parameters on the HBase configurations page.

Navigation Path	Parameter	Default Value	Description
HMaster > System	hbase.coprocessor.master.classes	org.apache.hadoop.hbase.hindex.server.master.HIndexMasterCoprocessor,com.xxx.hadoop.hbase.backup.services.RecoveryCoprocessor,org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor,org.apache.hadoop.hbase.security.access.ReadOnlyClusterEnabler,org.apache.hadoop.hbase.rsgroup.RSGroupAdminEndpoint	This coprocessor is used to handle Master-level operations after the HIndex function is enabled, for example, creating an index meta table, adding an index, and deleting an index, a table, and index metadata.
RegionServer > RegionServer	hbase.coprocessor.regionserver.classes	org.apache.hadoop.hbase.hindex.server.regionserver.HIndexRegionServerCoprocessor,org.apache.hadoop.hbase.JMXListener,org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor	This coprocessor is used to handle the operations that the Master delivers to RegionServer after the HIndex function is enabled.

Navigation Path	Parameter	Default Value	Description
	hbase.coprocessor.region.classes	org.apache.hadoop.hbase.hindex.server.regionserver.HIndexRegionCoprocessor,org.apache.hadoop.hbase.security.token.TokenProvider,com.xxx.hadoop.hbase.backup.services.RecoveryCoprocessor,org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor,org.apache.hadoop.hbase.security.access.SecureBulkLoadEndpoint,org.apache.hadoop.hbase.security.access.ReadOnlyClusterEnabler,org.apache.hadoop.hbase.coprocessor.MetaTableMetrics	This coprocessor is used to operate data in the Region after the HIndex function is enabled.

Navigation Path	Parameter	Default Value	Description
	hbase.coprocessor.wal.classes	org.apache.hadoop.hbase.hindex.server.regionserver.HIndexRegionServerCoprocesor,org.apache.hadoop.hbase.JMXListener,org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocesor	<p>This coprocessor is used for Replication, which filters index data to prevent the index data from being sent to the peer cluster. The peer cluster generates index data by itself.</p> <p>This parameter is supported only in versions earlier than MRS 3.x.</p>

 NOTE

1. The preceding default values need to be configured after the HBase HIndex function is enabled. In MRS clusters that support the HBase HIndex function, the values have been configured by default.
2. Ensure that the **master** parameter is configured on HMaster and the **region** and **regionserver** parameters are configured on RegionServer.

Related Interfaces

The APIs that use HIndex are in the **org.apache.hadoop.hbase.hindex.client.HIndexAdmin** class. The following table describes the related APIs.

Operation	API	Description	Precautions
Add an index.	addIndices()	Add an index to a table without data. Calling this API will add the specified index to a table but skips index data generation. Therefore, after this operation, the index cannot be used for the scanning and filtering operations. This API applies to scenarios where users want to add indexes in batches to tables that have a large amount of pre-existing user data. The specific operation is to use external tools such as the TableIndexer tool to build index data.	<ul style="list-style-type: none"> ● An index cannot be modified once it is added. To modify the index, you need to delete the old index and then create a new one. ● Do not create two indexes on the same column with different index names. Otherwise, storage and processing resources will be wasted. ● Indexes cannot be added to a system table. ● The append and increment operations are not supported when data is put into the index column. ● If any fault occurs on the client except DoNotRetryIOException, you need to try again. ● An index column family is selected from the following conditions in sequence based on availability: <ul style="list-style-type: none"> – Typically, the default index column family is d. However, if the value of hindex.default.family.name is set, the value will be used. – Symbol #, @, \$, or %

Operation	API	Description	Precautions
	addIndicesWithData()	Add an index to a table with data. This API is used to add the specified index to the table and create index data for the existing user data. Alternatively, the API can be called to generate an index and then generate index data when the user data is being stored. Therefore, after this operation, the index can be used for the scanning and filtering operations immediately.	<ul style="list-style-type: none"> - #0, @ 0, \$ 0, %0, #1, @ 1 ...to #255, @ 255, \$ 255, %255 - Throw exceptions. • You can use the HIndex TableIndexer tool to add indexes without building index data.
Delete an index.	dropIndices()	<p>This API is used to delete an index only. It deletes the specified index from a table but skips the corresponding index data. After this operation, the index cannot be used for the scanning and filtering operations. The cluster automatically deletes old index data during major compaction.</p> <p>This API applies to scenarios where a table contains a large amount of index data and dropIndicesWithData() is unavailable. In addition, you can use the TableIndexer tool to delete indexes and index data.</p>	<ul style="list-style-type: none"> • An index can be disabled when it is in the ACTIVE, INACTIVE, or DROPPING state. • If you use dropIndices() to delete an index, ensure that the index data has been deleted before the index is added to the table with the same index name (that is, major compaction has been completed). • If you delete an index, the following information will also be deleted: <ul style="list-style-type: none"> - A column family with an index - Any one of column families in a combination index • Indexes and index data can be deleted together

Operation	API	Description	Precautions
	dropIndicesWithData()	Delete index data. This API deletes the specified index and all index data corresponding to the index in a user table. After this operation, the index is completely deleted from the table and is no longer used for the scanning and filtering operations.	using the HIndexTableIndexer tool.
Enable/Disable an index.	disableIndices()	This API disables all indexes specified by a user so that they are no longer used for the scanning and filtering operations.	<ul style="list-style-type: none"> • An index can be enabled when the index is in the ACTIVE, INACTIVE, or BUILDING state. • An index can be disabled when the index is in the ACTIVE or INACTIVE state. • Before disabling an index, ensure that the index data is consistent with the user data. If no new data is added to the table when the index is disabled, the index data is consistent with the user data. • When enabling an index, you can use the TableIndexer tool to build index data to ensure data consistency.
	enableIndices()	This API enables all indexes specified by a user so that they can be used for the scanning and filtering operations.	
View the created index.	listIndices()	This API is used to list all indexes of a specified table.	N/A

Querying Data Based on Indexes

You can use a filter to query data in a user table with an index. The query result of a user table with a single or combination index is the same as that of a table without an index, but the table with an index provides higher data query performance than the table without an index.

The index usage rules are as follows:

- Scenario 1: A single index is created for one or more columns.
 - When this column is used for AND or OR query filtering, an index can improve query performance.
Example: `Filter_Condition(IndexCol1)AND / OR Filter_Condition(IndexCol2)`
 - When you use **Index Column AND Non-Index Column** for filtering in the query, the index can improve query performance.
Example: `Filter_Condition(IndexCol1)AND Filter_Condition(IndexCol2)AND Filter_Condition(NonIndexCol1)`
 - When you use **Index Column OR Non-Index Column** for filtering in the query but do not use an index, query performance will not be improved.
Example: `Filter_Condition(IndexCol1)AND / OR Filter_Condition(IndexCol2) OR Filter_Condition(NonIndexCol1)`
- Scenario 2: A combination index is created for multiple columns.
 - When the columns to be queried are all or part of the combination index and have the same order as the combination index, using the index improves query performance.
For example, create a combination index for C1, C2, and C3.
 - The index takes effect in the following situations:
`Filter_Condition(IndexCol1)AND Filter_Condition(IndexCol2)AND Filter_Condition(IndexCol3)`
`Filter_Condition(IndexCol1)AND Filter_Condition(IndexCol2)`
`FILTER_CONDITION(IndexCol1)`
 - The index does not take effect in the following situations:
`Filter_Condition(IndexCol2)AND Filter_Condition(IndexCol3)`
`Filter_Condition(IndexCol1)AND Filter_Condition(IndexCol3)`
`FILTER_CONDITION(IndexCol2)`
`FILTER_CONDITION(IndexCol3)`
 - When you use **Index Column AND Non-Index Column** for filtering in the query, the index can improve query performance.
Examples:
`Filter_Condition(IndexCol1)AND Filter_Condition(NonIndexCol1)`
`Filter_Condition(IndexCol1)AND Filter_Condition(IndexCol2)AND Filter_Condition(NonIndexCol1)`
 - When you use **Index Column OR Non-Index Column** for filtering in the query but do not use an index, query performance will not be improved.
Examples:
`Filter_Condition(IndexCol1)OR Filter_Condition(NonIndexCol1)`
`(Filter_Condition(IndexCol1)AND Filter_Condition(IndexCol2))OR(Filter_Condition(NonIndexCol1))`
 - When multiple columns are used for query, you can specify a value range for only the last column in the combination index and set other columns to specified values

For example, create a combination index for C1, C2, and C3. In a range query, only the value range of C3 can be set. The filter criteria are "C1 = XXX, C2 = XXX, and C3 = Value range."

Query Policy Selection

Use **SingleColumnValueFilter** or **SingleColumnRangeFilter**. It will provide the definite value **column_family:qualifierpair** (called **col1**) in filter criteria.

If **col1** is the first index column in the table, any index in the table can be a candidate index used during the query. The following provides an example:

If there is an index on **col1**, the index can be used as a candidate index because **col1** is the first and the only column of the index. If there is another index on **col1** and **col2**, you can consider this index as a candidate index because **col1** is the first column in the index list. However, if there is an index on **col2** and **col1**, this index cannot be used as a candidate index because the first column in the index list is not **col1**.

The most suitable method to use the index now is that when there are multiple candidate indexes, select the most suitable index for scanning data.

You can use the following solutions to learn how to select the best index policy.

- It is better to fully match.
Scenario: There are two indexes available, one for **col1&col2** and the other for **col1**.
In this scenario, the second index is better than the first one, because it scans less index data.
- If there are multiple candidate multi-column indexes, select an index with fewer index columns.
Scenario: There are two indexes available, one for **col1&col2** and the other for **col1&col2&col3**.
In this case, you had better use the index on **col1&col2**, because it scans less index data.

 NOTE

- During a query based on an index, the index state must be **ACTIVE**. You can call the **listIndices()** API to view the index state.
- To query the correct data based on the index, ensure the consistency between index data and user data.
- Run the following command to perform a complex query on the HBase shell client (assuming that an index has been created for the specified column):
scan 'tablename', {FILTER => "SingleColumnValueFilter(family, qualifier, compareOp, comparator, filterIfMissing, latestVersionOnly)"}
 Example: **scan 'test', {FILTER => "SingleColumnValueFilter('info', 'age', =, 'binary:26', true, true)"}**
 In the preceding scenario, if you want to save the row where no column is found in the result, you should not create any index in any such column, because if the column to be queried does not exist, the row will be filtered out when SCVF is used to scan the index columns. When the SCVF whose **filterIfMissing** is **false** (default value) scans non-index columns, rows where no column is queried will also be returned in the result. Therefore, to avoid inconsistent query results, you are advised to set **filterIfMissing** to **true** after creating SCVF for the index column.
- Run the following command on the HBase shell client to view the index data created for user data:
scan 'tablename', {ATTRIBUTES => {'FETCH_INDEX_DATA' => 'true'}}

8.8.2 Loading Index Data in Batches

Scenarios

HBase provides the ImportTsv&LoadIncremental tool to load user data in batches. HBase also provides the HIndexImportTsv tool to load both the user data and index data in batches. HIndexImportTsv inherits all functions of the HBase batch data loading tool ImportTsv. If a table is not created before the HIndexImportTsv tool is executed, an index will be created when the table is created, and index data is generated when user data is generated.

Procedure

1. Run the following commands to import data to HDFS:

```
hdfs dfs -mkdir <inputdir>  
hdfs dfs -put <local_data_file> <inputdir>
```

For example, define the data file **data.txt** as follows:

```
12005000201,Zhang San,Male,19,City a, Province a  
12005000202,Li Wanting,Female,23,City b, Province b  
12005000203,Wang Ming,Male,26,City c, Province c  
12005000204,Li Gang,Male,18,City d, Province d  
12005000205,Zhao Enru,Female,21,City e, Province e  
12005000206,Chen Long,Male,32,City f, Province f  
12005000207,Zhou Wei,Female,29,City g, Province g  
12005000208,Yang Yiwen,Female,30,City h, Province h  
12005000209,Xu Bing,Male,26,City i, Province i  
12005000210,Xiao Kai,Male,25,City j, Province j
```

Run the following commands:

```
hdfs dfs -mkdir /datadirImport  
hdfs dfs -put data.txt /datadirImport
```

- Go to HBase shell and run the following command to create the **bulkTable** table:

```
create 'bulkTable', {NAME => 'info',COMPRESSION => 'SNAPPY',  
DATA_BLOCK_ENCODING => 'FAST_DIFF'},{NAME=>'address'}
```

After the execution is complete, exit the HBase shell.

- Run the following commands to generate an HFile file (StoreFiles):

```
hbase org.apache.hadoop.hbase.index.mapreduce.HIndexImportTsv -  
Dimporttsv.separator=<separator>
```

```
-Dimporttsv.bulk.output=</path/for/output> -
```

```
Dindexspecs.to.add=<indexspecs> -Dimporttsv.columns=<columns>  
tableName <inputdir>
```

- **-Dimport.separator**: indicates a separator, for example, - **Dimport.separator=','**.
- **-Dimport.bulk.output=</path/for/output>**: indicates the output path of the execution result. You need to specify a path that does not exist.
- **<columns>**: Indicates the mapping of the imported data in a table, for example, - **Dimporttsv.columns=HBASE_ROW_KEY,info:name,info:gender,info:age, address:city,address:province.**
- **<tablename>**: Indicates the name of a table to be operated.
- **<inputdir>**: Indicates the directory where data is loaded in batches.
- **-Dindexspecs.to.add=<indexspecs>**: Indicates the mapping between an index name and a column, for example, - **Dindexspecs.to.add='index_bulk=>info:[age->String]'**. The index composition can be represented as follows:

```
indexNameN=>familyN :[columnQualifierN-> columnQualifierDataType],  
[columnQualifierM-> columnQualifierDataType];familyM:  
[columnQualifierO-> columnQualifierDataType]# indexNameN=>  
familyM: [columnQualifierO-> columnQualifierDataType]
```

Column qualifiers are separated by commas (,).

Example: "index1 => f1:[c1-> String],[c2-> String]"

Column families are separated by semicolons (;).

Example: "index1 => f1:[c1-> String],[c2-> String]; f2:[c3-> Long]"

Multiple indexes are separated by pound keys (#).

Example: "index1 => f1:[c1-> String],[c2-> String]; f2:[c3-> Long]#index2 => f2:[c3-> Long]"

The following data types are supported by columns.

Available data types are as follows: STRING, INTEGER, FLOAT, LONG, DOUBLE, SHORT, BYTE, CHAR

NOTE

Data types can also be transferred in lowercase.

For example, run the following command:

```
hbase org.apache.hadoop.hbase.index.mapreduce.HIndexImportTsv -  
Dimporttsv.separator=', ' -Dimporttsv.bulk.output=/dataOutput -  
Dindexspecs.to.add='index_bulk=>info:[age->String]' -
```


Dimporttsv.columns=HBASE_ROW_KEY,info:name,info:gender,info:age,address:city,address:province bulkTable /datadirImport/data.txt

Command output:

```
[root@shap000000406 opt]# hbase org.apache.hadoop.hbase.hindex.mapreduce.HIndexImportTsv -
Dimporttsv.separator=';' -Dimporttsv.bulk.output=/dataOutput -Dindexspecs.to.add='index_bulk=>info:
[age->String]' -
Dimporttsv.columns=HBASE_ROW_KEY,info:name,info:gender,info:age,address:city,address:province
bulkTable /datadirImport/data.txt
2018-05-08 21:29:16,059 INFO [main] mapreduce.HFileOutputFormat2: Incremental table bulkTable
output configured.
2018-05-08 21:29:16,069 INFO [main] client.ConnectionManager$HConnectionImplementation:
Closing master protocol: MasterService
2018-05-08 21:29:16,069 INFO [main] client.ConnectionManager$HConnectionImplementation:
Closing zookeeper sessionId=0x80007c2cb4fd5b4d
2018-05-08 21:29:16,072 INFO [main] zookeeper.ZooKeeper: Session: 0x80007c2cb4fd5b4d closed
2018-05-08 21:29:16,072 INFO [main-EventThread] zookeeper.ClientCnxn: EventThread shut down
for session: 0x80007c2cb4fd5b4d
2018-05-08 21:29:16,379 INFO [main] client.ConfiguredRMFailoverProxyProvider: Failing over to 147
2018-05-08 21:29:17,328 INFO [main] input.FileInputFormat: Total input files to process : 1
2018-05-08 21:29:17,413 INFO [main] mapreduce.JobSubmitter: number of splits:1
2018-05-08 21:29:17,430 INFO [main] Configuration.deprecation: io.bytes.per.checksum is
deprecated. Instead, use dfs.bytes-per-checksum
2018-05-08 21:29:17,687 INFO [main] mapreduce.JobSubmitter: Submitting tokens for job:
job_1525338489458_0002
2018-05-08 21:29:18,100 INFO [main] impl.YarnClientImpl: Submitted application
application_1525338489458_0002
2018-05-08 21:29:18,136 INFO [main] mapreduce.Job: The url to track the job: http://
shap000000407:8088/proxy/application_1525338489458_0002/
2018-05-08 21:29:18,136 INFO [main] mapreduce.Job: Running job: job_1525338489458_0002
2018-05-08 21:29:28,248 INFO [main] mapreduce.Job: Job job_1525338489458_0002 running in uber
mode : false
2018-05-08 21:29:28,249 INFO [main] mapreduce.Job: map 0% reduce 0%
2018-05-08 21:29:38,344 INFO [main] mapreduce.Job: map 100% reduce 0%
2018-05-08 21:29:51,421 INFO [main] mapreduce.Job: map 100% reduce 100%
2018-05-08 21:29:51,428 INFO [main] mapreduce.Job: Job job_1525338489458_0002 completed
successfully
2018-05-08 21:29:51,523 INFO [main] mapreduce.Job: Counters: 50
```

4. Run the following command to import the generated HFile to HBase:

**hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles </
path/for/output> <tablename>**

For example, run the following command:

**hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles /
dataOutput bulkTable**

Command output:

```
[root@shap000000406 opt]# hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles /
dataOutput bulkTable
2018-05-08 21:30:01,398 WARN [main] mapreduce.LoadIncrementalHFiles: Skipping non-directory
hdfs://hacluster/dataOutput/_SUCCESS
2018-05-08 21:30:02,006 INFO [LoadIncrementalHFiles-0] hfile.CacheConfig: Created cacheConfig:
CacheConfig:disabled
2018-05-08 21:30:02,006 INFO [LoadIncrementalHFiles-2] hfile.CacheConfig: Created cacheConfig:
CacheConfig:disabled
2018-05-08 21:30:02,006 INFO [LoadIncrementalHFiles-1] hfile.CacheConfig: Created cacheConfig:
CacheConfig:disabled
2018-05-08 21:30:02,085 INFO [LoadIncrementalHFiles-2] compress.CodecPool: Got brand-new
decompressor [.snappy]
2018-05-08 21:30:02,120 INFO [LoadIncrementalHFiles-0] mapreduce.LoadIncrementalHFiles: Trying
to load hfile=hdfs://hacluster/dataOutput/address/042426c252f74e859858c7877b95e510
first=12005000201 last=12005000210
2018-05-08 21:30:02,120 INFO [LoadIncrementalHFiles-2] mapreduce.LoadIncrementalHFiles: Trying
to load hfile=hdfs://hacluster/dataOutput/info/f3995920ae0247a88182f637aa031c49
first=12005000201 last=12005000210
2018-05-08 21:30:02,128 INFO [LoadIncrementalHFiles-1] mapreduce.LoadIncrementalHFiles: Trying
to load hfile=hdfs://hacluster/dataOutput/d/c53b252248af42779f29442ab84f86b8 first=\x00index_bulk
```

```
\x00\x00\x00\x00\x00\x00\x00\x0018\x00\x0012005000204 last=\x00index_bulk
\x00\x00\x00\x00\x00\x00\x00\x0032\x00\x0012005000206
2018-05-08 21:30:02,231 INFO [main] client.ConnectionManager$HConnectionImplementation:
Closing master protocol: MasterService
2018-05-08 21:30:02,231 INFO [main] client.ConnectionManager$HConnectionImplementation:
Closing zookeeper sessionId=0x81007c2cf0f55cc5
2018-05-08 21:30:02,235 INFO [main] zookeeper.ZooKeeper: Session: 0x81007c2cf0f55cc5 closed
2018-05-08 21:30:02,235 INFO [main-EventThread] zookeeper.ClientCnxn: EventThread shut down
for session: 0x81007c2cf0f55cc5
```

8.8.3 Using an Index Generation Tool

Scenarios

To quickly create indexes for user data, HBase provides the `TableIndexer` tool for you to create, add, and delete indexes using MapReduce functions. The application scenarios are as follows:

- You want to add an index for a specified column in a table where a large amount of data exists. However, if you use the `addIndicesWithData()` API to add an index, index data corresponding to the related user data will be generated, which is time-consuming. If you use `addIndices()` to create an index, index data corresponding to user data will not be generated. Therefore, to create index data for user data, you can use the `TableIndexer` tool to create an index.

- If the index data is inconsistent with the user data, the tool can be used to rebuild index data.

If you temporarily disable the index, put new data to the disabled index column, and then directly enable the index from the disabled state, index data and user data may be inconsistent. Therefore, you must rebuild all index data before using it again.

- You can use the `TableIndexer` tool to completely delete a large amount of existing index data from a user table.
- For user tables that do not have indexes, this tool allows you to add and build indexes at the same time.

How to Use

- **Adding a new index to a user table**

The command is as follows:

```
hbase org.apache.hadoop.hbase.hindex.mapreduce.TableIndexer -  
Dtablename.to.index=tablename -Dindexspecs.to.add='idx_0=>cf_0:[q_0-  
>string],[q_1];cf_1:[q_2],[q_3]#idx_1=>cf_1:[q_4]'
```

The following parameters are required.

- **tablename.to.index**: Indicates the name of a table for which an index is created.
- **indexspecs.to.add**: Indicates the mapping between the index name and the column in the corresponding user table.
- **scan.caching** (optional): Contains an integer value, indicating the number of cached rows to be transmitted to the scanner during data table scanning.

The parameters in the preceding command are described as follows:

- **idx_1**: Indicates an index name.
- **cf_0**: Indicates the name of a column family.
- **q_0**: Indicates the name of a column.
- **string**: Indicates a data type. The parameter value can be STRING, INTEGER, FLOAT, LONG, DOUBLE, SHORT, BYTE, or CHAR.

 **NOTE**

- The pound key (#) is used to separate indexes. The semicolon (;) is used to separate column families. The comma (,) is used to separate column qualifiers.
- The column name and its data type must be included in '[]'.
- Column names and their data types are separated by '->'.
- If the data type of a specific column is not specified, the default data type (string) is used.
- If **scan.caching** is not configured, the default value **1000** is used.
- The user table must exist.
- The index specified in the table must not exist.
- If a column family named **d** exists in the user table, you must use the TableIndexer tool to build index data.

After the preceding command is executed, the specified index is added to the table and is in INACTIVE state. This behavior is similar to the **addIndices()** API.

- **Creating index data for existing indexes in a user table**

The command is as follows:

```
hbase org.apache.hadoop.hbase.hindex.mapreduce.TableIndexer -  
Dtablename.to.index=tablename -Dindexnames.to.build='idx_0#idx_1'
```

The following parameters are required.

- **tablename.to.index**: Indicates the name of a table for which an index is created.
- **indexspecs.to.build**: Indicates an index name.
- **scan.caching** (optional): Contains an integer value, indicating the number of cached rows to be transmitted to the scanner during data table scanning.

The parameters in the preceding command are described as follows:

- **idx_1**: Indicates an index name.

 **NOTE**

- The pound key (#) is used to separate index names.
- If **scan.caching** is not configured, the default value **1000** is used.
- The user table must exist.

After the preceding command is executed, the specified index is set to the ACTIVE state. Users can use them when scanning data.

- **Deleting the existing indexes and their data from a user table**

The command is as follows:

```
hbase org.apache.hadoop.hbase.hindex.mapreduce.TableIndexer -  
Dtablename.to.index=tablename -Dindexnames.to.drop='idx_0#idx_1'
```

The following parameters are required.

- **tablename.to.index**: Indicates the name of a table for which an index is created.
- **indexnames.to.drop**: Indicates the name of the index that should be deleted with its data (must exist in the table).
- **scan.caching** (optional): Contains an integer value, indicating the number of cached rows to be transmitted to the scanner during data table scanning.

The parameters in the preceding command are described as follows:

- **idx_1**: Indicates an index name.

 **NOTE**

- The pound key (#) is used to separate index names.
- If **scan.caching** is not configured, the default value **1000** is used.
- The user table must exist.

After the preceding command is executed, the specified index is deleted from the table.

- **Adding new indexes to user tables and building data based on existing data**

The command is as follows:

```
hbase org.apache.hadoop.hbase.hindex.mapreduce.TableIndexer -  
Dtablename.to.index=tablename -Dindexspecs.to.add='idx_0 => cf_0:[q_0-  
> string],[q_1];cf_1:[q_2],[q_3]#idx_1 => cf_1:[q_4]' -  
Dindexnames.to.build='idx_0'
```

 **NOTE**

- The parameters are the same as the previous ones.
- The user table must exist.
- The indexes specified in **indexspecs.to.add** must not exist in the table.
- The index names specified in **indexnames.to.build** must exist in the table or be part of the value of **indexspecs.to.add**.

After the preceding command is executed, all indexes specified in **indexspecs.to.add** will be added to this table, and index data will be built for all specified indexes using **indexnames.to.build**.

8.8.4 Migrating Index Data

Scenario

The indexes used in MRS 1.7 or later are incompatible with secondary indexes used by HBase in earlier MRS versions. Therefore, you need to perform the following operations to migrate index data from an earlier version (MRS 1.5 or earlier) to MRS 1.7 or later.

Prerequisites

1. During data migration, the cluster of the old version must be MRS 1.5 or earlier, and the cluster of the new version must be MRS 1.7 or later.
2. Before data migration, you must have old index data.

3. A cross-cluster mutual trust relationship must be configured and the inter-cluster replication function must be enabled for a security cluster. For a common cluster, only the inter-cluster replication function needs to be enabled. For details, see [Configuring Cross-Cluster Mutual Trust Relationships](#) and [Enabling Cross-Cluster Copy](#).

Procedure

Migrate user data from an old cluster to a new cluster. To migrate data, you need to manually synchronize data of the old and new clusters in a single table by export, distcp, and import.

For example, the current old cluster has a user table (**t1**, index name: **idx_t1**) and its corresponding index table (**t1_idx**). Perform the following operations to migrate data.

1. Export table data from the old cluster.

```
hbase org.apache.hadoop.hbase.mapreduce.Export -Dhbase.mapreduce.include.deleted.rows=true
<tableName> <path/for/data>
```

- *<tableName>*: Indicates a table name, for example, **t1**.
- *<path/for/data>*: Indicates the path for storing source data, for example, **/user/hbase/t1**.

Example: **hbase org.apache.hadoop.hbase.mapreduce.Export -Dhbase.mapreduce.include.deleted.rows=true t1 /user/hbase/t1**

2. Copy the exported data to the new cluster as follows:

```
hadoop distcp <path/for/data> hdfs://ActiveNameNodeIP:9820/<path/for/newData>
```

- *<path/for/data>*: Indicates the path for storing source data in the old cluster, for example, **/user/hbase/t1**.
- *<path/for/newData>*: Indicates the path for storing source data in the new cluster, for example, **/user/hbase/t1**.

ActiveNameNodeIP indicates the IP address of the active NameNode in the new cluster.

Example: **hadoop distcp /user/hbase/t1 hdfs://192.168.40.2:9820/user/hbase/t1**

NOTE

- Manually copy the exported data to HDFS of the new cluster, for example, **/user/hbase/t1**.
3. Use the HBase table user of the new cluster to generate HFiles in the new cluster.

```
hbase org.apache.hadoop.hbase.mapreduce.Import -Dimport.bulk.output=<path/for/hfiles>
<tableName><path/for/newData>
```

- *<path/for/hfiles>*: Indicates the path of the HFiles generated in the new cluster, for example, **/user/hbase/output_t1**.
- *<tableName>*: Indicates a table name, for example, **t1**.
- *<path/for/newData>*: Indicates the path for storing source data in the new cluster, for example, **/user/hbase/t1**.

Example:

hbase org.apache.hadoop.hbase.mapreduce.Import -Dimport.bulk.output=/user/hbase/output_t1 t1 /user/hbase/t1

4. Import the generated HFiles to the table in the new cluster.

The command is as follows:

```
hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles <path/for/hfiles> <tableName>
```

- *<path/for/hfiles>*: Indicates the path of the HFiles generated in the new cluster, for example, **/user/hbase/output_t1**.
- *<tableName>*: Indicates a table name, for example, **t1**.

Example:

```
hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles /user/hbase/output_t1 t1
```

 NOTE

1. The preceding shows the process of migrating user data. You only need to perform the first three steps to migrate the index data of the old cluster and change the corresponding table name to an index table name (for example, **t1_idx**).
 2. Skip 4 when migrating index data.
5. Import index data to a table in the new cluster.
 - a. Add an index the same as that of the user table of the previous version to the user table of the new cluster (the user table cannot contain a column family named **d**).

The command is as follows:

```
hbase org.apache.hadoop.hbase.index.mapreduce.TableIndexer -Dtablename.to.index=<tableName> -Dindexspecs.to.add=<indexspecs>
```

- *-Dtablename.to.index=<tableName>*: Indicates a table name, for example, **-Dtablename.to.index=t1**.
- *-Dindexspecs.to.add=<indexspecs>*: Indicates the mapping between an index name and a column, for example, **-Dindexspecs.to.add='idx_t1=>info:[name->String]'**.

Example:

```
hbase org.apache.hadoop.hbase.index.mapreduce.TableIndexer -Dtablename.to.index=t1 -Dindexspecs.to.add='idx_t1=>info:[name->String]'
```

 NOTE

If a column family named **d** exists in the user table, you must use the TableIndexer tool to build index data.

- b. Run the LoadIncrementalHFiles tool to load the index data of the old cluster to a table in the new cluster.

The command is as follows:

```
hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles </path/for/hfiles> <tableName>
```

- *</path/for/hfiles>*: Indicates the path of index data on HDFS. The path is the index generation path specified in **-Dimport.bulk.output**, for example, **/user/hbase/output_t1_idx**.
- *<tableName>*: Indicates a table name of the new cluster, for example, **t1**.

Example:

`hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles / user/hbase/output_t1_idx t1`

8.9 Configuring HBase DR

Scenario

HBase disaster recovery (DR), a key feature that is used to ensure high availability (HA) of the HBase cluster system, provides the real-time remote DR function for HBase. HBase DR provides basic O&M tools, including tools for maintaining and re-establishing DR relationships, verifying data, and querying data synchronization progress. To implement real-time DR, back up data of an HBase cluster to another HBase cluster. DR in the HBase table common data writing and BulkLoad batch data writing scenarios is supported.

NOTE

This section applies to MRS 3.x or later.

Prerequisites

- The active and standby clusters are successfully installed and started, and you have the administrator permissions on the clusters.
- Ensure that the network connection between the active and standby clusters is normal and ports are available.
- If the active cluster is deployed in security mode and is not managed by one FusionInsight Manager, cross-cluster trust relationship has been configured for the active and standby clusters.. If the active cluster is deployed in normal mode, no cross-cluster mutual trust is required.
- Cross-cluster replication has been configured for the active and standby clusters.
- Time is consistent between the active and standby clusters and the NTP service on the active and standby clusters uses the same time source.
- Mapping relationships between the names of all hosts in the active and standby clusters and IP addresses have been configured in the hosts files of all the nodes in the active and standby clusters and of the node where the active cluster client resides.
- The network bandwidth between the active and standby clusters is determined based on service volume, which cannot be less than the possible maximum service volume.
- The MRS versions of the active and standby clusters must be the same.
- The scale of the standby cluster must be greater than or equal to that of the active cluster.

Constraints

- Although DR provides the real-time data replication function, the data synchronization progress is affected by many factors, such as the service volume in the active cluster and the health status of the standby cluster. In normal cases, the standby cluster should not take over services. In extreme

cases, system maintenance personnel and other decision makers determine whether the standby cluster takes over services according to the current data synchronization indicators.

- HBase clusters must be deployed in active/standby mode.
- Table-level operations on the DR table of the standby cluster are forbidden, such as modifying the table attributes and deleting the table. Misoperations on the standby cluster will cause data synchronization failure of the active cluster. As a result, table data in the standby cluster is lost.
- If the DR data synchronization function is enabled for HBase tables of the active cluster, the DR table structure of the standby cluster needs to be modified to ensure table structure consistency between the active and standby clusters during table structure modification.

Procedure

Configuring the common data writing DR parameters for the active cluster

- Step 1** Log in to Manager of the active cluster.
- Step 2** Choose **Cluster** > *Name of the desired cluster* > **Services** > **HBase** > **Configurations** and click **All Configurations**. The HBase configuration page is displayed.
- Step 3** (Optional) [Table 8-7](#) describes the optional configuration items during HBase DR. You can set the parameters based on the description or use the default values.

Table 8-7 Optional configuration items

Navigation Path	Parameter	Default Value	Description
HMaster > Performance	hbase.master.logcleaner.ttl	600000	Specifies the retention period of HLog. If the value is set to 604800000 (unit: millisecond), the retention period of HLog is 7 days.
	hbase.master.cleaner.interval	60000	Interval for the HMaster to delete historical HLog files. The HLog that exceeds the configured period will be automatically deleted. You are advised to set it to the maximum value to save more HLogs.
RegionServer > Replication	replication.source.size.capacity	16777216	Maximum size of edits, in bytes. If the edit size exceeds the value, HLog edits will be sent to the standby cluster.

Navigation Path	Parameter	Default Value	Description
	replication.source.nb.capacity	25000	Maximum number of edits, which is another condition for triggering HLog edits to be sent to the standby cluster. After data in the active cluster is synchronized to the standby cluster, the active cluster reads and sends data in HLog according to this parameter value. This parameter is used together with replication.source.size.capacity .
	replication.source.maxretriesmultiplier	10	Maximum number of retries when an exception occurs during replication.
	replication.source.sleepforretries	1000	Retry interval (Unit: ms)
	hbase.regionserver.replication.handler.count	6	Number of replication RPC server instances on RegionServer

Configuring the BulkLoad batch data writing DR parameters for the active cluster

Step 4 Determine whether to enable the BulkLoad batch data writing DR function.

If yes, go to [Step 5](#).

If no, go to [Step 8](#).

Step 5 Choose **Cluster** > *Name of the desired cluster* > **Services** > **HBase** > **Configurations** and click **All Configurations**. The HBase configuration page is displayed.

Step 6 Search for **hbase.replication.bulkload.enabled** and change its value to **true** to enable the BulkLoad batch data writing DR function.

Step 7 Search for **hbase.replication.cluster.id** and change the HBase ID of the active cluster. The ID is used by the standby cluster to connect to the active cluster. The value can contain uppercase letters, lowercase letters, digits, and underscores (_), and cannot exceed 30 characters.

Restarting the HBase service and install the client

Step 8 Click **Save**. In the displayed dialog box, click **OK**. Restart the HBase service.

Step 9 In the active and standby clusters, choose **Cluster** > *Name of the desired cluster* > **Service** > **HBase** > **More** > **Download Client** to download the client and install it.

Adding the DR relationship between the active and standby clusters

Step 10 Log in as user **hbase** to the HBase shell page of the active cluster.

Step 11 Run the following command on HBase Shell to create the DR synchronization relationship between the active cluster HBase and the standby cluster HBase.

```
add_peer 'Standby cluster ID', CLUSTER_KEY => "ZooKeeper service IP address in the standby cluster", CONFIG => {"hbase.regionserver.kerberos.principal" => "Standby cluster RegionServer principal", "hbase.master.kerberos.principal" => "Standby cluster HMaster principal"}
```

- The standby cluster ID indicates the ID for the active cluster to recognize the standby cluster. Enter an ID. The value can be specified randomly. Digits are recommended.
- The ZooKeeper address of the standby cluster includes the service IP address of ZooKeeper, the port for listening to client connections, and the HBase root directory of the standby cluster on ZooKeeper.
- Search for **hbase.master.kerberos.principal** and **hbase.regionserver.kerberos.principal** in the HBase **hbase-site.xml** configuration file of the standby cluster.

For example, to add the DR relationship between the active and standby clusters, run the **add_peer** '*Standby cluster ID*', **CLUSTER_KEY** =>

```
"192.168.40.2,192.168.40.3,192.168.40.4:24002:/hbase", CONFIG => {"hbase.regionserver.kerberos.principal" => "hbase/hadoop.hadoop.com@HADOOP.COM", "hbase.master.kerberos.principal" => "hbase/hadoop.hadoop.com@HADOOP.COM"}
```

Step 12 (Optional) If the BulkLoad batch data write DR function is enabled, the HBase client configuration of the active cluster must be copied to the standby cluster.

- Create the **/hbase/replicationConf/hbase.replication.cluster.id of the active cluster** directory in the HDFS of the standby cluster.
- HBase client configuration file, which is copied to the **/hbase/replicationConf/hbase.replication.cluster.id of the active cluster** directory of the HDFS of the standby cluster.

```
Example: hdfs dfs -put HBase/hbase/conf/core-site.xml HBase/hbase/conf/hdfs-site.xml HBase/hbase/conf/yarn-site.xml hdfs://NameNode IP.25000/hbase/replicationConf/source_cluster
```

Enabling HBase DR to synchronize data

Step 13 Check whether a naming space exists in the HBase service instance of the standby cluster and the naming space has the same name as the naming space of the HBase table for which the DR function is to be enabled.

- If the same namespace exists, go to [Step 14](#).
- If no, create a naming space with the same name in the HBase shell of the standby cluster and go to [Step 14](#).

Step 14 In the HBase shell of the active cluster, run the following command as user **hbase** to enable the real-time DR function for the table data of the active cluster to ensure that the data modified in the active cluster can be synchronized to the standby cluster in real time.

You can only synchronize the data of one HTable at a time.

`enable_table_replication 'table name'`

NOTE

- If the standby cluster does not contain a table with the same name as the table for which real-time synchronization is to be enabled, the table is automatically created.
- If a table with the same name as the table for which real-time synchronization is to be enabled exists in the standby cluster, the structures of the two tables must be the same.
- If the encryption algorithm SMS4 or AES is configured for '*Table name*', the function for synchronizing data from the active cluster to the standby cluster cannot be enabled for the HBase table.
- If the standby cluster is offline or has tables with the same name but different structures, the DR function cannot be enabled.
- If the DR data synchronization function is enabled for some Phoenix tables in the active cluster, the standby cluster cannot have common HBase tables with the same names as the Phoenix tables in the active cluster. Otherwise, the DR function fails to be enabled or the tables with the names in the standby cluster cannot be used properly.
- If the DR data synchronization function is enabled for Phoenix tables in the active cluster, you need to enable the DR data synchronization function for the metadata tables of the Phoenix tables. The metadata tables include SYSTEM.CATALOG, SYSTEM.FUNCTION, SYSTEM.SEQUENCE, and SYSTEM.STATS.
- If the DR data synchronization function is enabled for HBase tables of the active cluster, after adding new indexes to HBase tables, you need to manually add secondary indexes to DR tables in the standby cluster to ensure secondary index consistency between the active and standby clusters.
- The HBase multi-instance function also supports DR. You need to modify the parameters on the HBase service instance that corresponds to the standby cluster and run the commands on the clients of multiple instances. When adding the DR relationship, you need to select the directory, such as **hbase1**, for ZooKeeper of the standby cluster to store HBase multi-instance data.

Step 15 (Optional) If HBase does not use Ranger, run the following command as user **hbase** in the HBase shell of the active cluster to enable the real-time permission to control data DR function for the HBase tables in the active cluster.

```
enable_table_replication 'hbase:acl'
```

Creating Users

Step 16 Log in to FusionInsight Manager of the standby cluster, choose **System > Permission > Role > Create Role** to create a role, and add the same permission for the standby data table to the role based on the permission of the HBase source data table of the active cluster.

Step 17 Choose **System > Permission > User > Create** to create a user. Set the **User Type** to **Human-Machine** or **Machine-Machine** based on service requirements and add the user to the created role. Access the HBase DR data of the standby cluster as the newly created user.

NOTE

- After the permission of the active HBase source data table is modified, to ensure that the standby cluster can properly read data, modify the role permission for the standby cluster.
- If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. For details, see [Adding a Ranger Access Permission Policy for HBase](#).

Synchronizing the table data of the active cluster

Step 18 After HBase DR is configured and data synchronization is enabled, check whether tables and data exist in the active cluster and whether the historical data needs to be synchronized to the standby cluster.

- If yes, a table exists and data needs to be synchronized. Log in as the HBase table user to the node where the HBase client of the active cluster is installed and run the `kinit username` to authenticate the identity. The user must have the read and write permissions on tables and the execute permission on the `hbase:meta` table. Then go to [Step 19](#).
- If no, no further action is required.

Step 19 The HBase DR configuration does not support automatic synchronization of historical data in tables. You need to back up the historical data of the active cluster and then manually restore the historical data in the standby cluster.

Manual recovery refers to the recovery of a single table, which can be performed through Export, DistCp, or Import.

To manually recover a single table, perform the following steps:

1. Export table data from the active cluster.

hbase org.apache.hadoop.hbase.mapreduce.Export -
Dhbase.mapreduce.include.deleted.rows=true *Table name Directory where the source data is stored*

Example: **hbase org.apache.hadoop.hbase.mapreduce.Export -**
Dhbase.mapreduce.include.deleted.rows=true t1 /user/hbase/t1

2. Copy the data that has been exported to the standby cluster.

hadoop distcp *directory where the source data is stored on the active cluster*
hdfs://ActiveNameNodeIP:8020/directory where the source data is stored on the standby cluster

ActiveNameNodeIP indicates the IP address of the active NameNode in the standby cluster.

Example: **hadoop distcp /user/hbase/t1 hdfs://192.168.40.2:8020/user/hbase/t1**

3. Import data to the standby cluster as the HBase table user of the standby cluster.

On the HBase shell screen of the standby cluster, run the following command as user **hbase** to retain the data writing status:

set_clusterState_active

The command is run successfully if the following information is displayed:

```
hbase(main):001:0> set_clusterState_active  
=> true
```

hbase org.apache.hadoop.hbase.mapreduce.Import -
Dimport.bulk.output=Directory where the output data is stored in the standby cluster *Table name Directory where the source data is stored in the standby cluster*

hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles
Directory where the output data is stored in the standby cluster Table name

Example:

```
hbase(main):001:0> set_clusterState_active  
=> true
```

```
hbase org.apache.hadoop.hbase.mapreduce.Import -  
Dimport.bulk.output=/user/hbase/output_t1 t1 /user/hbase/t1  
hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles /user/  
hbase/output_t1 t1
```

Step 20 Run the following command on the HBase client to check the synchronized data of the active and standby clusters. After the DR data synchronization function is enabled, you can run this command to check whether the newly synchronized data is consistent.

```
hbase org.apache.hadoop.hbase.mapreduce.replication.VerifyReplication --  
starttime=Start time --endtime=End time Column family name ID of the standby  
cluster Table name
```

 **NOTE**

- The start time must be earlier than the end time.
- The values of **starttime** and **endtime** must be in the timestamp format. You need to run **date -d "2015-09-30 00:00:00" +%s** to change a common time format to a timestamp format.

Specify the data writing status for the active and standby clusters.

Step 21 On the HBase shell screen of the active cluster, run the following command as user **hbase** to retain the data writing status:

```
set_clusterState_active
```

The command is run successfully if the following information is displayed:

```
hbase(main):001:0> set_clusterState_active  
=> true
```

Step 22 On the HBase shell screen of the standby cluster, run the following command as user **hbase** to retain the data read-only status:

```
set_clusterState_standby
```

The command is run successfully if the following information is displayed:

```
hbase(main):001:0> set_clusterState_standby  
=> true
```

----End

Related Commands

Table 8-8 HBase DR

Operation	Command	Description
Set up a DR relationship.	<pre>add_peer '<i>Standby cluster ID</i>, CLUSTER_KEY => "<i>Standby cluster ZooKeeper service IP address</i>", CONFIG => {"hbase.regionserver.kerberos.principal" => "<i>Standby cluster RegionServer principal</i>", "hbase.master.kerberos.principal" => "<i>Standby cluster HMaster principal</i>"}</pre> <p>add_peer '1','zk1,zk2,zk3:2181:/hbase1' 2181: port number of ZooKeeper in the cluster</p>	<p>Set up the relationship between the active cluster and the standby cluster.</p> <p>If BulkLoad batch data write DR is enabled:</p> <ul style="list-style-type: none"> • Create the /hbase/replicationConf/hbase.replication.cluster.id of the active cluster directory in the HDFS of the standby cluster. • HBase client configuration file, which is copied to the /hbase/replicationConf/hbase.replication.cluster.id of the active cluster directory of the HDFS of the standby cluster.
Remove the DR relationship.	<pre>remove_peer '<i>Standby cluster ID</i>'</pre> <p>Example: remove_peer '1'</p>	Remove standby cluster information from the active cluster.
Querying the DR Relationship	list_peers	Query standby cluster information (mainly Zookeeper information) in the active cluster.
Enable the real-time user table synchronization function.	<pre>enable_table_replication '<i>Table name</i>'</pre> <p>Example: enable_table_replication 't1'</p>	Synchronize user tables from the active cluster to the standby cluster.
Disable the real-time user table synchronization function.	<pre>disable_table_replication '<i>Table name</i>'</pre> <p>Example: disable_table_replication 't1'</p>	Do not synchronize user tables from the active cluster to the standby cluster.

Operation	Command	Description
<p>Verify data of the active and standby clusters.</p>	<p>bin/hbase org.apache.hadoop.hbase.mapreduce.replication.VerifyReplication <i>--starttime=Start time --endtime=End time Column family name Standby cluster ID Table name</i></p>	<p>Verify whether data of the specified table is the same between the active cluster and the standby cluster.</p> <p>The description of the parameters in this command is as follows:</p> <ul style="list-style-type: none"> • Start time: If start time is not specified, the default value 0 will be used. • End time: If end time is not specified, the time when the current operation is submitted will be used by default. • Table name: If a table name is not entered, all user tables for which the real-time synchronization function is enabled will be verified by default.
<p>Switch the data writing status.</p>	<p>set_clusterState_active set_clusterState_standby</p>	<p>Specifies whether data can be written to the cluster HBase tables.</p>

Operation	Command	Description
Add or update the active cluster HDFS configurations saved in the peer cluster.	hdfs dfs -put -f HBase/hbase/conf/core-site.xml HBase/hbase/conf/hdfs-site.xml HBase/hbase/conf/yarn-site.xml hdfs://Standby cluster NameNode IP:PORT/hbase/replicationConf/Active cluster/hbase.replication.cluster.id	<p>Enable DR for data including bulkload data. When HDFS parameters are modified in the active cluster, the modification cannot be automatically synchronized from the active cluster to the standby cluster. You need to manually run the command to synchronize configuration. The affected parameters are as follows:</p> <ul style="list-style-type: none"> • fs.defaultFS • dfs.client.failover.proxy.provider.hacluster • dfs.client.failover.connection.retries.on.timeouts • dfs.client.failover.connection.retries <p>For example, change fs.defaultFS to hdfs://hacluster_sale, HBase client configuration file, which is copied to the /hbase/replicationConf/hbase.replication.cluster.id of the active cluster directory of the HDFS of the standby cluster.</p>

8.10 Performing an HBase DR Service Switchover

Scenario

The MRS cluster administrator can configure HBase cluster DR to improve system availability. If the active cluster in the DR environment is faulty and the connection to the HBase upper-layer application is affected, you need to configure the standby cluster information for the HBase upper-layer application so that the application can run in the standby cluster.

 **NOTE**

This section applies to MRS 3.x or later.

Impact on the System

After a service switchover, data written to the standby cluster is not synchronized to the active cluster by default. Add the active cluster is recovered, the data newly generated in the standby cluster needs to be synchronized to the active cluster by

backup and recovery. If automatic data synchronization is required, you need to switch over the active and standby HBase DR clusters.

Procedure

Step 1 Log in to FusionInsight Manager of the standby cluster.

Step 2 Download and install the HBase client.

Step 3 On the HBase client of the standby cluster, run the following command as user **hbase** to enable the data writing status in the standby cluster.

kinit hbase

hbase shell

set_clusterState_active

The command is run successfully if the following information is displayed:

```
hbase(main):001:0> set_clusterState_active  
=> true
```

Step 4 Check whether the original configuration files **hbase-site.xml**, **core-site.xml**, and **hdfs-site.xml** of the HBase upper-layer application are modified to adapt to the application running.

- If yes, update the related content to the new configuration file and replace the old configuration file.
- If no, use the new configuration file to replace the original configuration file of the HBase upper-layer application.

Step 5 Configure the network connection between the host where the HBase upper-layer application is located and the standby cluster.

NOTE

If the host where the client is installed is not a node in the cluster, configure network connections for the client to prevent errors when you run commands on the client.

1. Ensure that the host where the client is installed can communicate with the hosts listed in the **hosts** file in the directory where the client installation package is decompressed.
2. If the host where the client is located is not a node in the cluster, you need to set the mapping between the host name and the IP address (service plan) in the `/etc/hosts` file on the host. The host names and IP addresses must be mapped one by one.

Step 6 Set the time of the host where the HBase upper-layer application is located to be the same as that of the standby cluster. The time difference must be less than 5 minutes.

Step 7 Check the authentication mode of the active cluster.

- If the security mode is used, go to [Step 8](#).
- If the normal mode is used, no further action is required.

Step 8 Obtain the **keytab** and **krb5.conf** configuration files of the HBase upper-layer application user.

1. On FusionInsight Manager of the standby cluster, choose **System > Permission > User**.
2. Locate the row that contains the target user, click **More > Download Authentication Credential** in the **Operation** column, and download the **keytab** file to the local PC.
3. Decompress the package to obtain **user.keytab** and **krb5.conf**.

Step 9 Use the **user.keytab** and **krb5.conf** files to replace the original files in the HBase upper-layer application.

Step 10 Stop upper-layer applications.

Step 11 Determine whether to switch over the active and standby HBase clusters. If the switchover is not performed, data will not be synchronized.

- If yes, switch over the active and standby HBase DR clusters. For details, see [Performing an HBase DR Active/Standby Cluster Switchover](#). Then, go to [Step 12](#).
- If no, go to [Step 12](#).

Step 12 Start the upper-layer services.

----End

8.11 Performing an HBase DR Active/Standby Cluster Switchover

Scenario

The HBase cluster in the current environment is a DR cluster. Due to some reasons, the active and standby clusters need to be switched over. That is, the standby cluster becomes the active cluster, and the active cluster becomes the standby cluster.

NOTE

This section applies to MRS 3.x or later.

Impact on the System

After the active and standby clusters are switched over, data cannot be written to the original active cluster, and the original standby cluster becomes the active cluster to take over upper-layer services.

Procedure

Ensuring that upper-layer services are stopped

- Step 1** Ensure that the upper-layer services have been stopped. If not, perform operations by referring to [Performing an HBase DR Service Switchover](#).

Disabling the write function of the active cluster

Step 2 Download and install the HBase client.

Step 3 On the HBase client of the standby cluster, run the following command as user **hbase** to disable the data write function of the standby cluster:

```
kinit hbase
```

```
hbase shell
```

```
set_clusterState_standby
```

The command is run successfully if the following information is displayed:

```
hbase(main):001:0> set_clusterState_standby  
=> true
```

Checking whether the active/standby synchronization is complete

Step 4 Run the following command to ensure that the current data has been synchronized (SizeOfLogQueue=0 and SizeOfLogToReplicate=0 are required). If the values are not 0, wait and run the following command repeatedly until the values are 0.

```
status 'replication'
```

Disabling synchronization between the active and standby clusters

Step 5 Query all synchronization clusters and obtain the value of **PEER_ID**.

```
list_peers
```

Step 6 Delete all synchronization clusters.

```
remove_peer 'Standby cluster ID'
```

Example:

```
remove_peer '1'
```

Step 7 Query all synchronized tables.

```
list_replicated_tables
```

Step 8 Disable all synchronized tables queried in the preceding step.

```
disable_table_replication 'Table name'
```

Example:

```
disable_table_replication 't1'
```

Performing an active/standby switchover

Step 9 Reconfigure HBase DR. For details, see [Configuring HBase DR](#).

```
----End
```

8.12 Community BulkLoad Tool

The Apache HBase official website provides the function of importing data in batches. For details, see the description of the **Import** and **ImportTsv** tools at <http://hbase.apache.org/2.2/book.html#tools>.

8.13 Configuring the MOB

Scenario

In the actual application scenario, data in various sizes needs to be stored, for example, image data and documents. Data whose size is smaller than 10 MB can be stored in HBase. HBase can yield the best read-and-write performance for data whose size is smaller than 100 KB. If the size of data stored in HBase is greater than 100 KB or even reaches 10 MB and the same number of data files are inserted, the total data amount is large, causing frequent compaction and split, high CPU consumption, high disk I/O frequency, and low performance.

MOB data (100 KB to 10 MB data) is stored in a file system (such as the HDFS) in the HFile format. Files are centrally managed using the `expiredMobFileCleaner` and `Sweeper` tools. The addresses and size of files are stored in the HBase store as values. This greatly decreases the compaction and split frequency in HBase and improves performance.

The MOB function of HBase is enabled by default. For details about related configuration items, see [Table 8-9](#). To use the MOB function, you need to specify the MOB mode for storing data in the specified column family when creating a table or modifying table attributes.

NOTE

This section applies to MRS 3.x or later.

Configuration Description

To enable the HBase MOB function, you need to specify the MOB mode for storing data in the specified column family when creating a table or modifying table attributes.

Use code to declare that the MOB mode for storing data is used:

```
HColumnDescriptor hcd = new HColumnDescriptor("f");  
hcd.setMobEnabled(true);
```

Use code to declare that the MOB mode for storing data is used, the unit of `MOB_THRESHOLD` is byte:

```
hbase(main):009:0> create 't3',{NAME => 'd', MOB_THRESHOLD => '102400', IS_MOB => 'true'}  
0 row(s) in 0.3450 seconds  
=> Hbase::Table - t3  
hbase(main):010:0> describe 't3'  
Table t3 is ENABLED
```

```
t3
COLUMN FAMILIES DESCRIPTION

{NAME => 'd', MOB_THRESHOLD => '102400', VERSIONS => '1', KEEP_DELETED_CELLS => 'FALSE',
DATA_BLOCK_ENCODING => 'NONE',
TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW',
IN_MEMORY => 'false', IS_MOB => 'true', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE =>
'65536'}

1 row(s) in 0.0170 seconds
```

Navigation path for setting parameters:

On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **HBase** > **Configurations** > **All Configurations**. Enter a parameter name in the search box.

Table 8-9 Parameter description

Parameter	Description	Default Value
hbase.mob.file.cache.size	Size of the opened file handle cache. If this parameter is set to a large value, more file handles can be cached, reducing the frequency of opening and closing files. However, if this parameter is set to a large value, too many file handles will be opened. The default value is 1000 . This parameter is configured on the ResionServer.	1000
hbase.mob.cache.evict.period	Expiration time of cached MOB files in the MOB cache, in seconds.	3600
hbase.mob.cache.evict.remain.ratio	Ratio of the number of retained files after MOB cache reclamation to the number of cached files. hbase.mob.cache.evict.remain.ratio is an algorithm factor. When the number of cached MOB files reaches the product of hbase.mob.file.cache.size hbase.mob.cache.evict.remain.ratio , cache reclamation is triggered.	0.5
hbase.master.mob.ttl.cleaner.period	Interval for deleting expired files, in seconds. The default value is one day (86,400 seconds). NOTE If the validity period of an MOB file expires, that is, the file has been created for more than 24 hours, the MOB file will be deleted by the tool for deleting expired MOB files.	86400

8.14 Configuring Secure HBase Replication

Scenario

This topic provides the procedure to configure the secure HBase replication during cross-realm Kerberos setup in security mode.

Prerequisites

- Mapping for all the FQDNs to their realms should be defined in the Kerberos configuration file.
- The passwords and keytab files of **ONE.COM** and **TWO.COM** must be the same.

Procedure

Step 1 Create krbtgt principals for the two realms.

For example, if you have two realms called **ONE.COM** and **TWO.COM**, you need to add the following principals: **krbtgt/ONE.COM@TWO.COM** and **krbtgt/TWO.COM@ONE.COM**.

Add these two principals at both realms.

```
kadmin: addprinc -e "<enc_type_list>" krbtgt/ONE.COM@TWO.COM  
kadmin: addprinc -e "<enc_type_list>" krbtgt/TWO.COM@ONE.COM
```

NOTE

There must be at least one common keytab mode between these two realms.

Step 2 Add rules for creating short names in Zookeeper.

Dzookeeper.security.auth_to_local is a parameter of the ZooKeeper server process. Following is an example rule that illustrates how to add support for the realm called **ONE.COM**. The principal has two members (such as **service/instance@ONE.COM**).

```
Dzookeeper.security.auth_to_local=RULE:[2:$1@$0](.*@\QONE.COM\E$s)/@\QONE.COM\E$//DEFAULT
```

The above code example adds support for the **ONE.COM** realm in a different realm. Therefore, in the case of replication, you must add a rule for the master cluster realm in the slave cluster realm. **DEFAULT** is for defining the default rule.

Step 3 Add rules for creating short names in the Hadoop processes.

The following is the **hadoop.security.auth_to_local** property in the **core-site.xml** file in the slave cluster HBase processes. For example, to add support for the **ONE.COM** realm:

```
<property>  
<name>hadoop.security.auth_to_local</name>  
<value>RULE:[2:$1@$0](.*@\QONE.COM\E$s)/@\QONE.COM\E$//DEFAULT</value>  
</property>
```

 **NOTE**

If replication for bulkload data is enabled, then the same property for supporting the slave realm needs to be added in the **core-site.xml** file in the master cluster HBase processes.

Example:

```
<property>
<name>hadoop.security.auth_to_local</name>
<value>RULE:[2:$1@$0](.*@Q TWO.COM\E$)s/@\Q TWO.COM\E$//DEFAULT</value>
</property>
```

----End

8.15 Configuring Region In Transition Recovery Chore Service

Scenario

In a faulty environment, there are possibilities that a region may be stuck in transition for longer duration due to various reasons like slow region server response, unstable network, ZooKeeper node version mismatch. During region transition, client operation may not work properly as some regions will not be available.

Configuration

A chore service should be scheduled at HMaster to identify and recover regions that stay in the transition state for a long time.

The following table describes the parameters for enabling this function.

Table 8-10 Parameters

Parameter	Description	Default Value
hbase.region.assignment.auto.recovery.enabled	Configuration parameter used to enable/disable the region assignment recovery thread feature.	true

8.16 Using a Secondary Index

Scenario

HIndex enables HBase indexing based on specific column values, making the retrieval of data highly efficient and fast.

Constraints

- Column families are separated by semicolons (;).

- Columns and data types must be contained in square brackets ([]).
- The column data type is specified by using -> after the column name.
- If the column data type is not specified, the default data type (string) is used.
- The number sign (#) is used to separate two index details.
- The following is an optional parameter:
-Dscan.caching: number of cached rows when the data table is scanned.
The default value is set to 1000.
- Indexes are created for a single region to repair damaged indexes.
This function is not used to generate new indexes.

Procedure

Step 1 Install the HBase client. For details, see [Using an HBase Client](#).

Step 2 Go to the client installation directory, for example, `/opt/client`.

```
cd /opt/client
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 If the cluster is in security mode, run the following command to authenticate the user. In normal mode, user authentication is not required.

```
kinit Component service user
```

Step 5 Run the following command to access HIndex:

```
hbase org.apache.hadoop.hbase.hindex.mapreduce.TableIndexer
```

Table 8-11 Common HIndex commands

Description	Command
Add Index	TableIndexer-Dtablename.to.index=table1 - Dindexspecs.to.add='IDX1=>cf1:[q1->datatype],[q2],[q3];cf2: [q1->datatype],[q2->datatype]#IDX2=>cf1:[q5]'
Create Index	TableIndexer -Dtablename.to.index=table1 - Dindexnames.to.build='IDX1#IDX2'
Delete Index	TableIndexer -Dtablename.to.index=table1 - Dindexnames.to.drop='IDX1#IDX2'
Disable Index	TableIndexer -Dtablename.to.index=table1 - Dindexnames.to.disable='IDX1#IDX2'
Add and Create Index	TableIndexer -Dtablename.to.index=table1 - Dindexspecs.to.add='IDX1=>cf1:[q1->datatype],[q2],[q3];cf2: [q1->datatype],[q2->datatype]#IDX2=>cf1:[q5] - Dindexnames.to.build='IDX1'

Description	Command
Create Index for a Single Region	TableIndexer -Dtablename.to.index=table1 -Dregion.to.index=regionEncodedName -Dindexnames.to.build='IDX1#IDX2'

 NOTE

- **IDX1**: indicates the index name.
- **cf1**: indicates the column family name.
- **q1**: indicates the column name.
- **datatype**: indicates the data type, including String, Integer, Double, Float, Long, Short, Byte and Char.

----End

8.17 HBase Log Overview

Log Description

Log path: The default storage path of HBase logs is `/var/log/Bigdata/hbase/Role name`.

- HMaster: `/var/log/Bigdata/hbase/hm` (run logs) and `/var/log/Bigdata/audit/hbase/hm` (audit logs)
- RegionServer: `/var/log/Bigdata/hbase/rs` (run logs) and `/var/log/Bigdata/audit/hbase/rs` (audit logs)
- ThriftServer: `/var/log/Bigdata/hbase/ts2` (run logs, **ts2** is the instance name) and `/var/log/Bigdata/audit/hbase/ts2` (audit logs, **ts2** is the instance name)

Log archive rule: The automatic log compression and archiving function of HBase is enabled. By default, when the size of a log file exceeds 30 MB, the log file is automatically compressed. The naming rule of a compressed log file is as follows: `<Original log name>-<yyyy-mm-dd_hh-mm-ss>.[ID].log.zip` A maximum of 20 latest compressed files are reserved. The number of compressed files can be configured on the Manager portal.

Table 8-12 HBase log list

Type	Name	Description
Run logs	hbase-<SSH_USER>-<process_name>-<hostname>.log	HBase system log that records the startup time, startup parameters, and most logs generated when the HBase system is running.
	hbase-<SSH_USER>-<process_name>-<hostname>.out	Log that records the HBase running environment information.

Type	Name	Description
	<process_name>-<SSH_USER>-<DATE>-<PID>-gc.log	Log that records HBase junk collections.
	checkServiceDetail.log	Log that records whether the HBase service starts successfully.
	hbase.log	Log generated when the HBase service health check script and some alarm check scripts are executed.
	sendAlarm.log	Log that records alarms reported after execution of HBase alarm check scripts.
	hbase-haCheck.log	Log that records the active and standby status of HMaster
	stop.log	Log that records the startup and stop processes of HBase.
Audit logs	hbase-audit-<process_name>.log	Log that records HBase security audit.

Log Level

Table 8-13 describes the log levels supported by HBase. The priorities of log levels are FATAL, ERROR, WARN, INFO, and DEBUG in descending order. Logs whose levels are higher than or equal to the specified level are printed. The number of printed logs decreases as the specified log level increases.

Table 8-13 Log levels

Level	Description
FATAL	Logs of this level record fatal error information about the current event processing that may result in a system crash.
ERROR	Logs of this level record error information about the current event processing, which indicates that system running is abnormal.
WARN	Logs of this level record abnormal information about the current event processing. These abnormalities will not result in system faults.
INFO	Logs of this level record normal running status information about the system and events.

Level	Description
DEBUG	Logs of this level record the system information and system debugging information.

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page of the HBase service. For details, see [Modifying Cluster Service Configuration Parameters](#).
- Step 2** On the left menu bar, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.

 **NOTE**

The configurations take effect immediately without the need to restart the service.

----End

Log Formats

The following table lists the HBase log formats.

Table 8-14 Log formats

Type	Component	Format	Example
Run logs	HMaster	<yyyy-MM-dd HH:mm:ss,SSS> <Log Level> <Thread that generates the log> <Message in the log> <Location of the log event>	2020-01-19 16:04:53,558 INFO main env:HBASE_THRIFT_OPTS= org.apache.hadoop.hbase.util.ServerCommandLine.log ProcessInfo(ServerCommandLine.java:113)
	RegionServer	<yyyy-MM-dd HH:mm:ss,SSS> <Log Level> <Thread that generates the log> <Message in the log> <Location of the log event>	2020-01-19 16:05:18,589 INFO regionserver16020-SendThread(linux-k6da:2181) Client will use GSSAPI as SASL mechanism. org.apache.zookeeper.client.ZooKeeperSaslClient\$1.run(ZooKeeperSaslClient.java:285)

Type	Component	Format	Example
	ThriftServer	<yyyy-MM-dd HH:mm:ss,SSS> <Log Level> <Thread that generates the log> <Message in the log> <Location of the log event>	2020-02-16 09:42:55,371 INFO main loaded properties from hadoop-metrics2.properties org.apache.hadoop.metrics2.impl.MetricsConfig.loadFirst(MetricsConfig.java:111)
Audit logs	HMaster	<yyyy-MM-dd HH:mm:ss,SSS> <Log Level> <Thread that generates the log> <Message in the log> <Location of the log event>	2020-02-16 09:42:40,934 INFO master:linux-k6da:16000 Master: [master:linux-k6da:16000] start operation called. org.apache.hadoop.hbase.master.HMaster.run(HMaster.java:581)
	RegionServer	<yyyy-MM-dd HH:mm:ss,SSS> <Log Level> <Thread that generates the log> <Message in the log> <Location of the log event>	2020-02-16 09:42:51,063 INFO main RegionServer: [regionserver16020] start operation called. org.apache.hadoop.hbase.regionserver.HRegionServer.startRegionServer(HRegionServer.java:2396)
	ThriftServer	<yyyy-MM-dd HH:mm:ss,SSS> <Log Level> <Thread that generates the log> <Message in the log> <Location of the log event>	2020-02-16 09:42:55,512 INFO main thrift2 server start operation called. org.apache.hadoop.hbase.thrift2.ThriftServer.main(ThriftServer.java:421)

8.18 HBase Performance Tuning

8.18.1 Improving the BulkLoad Efficiency

Scenario

BulkLoad uses MapReduce jobs to directly generate files that comply with the internal data format of HBase, and then loads the generated StoreFiles to a running cluster. Compared with HBase APIs, BulkLoad saves more CPU and network resources.

ImportTSV is an HBase table data loading tool.

 NOTE

This section applies to MRS 3.x and later versions.

Prerequisites

When using BulkLoad, the output path of the file has been specified using the **Dimporttsv.bulk.output** parameter.

Procedure

Add the following parameter to the BulkLoad command when performing a batch loading task:

Table 8-15 Parameter for improving BulkLoad efficiency

Parameter	Description	Value
- Dimporttsv.map per.class	<p>The construction of key-value pairs is moved from the user-defined mapper to reducer to improve performance. The mapper only needs to send the original text in each row to the reducer. The reducer parses the record in each row and creates a key-value) pair.</p> <p>NOTE When this parameter is set to org.apache.hadoop.hbase.mapreduce.TsvImporterByteMapper, this parameter is used only when the batch loading command without the <i>HBASE_CELL_VISIBILITY OR HBASE_CELL_TTL</i> option is executed. The org.apache.hadoop.hbase.mapreduce.TsvImporterByteMapper provides better performance.</p>	<p>org.apache.hadoop.hbase.mapreduce.TsvImporterByteMapper and org.apache.hadoop.hbase.mapreduce.TsvImporterTextMapper</p>

8.18.2 Improving Put Performance

Scenario

In the scenario where a large number of requests are continuously put, setting the following two parameters to **false** can greatly improve the Put performance.

- **hbase.regionserver.wal.durable.sync**
- **hbase.regionserver.hfile.durable.sync**

When the performance is improved, there is a low probability that data is lost if three DataNodes are faulty at the same time. Exercise caution when configuring the parameters in scenarios that have high requirements on data reliability.

 NOTE

This section applies to MRS 3.x and later versions.

Procedure

Navigation path for setting parameters:

On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **HBase** > **Configurations** > **All Configurations**. Enter the parameter name in the search box, and change the value.

Table 8-16 Parameters for improving put performance

Parameter	Description	Value
hbase.wal.hsync	Specifies whether to enable WAL file durability to make the WAL data persistence on disks. If this parameter is set to true , the performance is affected because each WAL file is synchronized to the disk by the Hadoop fsync.	false
hbase.hfile.hsync	Specifies whether to enable the HFile durability to make data persistence on disks. If this parameter is set to true, the performance is affected because each Hfile file is synchronized to the disk by the Hadoop fsync.	false

8.18.3 Optimizing Put and Scan Performance

Scenario

HBase has many configuration parameters related to read and write performance. The configuration parameters need to be adjusted based on the read/write request loads. This section describes how to optimize read and write performance by modifying the RegionServer configurations.

 **NOTE**

This section applies to MRS 3.x and later versions.

Procedure

- JVM GC parameters
Suggestions on setting the RegionServer **GC_OPTS** parameter:
 - Set **-Xms** and **-Xmx** to the same value based on your needs. Increasing the memory can improve the read and write performance. For details, see the description of **hfile.block.cache.size** in [Table 8-18](#) and **hbase.regionserver.global.memstore.size** in [Table 8-17](#).

- Set **-XX:NewSize** and **-XX:MaxNewSize** to the same value. You are advised to set the value to **512M** in low-load scenarios and **2048M** in high-load scenarios.
- Set **X-XX:CMSInitiatingOccupancyFraction** to be less than and equal to 90, and it is calculated as follows: **100 x (hfile.block.cache.size + hbase.regionserver.global.memstore.size + 0.05)**.
- **-XX:MaxDirectMemorySize** indicates the non-heap memory used by the JVM. You are advised to set this parameter to **512M** in low-load scenarios and **2048M** in high-load scenarios.

 NOTE

The **-XX:MaxDirectMemorySize** parameter is not used by default. If you need to set this parameter, add it to the **GC_OPTS** parameter.

- Put parameters
RegionServer processes the data of the put request and writes the data to memstore and HLog.
 - When the size of memstore reaches the value of **hbase.hregion.memstore.flush.size**, memstore is updated to HDFS to generate HFiles.
 - Compaction is triggered when the number of HFiles in the column cluster of the current region reaches the value of **hbase.hstore.compaction.min**.
 - If the number of HFiles in the column cluster of the current region reaches the value of **hbase.hstore.blockingStoreFiles**, the operation of refreshing the memstore and generating HFiles is blocked. As a result, the put request is blocked.

Table 8-17 Put parameters

Parameter	Description	Default Value
hbase.wal.hsync	Indicates whether each WAL is persistent to disks. For details, see Improving Put Performance .	true
hbase.hfile.hsync	Indicates whether HFile write operations are persistent to disks. For details, see Improving Put Performance .	true

Parameter	Description	Default Value
hbase.hregion.memstore.flush.size	If the size of MemStore (unit: Byte) exceeds a specified value, MemStore is flushed to the corresponding disk. The value of this parameter is checked by each thread running hbase.server.thread.wakefrequency . It is recommended that you set this parameter to an integer multiple of the HDFS block size. You can increase the value if the memory is sufficient and the put load is heavy.	134217728
hbase.regionserver.global.memstore.size	Updates the size of all MemStores supported by the RegionServer before locking or forcible flush. It is recommended that you set this parameter to hbase.hregion.memstore.flush.size x Number of regions with active writes/ RegionServer GC -Xmx . The default value is 0.4 , indicating that 40% of RegionServer GC -Xmx is used.	0.4
hbase.hstore.flusher.count	Indicates the number of memstore flush threads. You can increase the parameter value in heavy-put-load scenarios.	2
hbase.regionserver.thread.compaction.small	Indicates the number of small compaction threads. You can increase the parameter value in heavy-put-load scenarios.	10

Parameter	Description	Default Value
hbase.hstore.blockingStoreFiles	If the number of HStoreFile files in a Store exceeds the specified value, the update of the HRegion will be locked until a compression is completed or the value of base.hstore.blockingWaitTime is exceeded. Each time MemStore is flushed, a StoreFile file is written into MemStore. Set this parameter to a larger value in heavy-put-load scenarios.	15

- Scan parameters

Table 8-18 Scan parameters

Parameter	Description	Default Value
hbase.client.scanner.timeout.period	Client and RegionServer parameters, indicating the lease timeout period of the client executing the scan operation. You are advised to set this parameter to an integer multiple of 60000 ms. You can set this parameter to a larger value when the read load is heavy. The unit is milliseconds.	60000
hfile.block.cache.size	Indicates the data cache percentage in the RegionServer GC -Xmx. You can increase the parameter value in heavy-read-load scenarios, in order to improve cache hit ratio and performance. It indicates the percentage of the maximum heap (-Xmx setting) allocated to the block cache of HFiles or StoreFiles.	When offheap is disabled, the default value is 0.25 . When offheap is enabled, the default value is 0.1 .

- Handler parameters

Table 8-19 Handler parameters

Parameter	Description	Default Value
hbase.regionserver.handler.count	Indicates the number of RPC server instances on RegionServer. The recommended value ranges from 200 to 400.	200
hbase.regionserver.metahandler.count	Indicates the number of program instances for processing prioritized requests. The recommended value ranges from 200 to 400.	200

8.18.4 Improving Real-time Data Write Performance

Scenario

Scenarios where data needs to be written to HBase in real time, or large-scale and consecutive put scenarios

 **NOTE**

This section applies to MRS 3.x and later versions.

Prerequisites

The HBase put or delete interface can be used to save data to HBase.

Procedure

- **Data writing server tuning**

Parameter portal:

Go to the **All Configurations** page of the HBase service. For details, see [Modifying Cluster Service Configuration Parameters](#).

Table 8-20 Configuration items that affect real-time data writing

Parameter	Description	Default Value
hbase.wal.hsync	Controls the synchronization degree when HLogs are written to the HDFS. If the value is true , HDFS returns only when data is written to the disk. If the value is false , HDFS returns when data is written to the OS cache. Set the parameter to false to improve write performance.	true
hbase.hfile.hsync	Controls the synchronization degree when HFiles are written to the HDFS. If the value is true , HDFS returns only when data is written to the disk. If the value is false , HDFS returns when data is written to the OS cache. Set the parameter to false to improve write performance.	true

Parameter	Description	Default Value
GC_OPTS	<p>You can increase HBase memory to improve HBase performance because read and write operations are performed in HBase memory. HeapSize and NewSize need to be adjusted. When you adjust HeapSize, set Xms and Xmx to the same value to avoid performance problems when JVM dynamically adjusts HeapSize. Set NewSize to 1/8 of HeapSize.</p> <ul style="list-style-type: none"> • HMaster: If HBase clusters enlarge and the number of Regions grows, properly increase the GC_OPTS parameter value of the HMaster. • RegionServer: A RegionServer needs more memory than an HMaster. If sufficient memory is available, increase the HeapSize value. <p>NOTE When the value of HeapSize for the active HMaster is 4 GB, the HBase cluster can support 100,000 regions. Empirically, each time 35,000 regions are added to the cluster, the value of HeapSize must be increased by 2 GB. It is recommended that the value of HeapSize for the active HMaster not exceed 32 GB.</p>	<ul style="list-style-type: none"> • HMaster -server - Xms4G - Xmx4G - XX:NewSize= 512M - XX:MaxNewSi ze=512M - XX:Metaspac eSize=128M - XX:MaxMetas paceSize=512 M -XX: +UseConcMa rkSweepGC - XX: +CMSParallel RemarkEnabl ed - XX:CMSInitiat ingOccupanc yFraction=65 -XX: +PrintGCDeta ils - Dsun.rmi.dgc. client.gcInter val=0x7FFFFFF FFFFFFFFFE - Dsun.rmi.dgc. server.gcInter val=0x7FFFFFF FFFFFFFFFE - XX:- OmitStackTra ceInFastThro w -XX: +PrintGCTim eStamps -XX: +PrintGCDate Stamps -XX: +UseGCLogFi leRotation - XX:NumberO fGCLogFiles= 10 - XX:GCLogFile Size=1M • Region Server

Parameter	Description	Default Value
		-server - Xms6G - Xmx6G - XX:NewSize= 1024M - XX:MaxNewSi ze=1024M - XX:Metaspac eSize=128M - XX:MaxMetas paceSize=512 M -XX: +UseConcMa rkSweepGC - XX: +CMSParallel RemarkEnabl ed - XX:CMSInitiat ingOccupanc yFraction=65 -XX: +PrintGC Deta ils - Dsun.rmi.dgc. client.gcInter val=0x7FFFFFF FFFFFFFE - Dsun.rmi.dgc. server.gcInter val=0x7FFFFFF FFFFFFFE - XX:- OmitStackTra ceInFastThro w -XX: +PrintGCTim eStamps -XX: +PrintGCDate Stamps -XX: +UseGCLogFi leRotation - XX:NumberO fGCLogFiles= 10 - XX:GCLogFile Size=1M

Parameter	Description	Default Value
hbase.regionserver.handler.count	<p>Indicates the number of RPC server instances started on RegionServer. If the parameter is set to an excessively large value, threads will compete fiercely. If the parameter is set to an excessively small value, requests will be waiting for a long time in RegionServer, reducing the processing capability. You can add threads based on resources.</p> <p>It is recommended that the value be set to 100 to 300 based on the CPU usage.</p>	200
hbase.hregion.max.filesize	<p>Indicates the maximum size of an HStoreFile, in bytes. If the size of any HStoreFile exceeds the value of this parameter, the managed Hregion is divided into two parts.</p>	10737418240
hbase.hregion.memstore.flush.size	<p>On the RegionServer, when the size of memstore that exists in memory of write operations exceeds memstore.flush.size, MemStoreFlusher performs the Flush operation to write the memstore to the corresponding store in the format of HFile.</p> <p>If RegionServer memory is sufficient and active Regions are few, increase the parameter value and reduce compaction times to improve system performance.</p> <p>The Flush operation may be delayed after it takes place. Write operations continue and memstore keeps increasing during the delay. The maximum size of memstore is: memstore.flush.size x hbase.hregion.memstore.block.multiplier. When the memstore size exceeds the maximum value, write operations are blocked. Properly increasing the value of hbase.hregion.memstore.block.multiplier can reduce the blocks and make performance become more stable. Unit: byte</p>	134217728

Parameter	Description	Default Value
<p>hbase.regionserver.global.memstore.size</p>	<p>Updates the size of all MemStores supported by the RegionServer before locking or forcible flush. On the RegionServer, the MemStoreFlusher thread performs the flush. The thread regularly checks memory occupied by write operations. When the total memory volume occupied by write operations exceeds the threshold, MemStoreFlusher performs the flush. Larger memstore will be flushed first and then smaller ones until the occupied memory is less than the threshold.</p> <p>Threshold = hbase.regionserver.global.memstore.size x hbase.regionserver.global.memstore.size.lower.limit x HBase_HEAPSIZE</p> <p>NOTE The sum of the parameter value and the value of hfile.block.cache.size cannot exceed 0.8, that is, memory occupied by read and write operations cannot exceed 80% of HeapSize, ensuring stable running of other operations.</p>	<p>0.4</p>

Parameter	Description	Default Value
hbase.hstore.blockingStoreFiles	<p>Check whether the number of files is larger than the value of hbase.hstore.blockingStoreFiles before you flush regions.</p> <p>If it is larger than the value of hbase.hstore.blockingStoreFiles, perform a compaction and configure hbase.hstore.blockingWaitTime to 90s to make the flush delay for 90s. During the delay, write operations continue and the memstore size keeps increasing and exceeds the threshold (memstore.flush.size x hbase.hregion.memstore.block.multiplier), blocking write operations. After compaction is complete, a large number of writes may be generated. As a result, the performance fluctuates sharply.</p> <p>Increase the value of hbase.hstore.blockingStoreFiles to reduce block possibilities.</p>	15
hbase.regionserver.thread.compaction.throttle	<p>The compression whose size is greater than the value of this parameter is executed by the large thread pool. The unit is bytes. Indicates a threshold of a total file size for compaction during a Minor Compaction. The total file size affects execution duration of a compaction. If the total file size is large, other compactions or flushes may be blocked.</p>	1610612736
hbase.hstore.compaction.min	<p>Indicates the minimum number of HStoreFiles on which minor compaction is performed each time. When the size of a file in a Store exceeds the value of this parameter, the file is compacted. You can increase the value of this parameter to reduce the number of times that the file is compacted. If there are too many files in the Store, read performance will be affected.</p>	6

Parameter	Description	Default Value
hbase.hstore.compaction.max	Indicates the maximum number of HStoreFiles on which minor compaction is performed each time. The functions of the parameter and hbase.hstore.compaction.max.size are similar. Both are used to limit the execution duration of one compaction.	10
hbase.hstore.compaction.max.size	If the size of an HFile is larger than the parameter value, the HFile will not be compacted in a Minor Compaction but can be compacted in a Major Compaction. The parameter is used to prevent HFiles of large sizes from being compacted. After a Major Compaction is forbidden, multiple HFiles can exist in a Store and will not be merged into one HFile, without affecting data access performance. The unit is byte.	9223372036854775807
hbase.hregion.majorcompaction	Main compression interval of all HStoreFile files in a region. The unit is milliseconds. Execution of Major Compactions consumes much system resources and will affect system performance during peak hours. If service updates, deletion, and reclamation of expired data space are infrequent, set the parameter to 0 to disable Major Compactions. If you must perform a Major Compaction to reclaim more space, increase the parameter value and configure the hbase.offpeak.end.hour and hbase.offpeak.start.hour parameters to make the Major Compaction be triggered in off-peak hours.	604800000

Parameter	Description	Default Value
<ul style="list-style-type: none"> hbase.regionserver.maxlogs hbase.regionserver.hlog.blocksize 	<ul style="list-style-type: none"> Indicates the threshold for the number of HLog files that are not flushed on a RegionServer. If the number of HLog files is greater than the threshold, the RegionServer forcibly performs flush operations. Indicates the maximum size of an HLog file. If the size of an HLog file is greater than the value of this parameter, a new HLog file is generated. The old HLog file is disabled and archived. <p>The two parameters determine the number of HLogs that are not flushed in a RegionServer. When the data volume is less than the total size of memstore, the flush operation is forcibly triggered due to excessive HLog files. In this case, you can adjust the values of the two parameters to avoid forcible flush. Unit: byte</p>	<ul style="list-style-type: none"> 32 134217728

- Data writing client tuning**

It is recommended that data is written in Put List mode if necessary, which greatly improves write performance. The length of each put list needs to be set based on the single put size and parameters of the actual environment. You are advised to do some basic tests before configuring parameters.

- Data table writing design optimization**

Table 8-21 Parameters affecting real-time data writing

Parameter	Description	Default Value
COMPRESSION	<p>The compression algorithm compresses blocks in HFiles. For compressible data, configure the compression algorithm to efficiently reduce disk I/Os and improve performance.</p> <p>NOTE Some data cannot be efficiently compressed. For example, a compressed figure can hardly be compressed again. The common compression algorithm is SNAPPY, because it has a high encoding/decoding speed and acceptable compression rate.</p>	NONE

Parameter	Description	Default Value
BLOCKSIZE	Different block sizes affect HBase data read and write performance. You can configure sizes for blocks in an HFile. Larger blocks have a higher compression rate. However, they have poor performance in random data read, because HBase reads data in a unit of blocks. Set the parameter to 128 KB or 256 KB to improve data write efficiency without greatly affecting random read performance. The unit is byte.	65536
IN_MEMORY	Whether to cache table data in the memory first, which improves data read performance. If you will frequently access some small tables, set the parameter.	false

8.18.5 Improving Real-time Data Read Performance

Scenario

HBase data needs to be read.

Prerequisites

The get or scan interface of HBase has been invoked and data is read in real time from HBase.

Procedure

- **Data reading server tuning**

Parameter portal:

Go to the **All Configurations** page of the HBase service. For details, see [Modifying Cluster Service Configuration Parameters](#).

Table 8-22 Configuration items that affect real-time data reading

Parameter	Description	Default Value
GC_OPTS	<p>You can increase HBase memory to improve HBase performance because read and write operations are performed in HBase memory.</p> <p>HeapSize and NewSize need to be adjusted. When you adjust HeapSize, set Xms and Xmx to the same value to avoid performance problems when JVM dynamically adjusts HeapSize. Set NewSize to 1/8 of HeapSize.</p> <ul style="list-style-type: none"> • HMaster: If HBase clusters enlarge and the number of Regions grows, properly increase the GC_OPTS parameter value of the HMaster. • RegionServer: A RegionServer needs more memory than an HMaster. If sufficient memory is available, increase the HeapSize value. <p>NOTE When the value of HeapSize for the active HMaster is 4 GB, the HBase cluster can support 100,000 regions. Empirically, each time 35,000 regions are added to the cluster, the value of HeapSize must be increased by 2 GB. It is recommended that the value of HeapSize for the active HMaster not exceed 32 GB.</p>	<p>For versions earlier than MRS 3.x:</p> <ul style="list-style-type: none"> • HMaster: <ul style="list-style-type: none"> -server -Xms2G -Xmx2G -XX:NewSize=256M -XX:MaxNewSize=256M -XX:MetaspaceSize=128M -XX:MaxMetaspaceSize=512M -XX:MaxDirectMemorySize=512M -XX:+UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled -XX:CMSInitiatingOccupancyFraction=65 -XX:+PrintGCDetails -Dsun.rmi.dgc.client.gclnterval=0x7FFFFFFF -Dsun.rmi.dgc.server.gclnterval=0x7FFFFFFF -XX:-OmitStackTraceInFastThread -XX:+PrintGCTi

Parameter	Description	Default Value
		<p>meStamps - XX: +PrintGCDateStamps - XX: +UseGCLogFileRotation - XX:NumberOfGCLogFiles=10 - XX:GCLogFileSize=1M</p> <ul style="list-style-type: none"> ● RegionServer: <ul style="list-style-type: none"> -server - Xms4G - Xmx4G - XX:NewSize=512M - XX:MaxNewSize=512M - XX:MetaspaceSize=128M - XX:MaxMetaspaceSize=512M - XX:MaxDirectMemorySize=512M - XX: +UseConcMarkSweepGC -XX: +CMSParallelRemarkEnabled - XX:CMSInitiatingOccupancyFraction=65 -XX: +PrintGCDetails - Dsun.rmi.dgc.client.gcInterval=0x7FFFFFFF - FFE -

Parameter	Description	Default Value
		<p>Dsun.rmi.dgc.server.gcinterval=0x7FFFFFFF -XX:-OmitStackTraceInFastThrow -XX:+PrintGCTimeStamps -XX:+PrintGCDateStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -XX:GCLogFileSize=1M</p> <p>For MRS 3.x or later:</p> <ul style="list-style-type: none"> <p>HMaster</p> <p>-server -Xms4G -Xmx4G -XX:NewSize=512M -XX:MaxNewSize=512M -XX:MetaspaceSize=128M -XX:MaxMetaspaceSize=512M -XX:+UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled -XX:CMSInitiatingOccupancyFraction=65 -XX:</p>

Parameter	Description	Default Value
		<p>+PrintGCDe tails - Dsun.rmi.dg c.client.gcln terval=0x7F FFFFFFFF FFE - Dsun.rmi.dg c.server.gcln terval=0x7F FFFFFFFF FFE -XX:- OmitStackTr aceInFastTh row -XX: +PrintGCTi meStamps - XX: +PrintGC Da teStamps - XX: +UseGCLog FileRotation - XX:Number OfGCLogFil es=10 - XX:GCLogFil eSize=1M</p> <ul style="list-style-type: none"> ● Region Server <p>-server - Xms6G - Xmx6G - XX:NewSize =1024M - XX:MaxNew Size=1024M - - XX:Metaspa ceSize=128 M - XX:MaxMet aspaceSize= 512M -XX: +UseConcM arkSweepG C -XX: +CMSParall elRemarkEn</p>

Parameter	Description	Default Value
		abled - XX:CMSInitiatingOccupancyFraction=65 -XX: +PrintGCDetails - Dsun.rmi.dgc.client.gcInterval=0x7FFFFFFF - Dsun.rmi.dgc.server.gcInterval=0x7FFFFFFF -XX:- OmitStackTraceInFastThread -XX: +PrintGCTimeStamps -XX: +PrintGCDateStamps -XX: +UseGCLogFileRotation - XX:NumberOfGCLogFiles=10 - XX:GCLogFileSize=1M
hbase.regionserver.handler.count	Indicates the number of requests that RegionServer can process concurrently. If the parameter is set to an excessively large value, threads will compete fiercely. If the parameter is set to an excessively small value, requests will be waiting for a long time in RegionServer, reducing the processing capability. You can add threads based on resources. It is recommended that the value be set to 100 to 300 based on the CPU usage.	200

Parameter	Description	Default Value
hfile.block.cache.size	HBase cache sizes affect query efficiency. Set cache sizes based on query modes and query record distribution. If random query is used to reduce the hit ratio of the buffer, you can reduce the buffer size.	When offheap is disabled, the default value is 0.25 . When offheap is enabled, the default value is 0.1 .

 **NOTE**

If read and write operations are performed at the same time, the performance of the two operations affects each other. If flush and compaction operations are frequently performed due to data writes, a large number of disk I/O operations are occupied, affecting read performance. If a large number of compaction operations are blocked due to write operations, multiple HFiles exist in the region, affecting read performance. Therefore, if the read performance is unsatisfactory, you need to check whether the write configurations are proper.

- **Data reading client tuning**

When scanning data, you need to set **caching** (the number of records read from the server at a time. The default value is **1**.). If the default value is used, the read performance will be extremely low.

If you do not need to read all columns of a piece of data, specify the columns to be read to reduce network I/O.

If you only need to read the row key, add a filter (FirstKeyOnlyFilter or KeyOnlyFilter) that only reads the row key.

- **Data table reading design optimization**

Table 8-23 Parameters affecting real-time data reading

Parameter	Description	Default Value
COMPRESSION	The compression algorithm compresses blocks in HFiles. For compressible data, configure the compression algorithm to efficiently reduce disk I/Os and improve performance. NOTE Some data cannot be efficiently compressed. For example, a compressed figure can hardly be compressed again. The common compression algorithm is SNAPPY, because it has a high encoding/decoding speed and acceptable compression rate.	NONE

Parameter	Description	Default Value
BLOCKSIZE	Different block sizes affect HBase data read and write performance. You can configure sizes for blocks in an HFile. Larger blocks have a higher compression rate. However, they have poor performance in random data read, because HBase reads data in a unit of blocks. Set the parameter to 128 KB or 256 KB to improve data write efficiency without greatly affecting random read performance. The unit is byte.	65536
DATA_BLOCK_ENCODING	Encoding method of the block in an HFile. If a row contains multiple columns, set FAST_DIFF to save data storage space and improve performance.	NONE

8.18.6 Optimizing JVM Parameters

Scenario

When the number of clusters reaches a certain scale, the default settings of the Java virtual machine (JVM) cannot meet the cluster requirements. In this case, the cluster performance deteriorates or the clusters may be unavailable. Therefore, JVM parameters must be properly configured based on actual service conditions to improve the cluster performance.

Procedure

Navigation path for setting parameters:

The JVM parameters related to the HBase role must be configured in the **hbase-env.sh** file in the `/${BIGDATA_HOME}/FusionInsight_HD_*/install/FusionInsight-HBase-2.2.3/hbase/conf/` directory.

Each role has JVM parameter configuration variables, as shown in [Table 8-24](#).

Table 8-24 HBase-related JVM parameter configuration variables

Variable	Affected Role
HBASE_OPTS	All roles of HBase
SERVER_GC_OPTS	All roles on the HBase server, such as Master and RegionServer
CLIENT_GC_OPTS	Client process of HBase

Variable	Affected Role
HBASE_MASTER_OPTS	Master of HBase
HBASE_REGIONSERVER_OPTS	RegionServer of HBase
HBASE_THRIFT_OPTS	Thrift of HBase

Configuration example:

```
export HADOOP_NAMENODE_OPTS="-Dhadoop.security.logger=${HADOOP_SECURITY_LOGGER:-INFO,RFAS} -Dhdfs.audit.logger=${HDFS_AUDIT_LOGGER:-INFO,NullAppender} $HADOOP_NAMENODE_OPTS"
```

8.19 Common Issues About HBase

8.19.1 Why Does a Client Keep Failing to Connect to a Server for a Long Time?

Question

A HBase server is faulty and cannot provide services. In this case, when a table operation is performed on the HBase client, why is the operation suspended and no response is received for a long time?

Answer

Problem Analysis

When the HBase server malfunctions, the table operation request from the HBase client is tried for several times and times out. The default timeout value is **Integer.MAX_VALUE (2147483647 ms)**. The table operation request is retired constantly during such a long period of time and is suspended at last.

Solution

The HBase client provides two configuration items to configure the retry and timeout of the client. [Table 8-25](#) describes them.

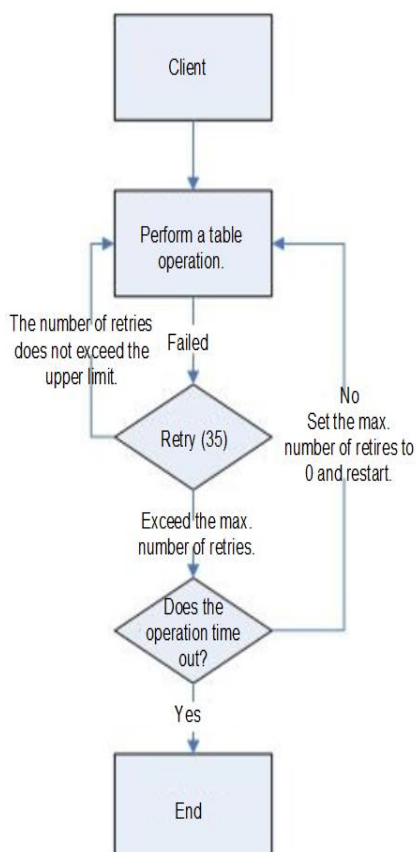
Set the following parameters in the *Client installation path*/HBase/hbase/conf/**hbase-site.xml** configuration file:

Table 8-25 Configuration parameters of retry and timeout

Parameter	Description	Default Value
hbase.client.operation.timeout	Client operation timeout period You need to manually add the information to the configuration file.	2147483647 ms
hbase.client.retries.number	Maximum retry times supported by all retryable operations.	35

Figure 8-2 describes the working principles of retry and timeout.

Figure 8-2 Process for HBase client operation retry timeout



The process indicates that a suspension occurs if the preceding parameters are not configured based on site requirements. It is recommended that a proper timeout period be set based on scenarios. If the operation takes a long time, set a long timeout period. If the operation takes a short time, set a short timeout period. The number of retries can be set to **(hbase.client.retries.number)*60*1000(ms)**. The timeout period can be slightly greater than **hbase.client.operation.timeout**.

8.19.2 Operation Failures Occur in Stopping BulkLoad On the Client

Question

Why submitted operations fail by stopping BulkLoad on the client during BulkLoad data importing?

Answer

When BulkLoad is enabled on the client, a partitioner file is generated and used to demarcate the range of Map task data inputting. The file is automatically deleted when BulkLoad exists on the client. In general, if all map tasks are enabled and running, the termination of BulkLoad on the client does not cause the failure of submitted operations. However, due to the retry and speculative execution mechanism of Map tasks, a Map task is performed again if failures of the Reduce task to download the data of the completed Map task exceed the limit. In this case, if BulkLoad already exists on the client, the retry Map task fails and the operation failure occurs because the partitioner file is missing. Therefore, it is recommended not to stop BulkLoad on the client during BulkLoad data importing.

8.19.3 Why May a Table Creation Exception Occur When HBase Deletes or Creates the Same Table Consecutively?

Question

When HBase consecutively deletes and creates the same table, why may a table creation exception occur?

Answer

Execution process: Disable Table > Drop Table > Create Table > Disable Table > Drop Table > And more

1. When a table is disabled, HMaster sends an RPC request to RegionServer, and RegionServer brings the region offline. When the time required for closing a region on RegionServer exceeds the timeout period for HBase HMaster to wait for the region to enter the RIT state, HMaster considers that the region is offline by default. Actually, the region may be in the flush memstore phase.
2. After an RPC request is sent to close a region, HMaster checks whether all regions in the table are offline. If the closure times out, HMaster considers that the regions are offline and returns a message indicating that the regions are successfully closed.
3. After the closure is successful, the data directory corresponding to the HBase table is deleted.
4. After the table is deleted, the data directory is recreated by the region that is still in the flush memstore phase.
5. When the table is created again, the **temp** directory is copied to the HBase data directory. However, the HBase data directory is not empty. As a result, when the HDFS rename API is called, the data directory changes to the last

layer of the **temp** directory and is appended to the HBase data directory, for example, **\$rootDir/data/\$nameSpace/\$tableName/\$tableName**. In this case, the table fails to be created.

Troubleshooting Method

When this problem occurs, check whether the HBase data directory corresponding to the table exists. If it exists, rename the directory.

The HBase data directory consists of **\$rootDir/data/\$nameSpace/\$tableName**, for example, **hdfs://hacluster/hbase/data/default/TestTable**. **\$rootDir** is the HBase root directory, which can be obtained by configuring **hbase.rootdir.perms** in **hbase-site.xml**. The **data** directory is a fixed directory of HBase. **\$nameSpace** indicates the nameSpace name. **\$tableName** indicates the table name.

8.19.4 Why Other Services Become Unstable If HBase Sets up A Large Number of Connections over the Network Port?

Question

Why other services become unstable if HBase sets up a large number of connections over the network port?

Answer

When the OS command **lsof** or **netstat** is run, it is found that many TCP connections are in the CLOSE_WAIT state and the owner of the connections is HBase RegionServer. This can cause exhaustion of network ports or limit exceeding of HDFS connections, resulting in instability of other services. The HBase CLOSE_WAIT phenomenon is the HBase mechanism.

The reason why HBase CLOSE_WAIT occurs is as follows: HBase data is stored in the HDFS as HFile, which can be called StoreFiles. HBase functions as the client of the HDFS. When HBase creates a StoreFile or starts loading a StoreFile, it creates an HDFS connection. When the StoreFile is created or loaded successfully, the HDFS considers that the task is completed and transfers the connection close permission to HBase. However, HBase may choose not to close the connection to ensure real-time response; that is, HBase may maintain the connection so that it can quickly access the corresponding data file upon request. In this case, the connection is in the CLOSE_WAIT, which indicates that the connection needs to be closed by the client.

When a StoreFile will be created: HBase executes the Flush operation.

When Flush is executed: The data written by HBase is first stored in memstore. The Flush operation is performed only when the usage of memstore reaches the threshold or the **flush** command is run to write data into the HDFS.

To resolve the issue, use either of the following methods:

Because of the HBase connection mechanism, the number of StoreFiles must be restricted to reduce the occupation of HBase ports. This can be achieved by triggering HBase's the compaction action, that is, HBase file merging.

Method 1: On HBase shell client, run **major_compact**.

Method 2: Compile HBase client code to invoke the compact method of the HBaseAdmin class to trigger HBase's compaction action.

If the HBase port occupation issue cannot be resolved through compact, it indicates that the HBase usage has reached the bottleneck. In such a case, you are advised to perform the following:

- Check whether the initial number of Regions configured in the table is appropriate.
- Check whether useless data exists.

If useless data exists, delete the data to reduce the number of storage files for the HBase. If the preceding conditions are not met, then you need to consider a capacity expansion.

8.19.5 Why Does the HBase BulkLoad Task (One Table Has 26 TB Data) Consisting of 210,000 Map Tasks and 10,000 Reduce Tasks Fail?

Question

The HBase bulkLoad task (a single table contains 26 TB data) has 210,000 maps and 10,000 reduce tasks (in MRS 3.x or later), and the task fails.

Answer

ZooKeeper I/O bottleneck observation methods:

1. On the monitoring page of Manager, check whether the number of ZooKeeper requests on a single node exceeds the upper limit.
2. View ZooKeeper and HBase logs to check whether a large number of I/O Exception Timeout or SocketTimeout Exception exceptions occur.

Optimization suggestions:

1. Change the number of ZooKeeper instances to 5 or more. You are advised to set **peerType** to **observer** to increase the number of observers.
2. Control the number of concurrent maps of a single task or reduce the memory for running tasks on each node to lighten the node load.
3. Upgrade ZooKeeper data disks, such as SSDs.

8.19.6 How Do I Restore a Region in the RIT State for a Long Time?

Question

How do I restore a region in the RIT state for a long time?

Answer

Log in to the HMaster WebUI, choose **Procedure & Locks** in the navigation tree, and check whether any process ID is in the **Waiting** state. If yes, run the following command to release the procedure lock:

```
hbase hbck -j /opt/client/HBase/hbase/tools/hbase-hbck2-*.jar bypass -o pid
```

Check whether the state is in the **Bypass** state. If the procedure on the UI is always in **RUNNABLE(Bypass)** state, perform an active/standby switchover. Run the **assigns** command to bring the region online again.

```
hbase hbck -j /opt/client/HBase/hbase/tools/hbase-hbck2-*.jar assigns -o  
regionName
```

8.19.7 Why Does HMaster Exits Due to Timeout When Waiting for the Namespace Table to Go Online?

Question

Why does HMaster exit due to timeout when waiting for the namespace table to go online?

Answer

During the HMaster active/standby switchover or startup, HMaster performs WAL splitting and region recovery for the RegionServer that failed or was stopped previously.

Multiple threads are running in the background to monitor the HMaster startup process.

- **TableNamespaceManager**
This is a help class, which is used to manage the allocation of namespace tables and monitoring table regions during HMaster active/standby switchover or startup. If the namespace table is not online within the specified time (**hbase.master.namespace.init.timeout**, which is 3,600,000 ms by default), the thread terminates HMaster abnormally.
- **InitializationMonitor**
This is an initialization thread monitoring class of the primary HMaster, which is used to monitor the initialization of the primary HMaster. If a thread fails to be initialized within the specified time (**hbase.master.initializationmonitor.timeout**, which is 3,600,000 ms by default), the thread terminates HMaster abnormally. If **hbase.master.initializationmonitor.haltontimeout** is started, the default value is **false**.

During the HMaster active/standby switchover or startup, if the **WAL hlog** file exists, the WAL splitting task is initialized. If the WAL hlog splitting task is complete, it initializes the table region allocation task.

HMaster uses ZooKeeper to coordinate log splitting tasks and valid RegionServers and track task development. If the primary HMaster exits during the log splitting task, the new primary HMaster attempts to resend the unfinished task, and RegionServer starts the log splitting task from the beginning.


```
{
  "call": "Scan(org.apache.hadoop.hbase.protobuf.generated.ClientProtos$ScanRequest)",
  "starttimems": 1450118746297,
  "responseSize": 416,
  "method": "Scan",
  "processingtimems": 11425,
  "client": "10.91.8.175:41182",
  "queuetimems": 1746,
  "class": "HRegionServer"
} | org.apache.hadoop.hbase.ipc.RpcServer.logResponse(RpcServer.java:2221)
2015-12-15 02:47:21,668 | INFO | LruBlockCacheStatsExecutor | totalSize=7.54 GB, freeSize=369.52 MB, max=7.90 GB, blockCount=406107, accesses=35400006, hits=16803205, hitRatio=47.47%, , cachingAccesses=31864266, cachingHits=14806045, cachingHitsRatio=46.47%, evictions=17654, evicted=16642283, evictedPerRun=942.69189453125 | org.apache.hadoop.hbase.io.hfile.LruBlockCache.logStats(LruBlockCache.java:858)
2015-12-15 02:52:21,668 | INFO | LruBlockCacheStatsExecutor | totalSize=7.51 GB, freeSize=395.34 MB, max=7.90 GB, blockCount=403080, accesses=35685793, hits=16933684, hitRatio=47.45%, , cachingAccesses=32150053, cachingHits=14936524, cachingHitsRatio=46.46%, evictions=17684, evicted=16800617, evictedPerRun=950.046142578125 | org.apache.hadoop.hbase.io.hfile.LruBlockCache.logStats(LruBlockCache.java:858)
```

Answer

The memory allocated to RegionServer is too small and the number of Regions is too large. As a result, the memory is insufficient during the running, and the server responds slowly to the client. Modify the following memory allocation parameters in the **hbase-site.xml** configuration file of RegionServer:

Table 8-26 RegionServer memory allocation parameters

Parameter	Description	Default Value
GC_OPTS	Initial memory and maximum memory allocated to RegionServer in startup parameters.	-Xms8G -Xmx8G
hfile.block.cache.size	Percentage of the maximum heap (-Xmx setting) allocated to the block cache of HFiles or StoreFiles.	When offheap is disabled, the default value is 0.25 . When offheap is enabled, the default value is 0.1 .

8.19.9 Why Modified and Deleted Data Can Still Be Queried by Using the Scan Command?

Question

Why modified and deleted data can still be queried by using the **scan** command?

```
scan '<table_name>',{FILTER=>"SingleColumnValueFilter('<column_family>','column',=,'binary:<value>')"} }
```

Answer

Because of the scalability of HBase, all values specific to the versions in the queried column are all matched by default, even if the values have been modified

or deleted. For a row where column matching has failed (that is, the column does not exist in the row), the HBase also queries the row.

If you want to query only the new values and rows where column matching is successful, you can use the following statement:

```
scan '<table_name>',  
{FILTER=>"SingleColumnValueFilter('<column_family>','column',='binary:<value>',true,true)"}
```

This command can filter all rows where column query has failed. It queries only the latest values of the current data in the table; that is, it does not query the values before modification or the deleted values.

NOTE

The related parameters of **SingleColumnValueFilter** are described as follows:

SingleColumnValueFilter(final byte[] family, final byte[] qualifier, final CompareOp compareOp, ByteArrayComparable comparator, final boolean filterIfMissing, final boolean latestVersionOnly)

Parameter description:

- family: family of the column to be queried.
- qualifier: column to be queried.
- compareOp: comparison operation, such as = and >.
- comparator: target value to be queried.
- filterIfMissing: whether a row is filtered out if the queried column does not exist. The default value is false.
- latestVersionOnly: whether values of the latest version are queried. The default value is false.

8.19.10 Why "java.lang.UnsatisfiedLinkError: Permission denied" exception thrown while starting HBase shell?

Question

Why "java.lang.UnsatisfiedLinkError: Permission denied" exception thrown while starting HBase shell?

Answer

During HBase shell execution JRuby create temporary files under **java.io.tmpdir** path and default value of **java.io.tmpdir** is **/tmp**. If NOEXEC permission is set to /tmp directory then HBase shell start will fail with "java.lang.UnsatisfiedLinkError: Permission denied" exception.

So "java.io.tmpdir" must be set to a different path in HBASE_OPTS/CLIENT_GC_OPTS if NOEXEC is set to /tmp directory.

8.19.11 When does the RegionServers listed under "Dead Region Servers" on HMaster WebUI gets cleared?

Question

When does the RegionServers listed under "Dead Region Servers" on HMaster WebUI gets cleared?

Answer

When an online RegionServer goes down abruptly, it is displayed under "Dead Region Servers" in the HMaster WebUI. When dead RegionServer restarts and reports back to HMaster successfully, the "Dead Region Servers" in the HMaster WebUI gets cleared.

The "Dead Region Servers" is also gets cleared, when the HMaster failover operation is performed successfully.

In cases when an Active HMaster hosting some regions is abruptly killed, Backup HMaster will become the new Active HMater and displays previous Active HMaster as dead RegionServer.

8.19.12 Why Are Different Query Results Returned After I Use Same Query Criteria to Query Data Successfully Imported by HBase bulkload?

Question

If the data to be imported by HBase bulkload has identical rowkeys, the data import is successful but identical query criteria produce different query results.

Answer

Data with an identical rowkey is loaded into HBase in the order in which data is read. The data with the latest timestamp is considered to be the latest data. By default, data is not queried by timestamp. Therefore, if you query for data with an identical rowkey, only the latest data is returned.

While data is being loaded by bulkload, the memory processes the data into HFiles quickly, leading to the possibility that data with an identical rowkey has a same timestamp. In this case, identical query criteria may produce different query results.

To avoid this problem, ensure that the same data file does not contain identical rowkeys while you are creating tables or loading data.

8.19.13 What Should I Do If I Fail to Create Tables Due to the FAILED_OPEN State of Regions?

Question

What should I do if I fail to create tables due to the FAILED_OPEN state of Regions?

Answer

If a network, HDFS, or Active HMaster fault occurs during the creation of tables, some Regions may fail to go online and therefore enter the FAILED_OPEN state. In this case, tables fail to be created.

The tables that fail to be created due to the preceding mentioned issue cannot be repaired. To solve this problem, perform the following operations to delete and re-create the tables:

1. Run the following command on the cluster client to repair the state of the tables:
hbase hbck -fixTableStates
2. Enter the HBase shell and run the following commands to delete the tables that fail to be created:
***truncate* '<table_name>'**
***disable* '<table_name>'**
***drop* '<table_name>'**
3. Create the tables using the recreation command.

8.19.14 How Do I Delete Residual Table Names in the /hbase/table-lock Directory of ZooKeeper?

Question

In security mode, names of tables that failed to be created are unnecessarily retained in the table-lock node (default directory is /hbase/table-lock) of ZooKeeper. How do I delete these residual table names?

Answer

Perform the following steps:

1. On the client, run the `kinit` command as the `hbase` user to obtain a security certificate.
2. Run the ***hbase zkcli*** command to launch the ZooKeeper Command Line Interface (zkCLI).
3. Run the ***ls /hbase/table*** command on the zkCLI to check whether the table name of the table that fails to be created exists.
 - If the table name exists, no further operation is required.
 - If the table name does not exist, run ***ls /hbase/table-lock*** to check whether the table name of the table fail to be created exist. If the table name exists, run the ***delete /hbase/table-lock/<table>*** command to delete the table name. In the ***delete /hbase/table-lock/<table>*** command, ***<table>*** indicates the residual table name.

8.19.15 Why Does HBase Become Faulty When I Set a Quota for the Directory Used by HBase in HDFS?

Question

Why does HBase become faulty when I set quota for the directory used by HBase in HDFS?

Answer

The flush operation of a table is to write memstore data to HDFS.

If the HDFS directory does not have sufficient disk space quota, the flush operation will fail and the region server will stop.

```
Caused by: org.apache.hadoop.hdfs.protocol.DSQuotaExceededException: The DiskSpace quota of /hbase/
data/<namespace>/<tableName> is exceeded: quota = 1024 B = 1 KB but diskspace consumed = 402655638
B = 384.00 MB
?at
org.apache.hadoop.hdfs.server.namenode.DirectoryWithQuotaFeature.verifyStoragespaceQuota(DirectoryWit
hQuotaFeature.java:211)
?at
org.apache.hadoop.hdfs.server.namenode.DirectoryWithQuotaFeature.verifyQuota(DirectoryWithQuotaFeatu
re.java:239)
?at org.apache.hadoop.hdfs.server.namenode.FSDirectory.verifyQuota(FSDirectory.java:882)
?at org.apache.hadoop.hdfs.server.namenode.FSDirectory.updateCount(FSDirectory.java:711)
?at org.apache.hadoop.hdfs.server.namenode.FSDirectory.updateCount(FSDirectory.java:670)
?at org.apache.hadoop.hdfs.server.namenode.FSDirectory.addBlock(FSDirectory.java:495)
```

In the preceding exception, the disk space quota of the **/hbase/data/<namespace>/<tableName>** table is 1 KB, but the memstore data is 384.00 MB. Therefore, the flush operation fails and the region server stops.

When the region server is terminated, HMaster replays the WAL file of the terminated region server to restore data. The disk space quota is limited. As a result, the replay operation of the WAL file fails, and the HMaster process exits unexpectedly.

```
2016-07-28 19:11:40,352 | FATAL | MASTER_SERVER_OPERATIONS-10-91-9-131:16000-0 | Caught throwable
while processing event M_SERVER_SHUTDOWN |
org.apache.hadoop.hbase.master.HMaster.abort(HMaster.java:2474)
java.io.IOException: failed log splitting for 10-91-9-131,16020,1469689987884, will retry
?at org.apache.hadoop.hbase.master.handler.ServerShutdownHandler.resubmit(ServerShutdownHandler.java:
365)
?at org.apache.hadoop.hbase.master.handler.ServerShutdownHandler.process(ServerShutdownHandler.java:
220)
?at org.apache.hadoop.hbase.executor.EventHandler.run(EventHandler.java:129)
?at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
?at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
?at java.lang.Thread.run(Thread.java:745)
Caused by: java.io.IOException: error or interrupted while splitting logs in [hdfs://hacluster/hbase/WALs/<RS-
Hostname>,<RS-Port>,<startcode>-splitting] Task = installed = 6 done = 3 error = 3
?at org.apache.hadoop.hbase.master.SplitLogManager.splitLogDistributed(SplitLogManager.java:290)
?at org.apache.hadoop.hbase.master.MasterFileSystem.splitLog(MasterFileSystem.java:402)
?at org.apache.hadoop.hbase.master.MasterFileSystem.splitLog(MasterFileSystem.java:375)
```

Therefore, you cannot set the quota value for the HBase directory in HDFS. If the exception occurs, perform the following operations:

- Step 1** Run the **kinit Username** command on the client to enable the HBase user to obtain security authentication.
- Step 2** Run the **hdfs dfs -count -q /hbase/data/<namespace>/<tableName>** command to check the allocated disk space quota.
- Step 3** Run the following command to cancel the quota limit and restore HBase:


```
hdfs dfsadmin -clrSpaceQuota /hbase/data/<namespace>/<tableName>
```

----End


```
?at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkLease(FSNamesystem.java:3432)
?at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.analyzeFileState(FSNamesystem.java:3223)
?at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getNewBlockTargets(FSNamesystem.java:3057)
?at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getAdditionalBlock(FSNamesystem.java:3011)
?at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.addBlock(NameNodeRpcServer.java:842)
?at
org.apache.hadoop.hdfs.protocolPB.ClientNamenodeProtocolServerSideTranslatorPB.addBlock(ClientNamenodeProtocolServerSideTranslatorPB.java:526)
?at org.apache.hadoop.hdfs.protocol.proto.ClientNamenodeProtocolProtos$ClientNamenodeProtocol$2.callBlockingMethod(ClientNamenodeProtocolProtos.java)
?at org.apache.hadoop.ipc.ProtobufRpcEngine$Server$ProtoBufRpcInvoker.call(ProtobufRpcEngine.java:616)
?at org.apache.hadoop.ipc.RPC$Server.call(RPC.java:973)
?at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2260)
?at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2256)
?at java.security.AccessController.doPrivileged(Native Method)
?at javax.security.auth.Subject.doAs(Subject.java:422)
?at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1769)
?at org.apache.hadoop.ipc.Server$Handler.run(Server.java:2254)

?at sun.reflect.GeneratedConstructorAccessor40.newInstance(Unknown Source)
?at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
?at java.lang.reflect.Constructor.newInstance(Constructor.java:423)
?at org.apache.hadoop.ipc.RemoteException.instantiateException(RemoteException.java:106)
?at org.apache.hadoop.ipc.RemoteException.unwrapRemoteException(RemoteException.java:73)
?at org.apache.hadoop.hdfs.DataStreamer.locateFollowingBlock(DataStreamer.java:1842)
?at org.apache.hadoop.hdfs.DataStreamer.nextBlockOutputStream(DataStreamer.java:1639)
?at org.apache.hadoop.hdfs.DataStreamer.run(DataStreamer.java:665)
```

Answer

During the WAL splitting process, the WAL splitting timeout period is specified by the **hbase.splitlog.manager.timeout** parameter. If the WAL splitting process fails to complete within the timeout period, the task is submitted again. Multiple WAL splitting tasks may be submitted during a specified period. If the **temp** file is deleted when one WAL splitting task completes, other tasks cannot find the file and the FileNotFound exception is reported. To avoid the problem, perform the following modifications:

The default value of **hbase.splitlog.manager.timeout** is 600,000 ms. The cluster specification is that each RegionServer has 2,000 to 3,000 regions. When the cluster is normal (HBase is normal and HDFS does not have a large number of read and write operations), you are advised to adjust this parameter based on the cluster specifications. If the actual specifications (the actual average number of regions on each RegionServer) are greater than the default specifications (the default average number of regions on each RegionServer, that is, 2,000), the adjustment solution is (actual specifications/default specifications) x Default time.

Set the **splitlog** parameter in the **hbase-site.xml** file on the server. [Table 8-27](#) describes the parameter.

Table 8-27 Description of the **splitlog** parameter

Parameter	Description	Default Value
hbase.splitlog.manager.timeout	Timeout period for receiving worker response by the distributed SplitLog management program.	600000

8.19.18 Insufficient Rights When a Tenant Accesses Phoenix

Question

When a tenant accesses Phoenix, a message is displayed indicating that the tenant has insufficient rights.

Answer

You need to associate the HBase service and Yarn queues when creating a tenant.

The tenant must be granted additional rights to perform operations on Phoenix, that is, the RWX permission on the Phoenix system table.

Example:

Tenant **hbase** has been created. Log in to the HBase Shell as user **admin** and run the **scan 'hbase:acl'** command to query the role of the tenant. The role is **hbase_1450761169920** (in the format of tenant name_timestamp).

Run the following commands to grant rights to the tenant (if the Phoenix system table has not been generated, log in to the Phoenix client as user **admin** first and then grant rights on the HBase Shell):

```
grant '@hbase_1450761169920','RWX','SYSTEM.CATALOG'
```

```
grant '@hbase_1450761169920','RWX','SYSTEM.FUNCTION'
```

```
grant '@hbase_1450761169920','RWX','SYSTEM.SEQUENCE'
```

```
grant '@hbase_1450761169920','RWX','SYSTEM.STATS'
```

Create user **phoenix** and bind it with tenant **hbase**, so that tenant **hbase** can access the Phoenix client as user **phoenix**.

8.19.19 What Can I Do When HBase Fails to Recover a Task and a Message Is Displayed Stating "Rollback recovery failed"?

Question

The system automatically rolls back data after an HBase recovery task fails. If "Rollback recovery failed" is displayed, the rollback fails. After the rollback fails, data stops being processed and the junk data may be generated. How can I resolve this problem?

Answer

You need to manually clear the junk data before performing the backup or recovery task next time.

Step 1 Install the cluster client in **/opt/client**.

Step 2 Run **source /opt/client/bigdata_env** as the client installation user to configure the environment variable.

Step 3 Run **kinit admin** to authenticate the user.

Step 4 Run **zkCli.sh -server business IP address of ZooKeeper:2181** to connect to the ZooKeeper.

Step 5 Run **deleteall /recovering** to delete the junk data. Run **quit** to disconnect ZooKeeper.

 **NOTE**

Running this command will cause data loss. Exercise caution.

Step 6 Run **hdfs dfs -rm -f -r /user/hbase/backup** to delete temporary data.

Step 7 On Manager, view related snapshot name information from recovery task records.
Snapshot [*snapshot name*] is created successfully before recovery.

Step 8 Switch to the client, run **hbase shell**, and then **delete_all_snapshot 'snapshot name.*'** to delete the temporary snapshot.

----End

8.19.20 How Do I Fix Region Overlapping?

Question

When the HBaseFsck tool is used to check the region status in MRS 3.x and later versions, if the log contains **ERROR: (regions region1 and region2) There is an overlap in the region chain** or **ERROR: (region region1) Multiple regions have the same startkey: xxx**, overlapping exists in some regions. How do I solve this problem?

Answer

To rectify the fault, perform the following steps:

Step 1 Run the **hbase hbck -repair tableName** command to restore the table that contains overlapping.

Step 2 Run the **hbase hbck tableName** command to check whether overlapping exists in the restored table.

- If overlapping does not exist, go to **Step 3**.
- If overlapping exists, go to **Step 1**.

Step 3 Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > HBase > More > Perform HMaster Switchover** to complete the HMaster active/standby switchover.

Step 4 Run the **hbase hbck tableName** command to check whether overlapping exists in the restored table.

- If overlapping does not exist, no further action is required.
- If overlapping still exists, start from **Step 1** to perform the recovery again.

----End

8.19.21 Why Does RegionServer Fail to Be Started When GC Parameters Xms and Xmx of HBase RegionServer Are Set to 31 GB?

Question

(MRS 3.x and later versions) Check the **hbase-omm-*.out** log of the node where RegionServer fails to be started. It is found that the log contains **An error report file with more information is saved as: /tmp/hs_err_pid*.log**. Check the **/tmp/hs_err_pid*.log** file. It is found that the log contains **#Internal Error (vtableStubs_aarch64.cpp:213), pid=9456, tid=0x0000ffff97fdd200 and #guarantee(__ pc() <= s->code_end()) failed: overflowed buffer**, indicating that the problem is caused by JDK. How do I solve this problem?

Answer

To rectify the fault, perform the following steps:

- Step 1** Run the **su - omm** command on a node where RegionServer fails to be started to switch to user **omm**.
- Step 2** Run the **java -XX:+PrintFlagsFinal -version |grep HeapBase** command as user **omm**. Information similar to the following is displayed:

```
uintx HeapBaseMinAddress = 2147483648 {pd product}
```
- Step 3** Change the values of **-Xms** and **-Xmx** in **GC_OPTS** to values that are not between **32G-HeapBaseMinAddress** and **32G**, excluding the values of **32G** and **32G-HeapBaseMinAddress**.
- Step 4** Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > HBase > Instance**, select the failed instance, and choose **More > Restart Instance** to restart the failed instance.

----End

8.19.22 Why Does the LoadIncrementalHFiles Tool Fail to Be Executed and "Permission denied" Is Displayed When Nodes in a Cluster Are Used to Import Data in Batches?

Question

Why does the LoadIncrementalHFiles tool fail to be executed and "Permission denied" is displayed when a Linux user is manually created in a normal cluster and DataNode in the cluster is used to import data in batches?

```
2020-09-20 14:53:53,808 WARN [main] shortcircuit.DomainSocketFactory: error creating DomainSocket
java.net.ConnectException: connect(2) error: Permission denied when trying to connect to '/var/run/
FusionInsight-HDFS/dn_socket'
    at org.apache.hadoop.net.unix.DomainSocket.connect0(Native Method)
    at org.apache.hadoop.net.unix.DomainSocket.connect(DomainSocket.java:256)
    at org.apache.hadoop.hdfs.shortcircuit.DomainSocketFactory.createSocket(DomainSocketFactory.java:168)
    at org.apache.hadoop.hdfs.client.impl.BlockReaderFactory.nextDomainPeer(BlockReaderFactory.java:804)
    at
    org.apache.hadoop.hdfs.client.impl.BlockReaderFactory.createShortCircuitReplicaInfo(BlockReaderFactory.java
```

```
:526)
  at org.apache.hadoop.hdfs.shortcircuit.ShortCircuitCache.create(ShortCircuitCache.java:785)
  at org.apache.hadoop.hdfs.shortcircuit.ShortCircuitCache.fetchOrCreate(ShortCircuitCache.java:722)
  at org.apache.hadoop.hdfs.client.impl.BlockReaderFactory.getBlockReaderLocal(BlockReaderFactory.java:
483)
  at org.apache.hadoop.hdfs.client.impl.BlockReaderFactory.build(BlockReaderFactory.java:360)
  at org.apache.hadoop.hdfs.DFSInputStream.getBlockReader(DFSInputStream.java:663)
  at org.apache.hadoop.hdfs.DFSInputStream.blockSeekTo(DFSInputStream.java:594)
  at org.apache.hadoop.hdfs.DFSInputStream.readWithStrategy(DFSInputStream.java:776)
  at org.apache.hadoop.hdfs.DFSInputStream.read(DFSInputStream.java:845)
  at java.io.DataInputStream.readFully(DataInputStream.java:195)
  at org.apache.hadoop.hbase.io.hfile.FixedFileTrailer.readFromStream(FixedFileTrailer.java:401)
  at org.apache.hadoop.hbase.io.hfile.HFile.isHFileFormat(HFile.java:651)
  at org.apache.hadoop.hbase.io.hfile.HFile.isHFileFormat(HFile.java:634)
  at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.visitBulkHFiles(LoadIncrementalHFiles.java:1090)
  at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.discoverLoadQueue(LoadIncrementalHFiles.java:
1006)
  at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.prepareHFileQueue(LoadIncrementalHFiles.java:
257)
  at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.doBulkLoad(LoadIncrementalHFiles.java:364)
  at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.run(LoadIncrementalHFiles.java:1263)
  at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.run(LoadIncrementalHFiles.java:1276)
  at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.run(LoadIncrementalHFiles.java:1311)
  at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:76)
  at org.apache.hadoop.hbase.tool.LoadIncrementalHFiles.main(LoadIncrementalHFiles.java:1333)
```

Answer

If the client that the LoadIncrementalHFiles tool depends on is installed in the cluster and is on the same node as DataNode, HDFS creates short-circuit read during the execution of the tool to improve performance. The short-circuit read depends on the **/var/run/FusionInsight-HDFS** directory (**dfs.domain.socket.path**). The default permission on this directory is **750**. This user does not have the permission to operate the directory.

To solve the preceding problem, perform the following operations:

Method 1: Create a user (recommended).

Step 1 Create a user on Manager. By default, the user group contains the **ficommon** group.

```
[root@xxx-xxx-xxx-xxx ~]# id test
uid=20038(test) gid=9998(ficommon) groups=9998(ficommon)
```

Step 2 Import data again.

----End

Method 2: Change the owner group of the current user.

Step 1 Add the user to the **ficommon** group.

```
[root@xxx-xxx-xxx-xxx ~]# usermod -a -G ficommon test
[root@xxx-xxx-xxx-xxx ~]# id test
uid=2102(test) gid=2102(test) groups=2102(test),9998(ficommon)
```

Step 2 Import data again.

----End

8.19.23 Why Is the Error Message "import argparse" Displayed When the Phoenix sqlline Script Is Used?

Question

When the sqlline script is used on the client, the error message "import argparse" is displayed.

Answer

- Step 1** Log in to the node where the HBase client is installed as user **root**. Perform security authentication using the **hbase** user.
- Step 2** Go to the directory where the sqlline script of the HBase client is stored and run the **python3 sqlline.py** command.

----End

8.19.24 How Do I Deal with the Restrictions of the Phoenix BulkLoad Tool?

Question

When the indexed field data is updated, if a batch of data exists in the user table, the BulkLoad tool cannot update the global and partial mutable indexes.

Answer

Problem Analysis

- Create a table.

```
CREATE TABLE TEST_TABLE(  
  DATE varchar not null,  
  NUM integer not null,  
  SEQ_NUM integer not null,  
  ACCOUNT1 varchar not null,  
  ACCOUNTDES varchar,  
  FLAG varchar,  
  SALL double,  
  CONSTRAINT PK PRIMARY KEY (DATE,NUM,SEQ_NUM,ACCOUNT1)  
);
```
- Create a global index.

```
CREATE INDEX TEST_TABLE_INDEX ON  
TEST_TABLE(ACCOUNT1,DATE,NUM,ACCOUNTDES,SEQ_NUM);
```
- Insert data.

```
UPSERT INTO TEST_TABLE  
(DATE,NUM,SEQ_NUM,ACCOUNT1,ACCOUNTDES,FLAG,SALL) values  
( '20201001',30201001,13,'367392332','sffa1','', );
```
- Execute the BulkLoad task to update data.

```
hbase org.apache.phoenix.mapreduce.CsvBulkLoadTool -t TEST_TABLE -  
i /tmp/test.csv, where the content of test.csv is as follows:
```

20201 001	30201 001	13	367392332	sffa88 8	12312 43	23
--------------	--------------	----	-----------	-------------	-------------	----

- Symptom: The existing index data cannot be directly updated. As a result, two pieces of index data exist.

```

+-----+-----+-----+-----+-----+
|:ACCOUNT1 | :DATE | :NUM | 0:ACCOUNTDES | :SEQ_NUM |
+-----+-----+-----+-----+-----+
| 367392332 | 20201001 | 30201001 | sffa1 | 13 |
| 367392332 | 20201001 | 30201001 | sffa888 | 13 |
+-----+-----+-----+-----+-----+

```

Solution

- Step 1 Delete the old index table.

```
DROP INDEX TEST_TABLE_INDEX ON TEST_TABLE;
```

- Step 2 Create an index table in asynchronous mode.

```
CREATE INDEX TEST_TABLE_INDEX ON  
TEST_TABLE(ACCOUNT1,DATE,NUM,ACCOUNTDES,SEQ_NUM) ASYNC;
```

- Step 3 Recreate a index.

```
hbase org.apache.phoenix.mapreduce.index.IndexTool --data-table  
TEST_TABLE --index-table TEST_TABLE_INDEX --output-path /user/test_table
```

```
----End
```

8.19.25 Why a Message Is Displayed Indicating that the Permission is Insufficient When CTBase Connects to the Ranger Plug-ins?

Question

When CTBase accesses the HBase service with the Ranger plug-ins enabled and you are creating a cluster table, a message is displayed indicating that the permission is insufficient.

```

ERROR: Create ClusterTable failed. Error: org.apache.hadoop.hbase.security.AccessDeniedException:
Insufficient permissions for user 'ctbase2@HADOOP.COM' (action=create)
at org.apache.ranger.authorization.hbase.AuthorizationSession.publishResults(AuthorizationSession.java:278)
at
org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor.authorizeAccess(RangerAuthorizati
nCoprocessor.java:654)
at
org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor.requirePermission(RangerAuthorizati
onCoprocessor.java:772)
at
org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor.preCreateTable(RangerAuthorizati
onCoprocessor.java:943)
at
org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor.preCreateTable(RangerAuthorizati
onCoprocessor.java:428)
at org.apache.hadoop.hbase.master.MasterCoprocessorHost$12.call(MasterCoprocessorHost.java:351)
at org.apache.hadoop.hbase.master.MasterCoprocessorHost$12.call(MasterCoprocessorHost.java:348)
at org.apache.hadoop.hbase.coprocessor.CoprocessorHost
$ObserverOperationWithoutResult.callObserver(CoprocessorHost.java:581)
at org.apache.hadoop.hbase.coprocessor.CoprocessorHost.execOperation(CoprocessorHost.java:655)

```

```
at org.apache.hadoop.hbase.master.MasterCoprocessorHost.preCreateTable(MasterCoprocessorHost.java:348)
at org.apache.hadoop.hbase.master.HMaster$5.run(HMaster.java:2192)
at
org.apache.hadoop.hbase.master.procedure.MasterProcedureUtil.submitProcedure(MasterProcedureUtil.java:134)
at org.apache.hadoop.hbase.master.HMaster.createTable(HMaster.java:2189)
at org.apache.hadoop.hbase.master.MasterRpcServices.createTable(MasterRpcServices.java:711)
at org.apache.hadoop.hbase.shaded.protobuf.generated.MasterProtos$MasterService$2.callBlockingMethod(MasterProtos.java)
at org.apache.hadoop.hbase.ipc.RpcServer.call(RpcServer.java:458)
at org.apache.hadoop.hbase.ipc.CallRunner.run(CallRunner.java:133)
at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:338)
at org.apache.hadoop.hbase.ipc.RpcExecutor$Handler.run(RpcExecutor.java:318)
```

Answer

CTBase users can configure permission policies on the Ranger page and grant the READ, WRITE, CREATE, ADMIN, and EXECUTE permissions to the CTBase metadata table `_ctmeta_`, cluster table, and index table.

9 Using HDFS

9.1 Using Hadoop from Scratch

You can use Hadoop to submit wordcount jobs. Wordcount is the most classic Hadoop job and is used to count the number of words in massive text.

Procedure

Step 1 Prepare the wordcount program.

Multiple open source Hadoop sample programs are provided, including wordcount. You can download the Hadoop sample program from <https://dist.apache.org/repos/dist/release/hadoop/common/>.

For example, select hadoop-2.10.x, download **hadoop-2.10.x.tar.gz**, decompress it, and obtain **hadoop-2.10.x\share\hadoop\mapreduce** (the Hadoop sample program) from **hadoop-mapreduce-examples-2.10.x.jar**. The **hadoop-mapreduce-examples-2.10.x.jar** sample program contains the wordcount program.

NOTE

hadoop-2.10.x indicates the Hadoop version.

Step 2 Prepare data files.

There is no format requirement for data files. Prepare one or more **.txt** files. The following are examples of the **.txt** file:

```
qwdsdfoedfrffrofhunckgktpmhutopmma  
jjpsffjfgorgjtyiuymhombmbogohoyhm  
jhheyombdhuacaqquyebchdhmamdhdemmj  
doeyhjwedcrfvgtgbmojijyhqssdddddfkf  
kjhjhkehdeiryudjfhfhffooqweopuyyyy
```

Step 3 Upload data to OBS.

1. Log in to OBS Console.
2. Click **Parallel File System** and choose **Create Parallel File System** to create a file system named **wordcount01**.

wordcount01 is only an example. The file system name must be globally unique. Otherwise, the parallel file system fails to be created.

3. In the OBS file system list, click **wordcount01** and choose **Files > Create Folder** to create the **program** and **input** folders.
 - **program**: stores user programs.
 - **input**: stores user data files.
4. Go to the **program** folder, choose **Upload File > add file**, select the program package downloaded in [Step 1](#) from the local host, and click **Upload**.
5. Go to the **input** folder and upload the data file prepared in [Step 2](#) to the **input** folder.

Step 4 Log in to the MRS console. In the navigation pane on the left, click **Clusters** and choose **Active Clusters**. Click the cluster name. The cluster must contain Hadoop components.

Step 5 Submit the wordcount job.

On the MRS console, click the **Jobs** tab and click **Create**. The **Create Job** page is displayed. For details, see [Running a MapReduce Job](#).

- Set **Type** to **MapReduce**.
- Set **Name** to **mr_01**.
- Set the path of the executable program to the address of the program stored on the OBS. For example: **obs://wordcount01/program/hadoop-mapreduce-examples-2.10.x.jar**
- Enter **wordcount obs://wordcount01/input/ obs://wordcount01/output/** in the **Parameter** pane.

 **NOTE**

- Replace the OBS file system name in **obs://wordcount01/input/** with the actual name of the file system created in the environment.
- Replace the OBS file system name in **obs://wordcount01/output/** with the actual name of the file system created in the environment. Enter a directory that does not exist in the **output** directory.
- **Service Parameter** can be left blank.

A job can be submitted only when the cluster is in the **Running** state.

After a job is submitted successfully, it is in the **Accepted** state by default. You do not need to manually execute the job.

Step 6 View the job execution result.

1. Go to the **Jobs** tab page and check whether the job is successfully executed.
It takes some time to run the job. After the job is complete, refresh the job list
Once a job has succeeded or failed, you cannot execute it again. However, you can add or copy a job, and set job parameters to submit a job again.
2. Log in to the OBS console, go to the OBS path, and view the job output information.
You can view output files in the **output** directory created in [Step 5](#). You need to download the file to the local host and open it in text format.

----End

9.2 Configuring Memory Management

Scenario

In HDFS, each file object needs to register corresponding information in the NameNode and occupies certain storage space. As the number of files increases, if the original memory space cannot store the corresponding information, you need to change the memory size.

Configuration Description

Navigation path for setting parameters:

Go to the **All Configurations** page of HDFS by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-1 Parameter description

Parameter	Description	Default Value
GC_PROFILE	<p>The NameNode memory size depends on the size of Fslmage, which can be calculated based on the following formula: Fslmage size = Number of files x 900 bytes. You can estimate the memory size of the NameNode of HDFS based on the calculation result.</p> <p>The value range of this parameter is as follows:</p> <ul style="list-style-type: none"> ● high: 4 GB ● medium: 2 GB ● low: 256 MB ● custom: The memory size can be set according to the data size in GC_OPTS. 	custom

Parameter	Description	Default Value
GC_OPTS	<p>JVM parameter used for garbage collection (GC). This parameter is valid only when GC_PROFILE is set to custom. Ensure that the GC_OPT parameter is set correctly. Otherwise, the process will fail to be started.</p> <p>NOTICE Exercise caution when you modify the configuration. If the configuration is incorrect, the services are unavailable.</p>	<p>-Xms2G -Xmx4G - XX:NewSize=128M - XX:MaxNewSize=256M - XX:MetaspaceSize=128M - XX:MaxMetaspaceSize=128M - XX:+UseConcMarkSweepGC -XX: +CMSParallelRemarkEnabled - XX:CMSInitiatingOccupancyFrac- tion=65 -XX:+PrintGCDetails - Dsun.rmi.dgc.client.gcInterval=0 x7FFFFFFFFFFFFFFE - Dsun.rmi.dgc.server.gcInterval=0 x7FFFFFFFFFFFFFFE -XX:- OmitStackTraceInFastThrow -XX: +PrintGCDateStamps -XX: +UseGCLogFileRotation - XX:NumberOfGCLogFiles=10 - XX:GCLogFileSize=1M - Djdk.tls.ephemeralDHKeySize=2 048</p>

9.3 Creating an HDFS Role

Scenario

This section describes how to create and configure an HDFS role on FusionInsight Manager. The HDFS role is granted the rights to read, write, and execute HDFS directories or files.

A user has the complete permission on the created HDFS directories or files, that is, the user can directly read data from and write data to as well as authorize others to access the HDFS directories or files.

NOTE

- This section applies to MRS 3.x or later.
- An HDFS role can be created only in security mode.
- If the current component uses Ranger for permission control, HDFS policies must be configured based on Ranger for permission management. For details, see [Adding a Ranger Access Permission Policy for HDFS](#).

Prerequisites

You have understood the service requirements.

Procedure

Step 1 Log in to FusionInsight Manager, and choose **System > Permission > Role**.

Step 2 On the displayed page, click **Create Role** and fill in **Role Name** and **Description**.

Step 3 Configure the resource permission. For details, see [Table 9-2](#).

File System: HDFS directory and file permission

Common HDFS directories are as follows:

- **flume:** Flume data storage directory
- **hbase:** HBase data storage directory
- **mr-history:** MapReduce task information storage directory
- **tmp:** temporary data storage directory
- **user:** user data storage directory

Table 9-2 Setting a role

Task	Operation
Setting the HDFS administrator permission	In the Configure Resource Permission area, choose <i>Name of the desired cluster</i> > HDFS, and select Cluster Admin Operations . NOTE The setting takes effect after the HDFS service is restarted.
Setting the permission for users to check and recover HDFS	<ol style="list-style-type: none"> 1. In the Configure Resource Permission area, choose <i>Name of the desired cluster</i> > HDFS > File System. 2. Locate the save path of specified directories or files on HDFS. 3. In the Permission column of the specified directories or files, select Read and Execute.
Setting the permission for users to read directories or files of other users	<ol style="list-style-type: none"> 1. In the Configure Resource Permission area, choose <i>Name of the desired cluster</i> > HDFS > File System. 2. Locate the save path of specified directories or files on HDFS. 3. In the Permission column of the specified directories or files, select Read and Execute.
Setting the permission for users to write data to files of other users	<ol style="list-style-type: none"> 1. In the Configure Resource Permission area, choose <i>Name of the desired cluster</i> > HDFS > File System. 2. Locate the save path of specified files on HDFS. 3. In the Permission column of the specified files, select Write and Execute.

Task	Operation
Setting the permission for users to create or delete sub-files or sub-directories in the directory of other users	<ol style="list-style-type: none"> 1. In the Configure Resource Permission area, choose <i>Name of the desired cluster</i> > HDFS > File System. 2. Locate the path where the specified directory is saved in the HDFS. 3. In the Permission column of the specified directories, select Write and Execute.
Setting the permission for users to execute directories or files of other users	<ol style="list-style-type: none"> 1. In the Configure Resource Permission area, choose <i>Name of the desired cluster</i> > HDFS > File System. 2. Locate the save path of specified directories or files on HDFS. 3. In the Permission column of the specified directories or files, select Execute.
Setting the permission for allowing subdirectories to inherit all permissions of their parent directories	<ol style="list-style-type: none"> 1. In the Configure Resource Permission area, choose <i>Name of the desired cluster</i> > HDFS > File System. 2. Locate the save path of specified directories or files on HDFS. 3. In the Permission column of the specified directories or files, select Recursive.

Step 4 Click **OK**, and return to the **Role** page.

----End

9.4 Using the HDFS Client

Scenario

This section describes how to use the HDFS client in an O&M scenario or service scenario.

Prerequisites

- The client has been installed.
For example, the installation directory is **/opt/hadoopclient**. The client directory in the following operations is only an example. Change it to the actual installation directory.
- Service component users are created by the MRS cluster administrator as required. In security mode, machine-machine users need to download the keytab file. A human-machine user needs to change the password upon the first login. (This operation is not required in normal mode.)

Using the HDFS Client

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 If the cluster is in security mode, run the following command to authenticate the user. In normal mode, user authentication is not required.

```
kinit Component service user
```

Step 5 Run the HDFS Shell command. Example:

```
hdfs dfs -ls /
```

```
----End
```

Common HDFS Client Commands

The following table lists common HDFS client commands.

For more commands, see https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/CommandsManual.html#User_Commands.

Table 9-3 Common HDFS client commands

Command	Description	Example
hdfs dfs -mkdir <i>Folder name</i>	Used to create a folder.	hdfs dfs -mkdir /tmp/mydir
hdfs dfs -ls <i>Folder name</i>	Used to view a folder.	hdfs dfs -ls /tmp
hdfs dfs -put <i>Local file on the client node</i>	Used to upload a local file to a specified HDFS path.	hdfs dfs -put /opt/test.txt /tmp Upload the /opt/test.txt file on the client node to the /tmp directory of HDFS.
hdfs dfs -get <i>Specified file on HDFS Specified path on the client node</i>	Used to download the HDFS file to the specified local path.	hdfs dfs -get /tmp/test.txt /opt/ Download the /tmp/test.txt file on HDFS to the /opt path on the client node.
hdfs dfs -rm -r -f <i>Specified folder on HDFS</i>	Used to delete a folder.	hdfs dfs -rm -r -f /tmp/mydir

Client-related FAQs

1. What do I do when the HDFS client exits abnormally and error message "java.lang.OutOfMemoryError" is displayed after the HDFS client command is running?

This problem occurs because the memory required for running the HDFS client exceeds the preset upper limit (128 MB by default). You can change the memory upper limit of the client by modifying **CLIENT_GC_OPTS** in *<Client installation path>/HDFS/component_env*. For example, if you want to set the upper limit to 1 GB, run the following command:

```
CLIENT_GC_OPTS="-Xmx1G"
```

After the modification, run the following command to make the modification take effect:

```
source <Client installation path>/bigdata_env
```

2. How can i set the log level when the HDFS client is running?

By default, the logs generated during the running of the HDFS client are printed to the console. The default log level is INFO. To enable the DEBUG log level for fault locating, run the following command to export an environment variable:

```
export HADOOP_ROOT_LOGGER=DEBUG,console
```

Then run the HDFS Shell command to generate the DEBUG logs.

If you want to print INFO logs again, run the following command:

```
export HADOOP_ROOT_LOGGER=INFO,console
```

9.5 Running the DistCp Command

Scenario

DistCp is a tool used to perform large-amount data replication between clusters or in a cluster. It uses MapReduce tasks to implement distributed copy of a large amount of data.

Prerequisites

- The Yarn client or a client that contains Yarn has been installed. For example, the installation directory is **/opt/client**.
- Service users of each component are created by the MRS cluster administrator based on service requirements. In security mode, machine-machine users need to download the keytab file. A human-machine user must change the password upon the first login. (Not involved in normal mode)
- To copy data between clusters, you need to enable the inter-cluster data copy function on both clusters.

Procedure

Step 1 Log in to the node where the client is installed.

Step 2 Run the following command to go to the client installation directory:

```
cd /opt/client
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 If the cluster is in security mode, the user group to which the user executing the DistCp command belongs must be **supergroup** and the user run the following command to perform user authentication. In normal mode, user authentication is not required.

```
kinit Component service user
```

Step 5 Run the DistCp command. The following provides an example:

```
hadoop distcp hdfs://hacluster/source hdfs://hacluster/target
```

----End

Common Usage of DistCp

1. The following is an example of the commonest usage of DistCp:

```
hadoop distcp -numListstatusThreads 40 -update -delete -prbugpaxtq hdfs://cluster1/source hdfs://cluster2/target
```

NOTE

In the preceding command:

- **-numListstatusThreads** specifies the number of threads for creating the list of 40 copied files.
- **-update -delete** specifies that files at the source location and the target location are synchronized, and that files with excessive target locations are deleted. If you need to copy files incrementally, delete **-delete**.
- If **-prbugpaxtq** and **-update** are used, it indicates that the status information of the copied file is also updated.
- **hdfs://cluster1/source** indicates the source location, and **hdfs://cluster2/target** indicates the target location.

2. The following is an example of data copy between clusters:

```
hadoop distcp hdfs://cluster1/foo/bar hdfs://cluster2/bar/foo
```

NOTE

The network between cluster1 and cluster2 must be reachable, and the two clusters must use the same HDFS version or compatible HDFS versions.

3. The following are multiple examples of data copy in a source directory:

```
hadoop distcp hdfs://cluster1/foo/a \  
hdfs://cluster1/foo/b \  
hdfs://cluster2/bar/foo
```

The preceding command is used to copy the folders a and b of cluster1 to the **/bar/foo** directory of cluster2. The effect is equivalent to that of the following commands:

```
hadoop distcp -f hdfs://cluster1/srclist \  
hdfs://cluster2/bar/foo
```

The content of **srclist** is as follows. Before running the DistCp command, upload the **srclist** file to HDFS.

```
hdfs://cluster1/foo/a  
hdfs://cluster1/foo/b
```

4. **-update** indicates that a to-be-copied file does not exist in the target location, or the content of the copied file in the target location is updated; and **-overwrite** is used to overwrite existing files in the target location.

The following is an example of the difference between no option and any one of the two options (either **update** or **overwrite**) that is added:

Assume that the structure of a file at the source location is as follows:

```
hdfs://cluster1/source/first/1
hdfs://cluster1/source/first/2
hdfs://cluster1/source/second/10
hdfs://cluster1/source/second/20
```

Commands without options are as follows:

```
hadoop distcp hdfs://cluster1/source/first hdfs://cluster1/source/second hdfs://cluster2/target
```

By default, the preceding command creates the **first** and **second** folders at the target location. Therefore, the copy results are as follows:

```
hdfs://cluster2/target/first/1
hdfs://cluster2/target/first/2
hdfs://cluster2/target/second/10
hdfs://cluster2/target/second/20
```

The command with any one of the two options (for example, **update**) is as follows:

```
hadoop distcp -update hdfs://cluster1/source/first hdfs://cluster1/source/second hdfs://cluster2/target
```

The preceding command copies only the content at the source location to the target location. Therefore, the copy results are as follows:

```
hdfs://cluster2/target/1
hdfs://cluster2/target/2
hdfs://cluster2/target/10
hdfs://cluster2/target/20
```

NOTE

- If files with the same name exist in multiple source locations, the DistCp command fails.
 - If neither **update** nor **overwrite** is used and the file to be copied already exists in the target location, the file will be skipped.
 - When **update** is used, if the file to be copied already exists in the target location but the file content is different, the file content in the target location is updated.
 - When **overwrite** is used, if the file to be copied already exists in the target location, the file in the target location is still overwritten.
5. The following table describes other command options:

Table 9-4 Other command options

Option	Description
-p[rbugpcaxtq]	When -update is also used, the status information of a copied file is updated even if the content of the copied file is not updated. r : number of copies b : size of a block u : user to which the files belong g : user group to which the user belongs p : permission c : check and type a : access control t : timestamp q : quota information
-i	Failures ignored during copying
-log <logdir>	Path of the specified log
-v	Additional information in the specified log
-m <num_maps>	Maximum number of concurrent copy tasks that can be executed at the same time
-numListstatusThreads	Number of threads for constituting the list of copied files. This option increases the running speed of DistCp.
-overwrite	File at the target location that is to be overwritten
-update	A file at the target location is updated if the size and check of a file at the source location are different from those of the file at the target location.
-append	When -update is also used, the content of the file at the source location is added to the file at the target location.
-f <urilist_uri>	Content of the <urilist_uri> file is used as the file list to be copied.
-filters	A local file is specified whose content contains multiple regular expressions. If the file to be copied matches a regular expression, the file is not copied.
-async	The distcp command is run asynchronously.
-atomic {-tmp <tmp_dir>}	An atomic copy can be performed. You can add a temporary directory during copying.

Option	Description
-bandwidth	The transmission bandwidth of each copy task. Unit: MB/s.
-delete	The files that exist in the target location is deleted but do not exist in the source location. This option is usually used with -update , and indicates that files at the source location are synchronized with those at the target location and the redundant files at the target location are deleted.
-diff <oldSnapshot> <newSnapshot>	The differences between the old and new versions are copied to a file in the old version at the target location.
-skipcrccheck	Whether to skip the cyclic redundancy check (CRC) between the source file and the target file.
-strategy {dynamic uniformsize}	The policy for copying a task. The default policy is uniformsize , that is, each copy task copies the same number of bytes.
-preserveec	Whether to retain the erasure code (EC) policy.

FAQs of DistCp

1. When you run the DistCp command, if the content of some copied files is large, you are advised to change the timeout period of MapReduce that executes the copy task. It can be implemented by specifying the **mapreduce.task.timeout** in the DistCp command. For example, run the following command to change the timeout to 30 minutes:

```
hadoop distcp -Dmapreduce.task.timeout=1800000 hdfs://cluster1/source hdfs://cluster2/target
```

Or, you can also use **filters** to exclude the large files out of the copy process. The command example is as follows:

```
hadoop distcp -filters /opt/client/filterfile hdfs://cluster1/source hdfs://cluster2/target
```

In the preceding command, *filterfile* indicates a local file, which contains multiple expressions used to match the path of a file that is not copied. The following is an example:

```
.*excludeFile1.*
.*excludeFile2.*
```

2. If the DistCp command unexpectedly quits, the error message "java.lang.OutOfMemoryError" is displayed.

This is because the memory required for running the copy command exceeds the preset memory limit (default value: 128 MB). You can change the memory upper limit of the client by modifying **CLIENT_GC_OPTS** in *<Client installation path>/HDFS/component_env*. For example, if you want to set the memory upper limit to 1 GB, refer to the following configuration:

```
CLIENT_GC_OPTS="-Xmx1G"
```

After the modification, run the following command to make the modification take effect:

```
source {Client installation path}/bigdata_env
```

- When the dynamic policy is used to run the DistCp command, the command exits unexpectedly and the error message "Too many chunks created with splitRatio" is displayed.

The cause of this problem is that the value of **distcp.dynamic.max.chunks.tolerable** (default value: 20,000) is less than the value of **distcp.dynamic.split.ratio** (default value: 2) multiplied by the number of Maps. This problem occurs when the number of Maps exceeds 10,000. You can use the **-m** parameter to reduce the number of Maps to less than 10,000.

```
hadoop distcp -strategy dynamic -m 9500 hdfs://cluster1/source hdfs://cluster2/target
```

Alternatively, you can use the **-D** parameter to set **distcp.dynamic.max.chunks.tolerable** to a large value.

```
hadoop distcp -strategy dynamic -Ddistcp.dynamic.max.chunks.tolerable=30000 hdfs://cluster1/source hdfs://cluster2/target
```

9.6 Overview of HDFS File System Directories

This section describes the directory structure in HDFS, as shown in the following table.

Table 9-5 HDFS directory structure (applicable to versions earlier than MRS 3.x)

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/tmp/spark/sparkhive-scratch	Fixed directory	Stores temporary files of metastore sessions in Spark JDBCServer.	No	Failed to run the task.
/tmp/sparkhive-scratch	Fixed directory	Stores temporary files of metastore session that are executed using Spark CLI.	No	Failed to run the task.
/tmp/carbon/	Fixed directory	Stores the abnormal data in this directory if abnormal CarbonData data exists during data import.	Yes	Error data is lost.
/tmp/Loader- $\{Job\ name\}_\{MR\ job\ ID\}$	Temporary directory	Stores the region information about Loader HBase bulkload jobs. The data is automatically deleted after the job running is completed.	No	Failed to run the Loader HBase Bulkload job.

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/tmp/logs	Fixed directory	Stores the collected MR task logs.	Yes	MR task logs are lost.
/tmp/archived	Fixed directory	Archives the MR task logs on HDFS.	Yes	MR task logs are lost.
/tmp/hadoop-yarn/staging	Fixed directory	Stores the run logs, summary information, and configuration attributes of ApplicationMaster running jobs.	No	Services are running improperly.
/tmp/hadoop-yarn/staging/history/done_intermediate	Fixed directory	Stores temporary files in the /tmp/hadoop-yarn/staging directory after all tasks are executed.	No	MR task logs are lost.
/tmp/hadoop-yarn/staging/history/done	Fixed directory	The periodic scanning thread periodically moves the done_intermediate log file to the done directory.	No	MR task logs are lost.
/tmp/mr-history	Fixed directory	Stores the historical record files that are pre-loaded.	No	Historical MR task log data is lost.
/tmp/hive	Fixed directory	Stores Hive temporary files.	No	Failed to run the Hive task.
/tmp/hive-scratch	Fixed directory	Stores temporary data (such as session information) generated during Hive running.	No	Failed to run the current task.
/user/{user}/sparkStaging	Fixed directory	Stores temporary files of the SparkJDBCServer application.	No	Failed to start the executor.
/user/spark/jars	Fixed directory	Stores running dependency packages of the Spark executor.	No	Failed to start the executor.

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/user/loader	Fixed directory	Stores dirty data of Loader jobs and data of HBase jobs.	No	Failed to execute the HBase job. Or dirty data is lost.
/user/loader/etl_dirty_data_dir				
/user/loader/etl_hbase_putlist_tmp				
/user/loader/etl_hbase_tmp				
/user/mapred	Fixed directory	Stores Hadoop-related files.	No	Failed to start Yarn.
/user/hive	Fixed directory	Stores Hive-related data by default, including the depended Spark lib package and default table data storage path.	No	User data is lost.
/user/omm-bulkload	Temporary directory	Stores HBase batch import tools temporarily.	No	Failed to import HBase tasks in batches.
/user/hbase	Temporary directory	Stores HBase batch import tools temporarily.	No	Failed to import HBase tasks in batches.
/sparkJobHistory	Fixed directory	Stores Spark event log data.	No	The History Server service is unavailable, and the task fails to be executed.
/flume	Fixed directory	Stores data collected by Flume from HDFS.	No	Flume runs improperly.

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/mr-history/tmp	Fixed directory	Stores logs generated by MapReduce jobs.	Yes	Log information is lost.
/mr-history/done	Fixed directory	Stores logs managed by MR JobHistory Server.	Yes	Log information is lost.
/tenant	Created when a tenant is added.	Directory of a tenant in the HDFS. By default, the system automatically creates a folder in the /tenant directory based on the tenant name. For example, the default HDFS storage directory for ta1 is tenant/ta1 . When a tenant is created for the first time, the system creates the /tenant directory in the HDFS root directory. You can customize the storage path.	No	The tenant account is unavailable.
/apps{1~5}/	Fixed directory	Stores the Hive package used by WebHCat.	No	Failed to run the WebHCat tasks.
/hbase	Fixed directory	Stores HBase data.	No	HBase user data is lost.
/hbaseFileStream	Fixed directory	Stores HFS files.	No	The HFS file is lost and cannot be restored.
/ats/active	Fixed directory	HDFS path used to store the timeline data of running applications.	No	Failed to run the tez task after the directory deletion.
/ats/done	Fixed directory	HDFS path used to store the timeline data of completed applications.	No	Automatically created after the deletion.

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/flink	Fixed directory	Stores the checkpoint task data.	No	Failed to run tasks after the deletion.

Table 9-6 Directory structure of the HDFS file system (applicable to MRS 3.x or later)

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/tmp/spark2x/sparkhive-scratch	Fixed directory	Stores temporary files of metastore session in Spark2x JDBCServer.	No	Failed to run the task.
/tmp/sparkhive-scratch	Fixed directory	Stores temporary files of metastore sessions that are executed in CLI mode using Spark2x CLI.	No	Failed to run the task.
/tmp/logs/	Fixed directory	Stores container log files.	Yes	Container log files cannot be viewed.
/tmp/carbon/	Fixed directory	Stores the abnormal data in this directory if abnormal CarbonData data exists during data import.	Yes	Error data is lost.
/tmp/Loader- <i>\${Job name}_</i> <i>\${MR job ID}</i>	Temporary directory	Stores the region information about Loader HBase bulkload jobs. The data is automatically deleted after the job running is completed.	No	Failed to run the Loader HBase Bulkload job.

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/tmp/hadoop-omm/yarn/system/rmstore	Fixed directory	Stores the ResourceManager running information.	Yes	Status information is lost after ResourceManager is restarted.
/tmp/archived	Fixed directory	Archives the MR task logs on HDFS.	Yes	MR task logs are lost.
/tmp/hadoop-yarn/staging	Fixed directory	Stores the run logs, summary information, and configuration attributes of ApplicationMaster running jobs.	No	Services are running improperly.
/tmp/hadoop-yarn/staging/history/done_intermediate	Fixed directory	Stores temporary files in the /tmp/hadoop-yarn/staging directory after all tasks are executed.	No	MR task logs are lost.
/tmp/hadoop-yarn/staging/history/done	Fixed directory	The periodic scanning thread periodically moves the done_intermediate log file to the done directory.	No	MR task logs are lost.
/tmp/mr-history	Fixed directory	Stores the historical record files that are pre-loaded.	No	Historical MR task log data is lost.
/tmp/hive-scratch	Fixed directory	Stores temporary data (such as session information) generated during Hive running.	No	Failed to run the current task.
/user/{user}/.spark Staging	Fixed directory	Stores temporary files of the SparkJDBCServer application.	No	Failed to start the executor.
/user/spark2x/jars	Fixed directory	Stores running dependency packages of the Spark2x executor.	No	Failed to start the executor.

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/user/loader	Fixed directory	Stores dirty data of Loader jobs and data of HBase jobs.	No	Failed to execute the HBase job. Or dirty data is lost.
/user/loader/etl_dirty_data_dir				
/user/loader/etl_hbase_pu_tlist_tmp				
/user/loader/etl_hbase_tmp				
/user/oozie	Fixed directory	Stores dependent libraries required for Oozie running, which needs to be manually uploaded.	No	Failed to schedule Oozie.
/user/mapred/hadoop-mapreduce-3.1.1.tar.gz	Fixed files	Stores JAR files used by the distributed MR cache.	No	The MR distributed cache function is unavailable.
/user/hive	Fixed directory	Stores Hive-related data by default, including the depended Spark lib package and default table data storage path.	No	User data is lost.
/user/omm-bulkload	Temporary directory	Stores HBase batch import tools temporarily.	No	Failed to import HBase tasks in batches.
/user/hbase	Temporary directory	Stores HBase batch import tools temporarily.	No	Failed to import HBase tasks in batches.

Path	Type	Function	Whether the Directory Can Be Deleted	Deletion Consequence
/spark2xJobHistory2x	Fixed directory	Stores Spark2.x eventlog data.	No	The History Server service is unavailable, and the task fails to be executed.
/flume	Fixed directory	Stores data collected by Flume from HDFS.	No	Flume runs improperly.
/mr-history/tmp	Fixed directory	Stores logs generated by MapReduce jobs.	Yes	Log information is lost.
/mr-history/done	Fixed directory	Stores logs managed by MR JobHistory Server.	Yes	Log information is lost.
/tenant	Created when a tenant is added.	Directory of a tenant in the HDFS. By default, the system automatically creates a folder in the / tenant directory based on the tenant name. For example, the default HDFS storage directory for ta1 is tenant/ta1 . When a tenant is created for the first time, the system creates the / tenant directory in the HDFS root directory. You can customize the storage path.	No	The tenant account is unavailable.
/apps{1~5}/	Fixed directory	Stores the Hive package used by WebHCat.	No	Failed to run the WebHCat tasks.
/hbase	Fixed directory	Stores HBase data.	No	HBase user data is lost.
/hbaseFileStream	Fixed directory	Stores HFS files.	No	The HFS file is lost and cannot be restored.

9.7 Changing the DataNode Storage Directory

Scenario

 NOTE

This section applies to MRS 3.x or later.

If the storage directory defined by the HDFS DataNode is incorrect or the HDFS storage plan changes, you need to modify the DataNode storage directory on FusionInsight Manager to ensure that the HDFS works properly. Changing the ZooKeeper storage directory includes the following scenarios:

- Change the storage directory of the DataNode role. In this way, the storage directories of all DataNode instances are changed.
- Change the storage directory of a single DataNode instance. In this way, only the storage directory of this instance is changed, and the storage directories of other instances remain the same.

Impact on the System

- The HDFS service needs to be stopped and restarted during the process of changing the storage directory of the DataNode role, and the cluster cannot provide services before it is completely started.
- The DataNode instance needs to be stopped and restarted during the process of changing the storage directory of the instance, and the instance at this node cannot provide services before it is started.
- The directory for storing service parameter configurations must also be updated.

Prerequisites

- New disks have been prepared and installed on each data node, and the disks are formatted.
- New directories have been planned for storing data in the original directories.
- The HDFS client has been installed.
- The user **hdfs** is available.
- When changing the storage directory of a single DataNode instance, ensure that the number of active DataNode instances is greater than the value of **dfs.replication**.

Procedure

Check the environment.

Step 1 Log in to the server where the HDFS client is installed as user **root**, and run the following command to configure environment variables:

source *Installation directory of the HDFS client*/**bigdata_env**

Step 2 If the cluster is in security mode, run the following command to authenticate the user:

```
kinit hdfs
```

Step 3 Run the following command on the HDFS client to check whether all directories and files in the HDFS root directory are normal:

```
hdfs fsck /
```

Check the fsck command output.

- If the following information is displayed, no file is lost or damaged. Go to [Step 4](#).
The filesystem under path '/' is HEALTHY
- If other information is displayed, some files are lost or damaged. Go to [Step 5](#).

Step 4 Log in to FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services**, and check whether **Running Status** of HDFS is **Normal**.

- If yes, go to [Step 6](#).
- If no, the HDFS status is unhealthy. Go to [Step 5](#).

Step 5 Rectify the HDFS fault.. The task is complete.

Step 6 Determine whether to change the storage directory of the DataNode role or that of a single DataNode instance:

- To change the storage directory of the DataNode role, go to [Step 7](#).
- To change the storage directory of a single DataNode instance, go to [Step 12](#).

Changing the storage directory of the DataNode role

Step 7 Choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS** > **Stop Instance** to stop the HDFS service.

Step 8 Log in to each data node where the HDFS service is installed as user **root** and perform the following operations:

1. Create a target directory (**data1** and **data2** are original directories in the cluster).

For example, to create a target directory `${BIGDATA_DATA_HOME}/hadoop/data3/dn`, run the following commands:

```
mkdir ${BIGDATA_DATA_HOME}/hadoop/data3 and mkdir $  
{BIGDATA_DATA_HOME}/hadoop/data3/dn
```

2. Mount the target directory to the new disk. For example, mount `${BIGDATA_DATA_HOME}/hadoop/data3` to the new disk.

3. Modify permissions on the new directory.

For example, to create a target directory `${BIGDATA_DATA_HOME}/hadoop/data3/dn`, run the following commands:

```
chmod 700 ${BIGDATA_DATA_HOME}/hadoop/data3/dn -R and chown  
omm:wheel ${BIGDATA_DATA_HOME}/hadoop/data3/dn -R
```

4. Copy the data to the target directory.

For example, if the old directory is `${BIGDATA_DATA_HOME}/hadoop/data1/dn` and the target directory is `${BIGDATA_DATA_HOME}/hadoop/data3/dn`, run the following command:

```
cp -af ${BIGDATA_DATA_HOME}/hadoop/data1/dn/* $  
{BIGDATA_DATA_HOME}/hadoop/data3/dn
```

- Step 9** On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS** > **Configurations** > **All Configurations** to go to the HDFS service configuration page.

Change the value of `dfs.datanode.data.dir` from the default value `%{@auto.detect.datapart.dn}` to the new target directory, for example, `${BIGDATA_DATA_HOME}/hadoop/data3/dn`.

For example, the original data storage directories are `/srv/BigData/hadoop/data1`, `/srv/BigData/hadoop/data2`. To migrate data from the `/srv/BigData/hadoop/data1` directory to the newly created `/srv/BigData/hadoop/data3` directory, replace the whole parameter with `/srv/BigData/hadoop/data2`, `/srv/BigData/hadoop/data3`. Separate multiple storage directories with commas (,). In this example, changed directories are `/srv/BigData/hadoop/data2`, `/srv/BigData/hadoop/data3`.

- Step 10** Click **Save**. Choose **Cluster** > *Name of the desired cluster* > **Services**. On the page that is displayed, start the services that have been stopped.
- Step 11** After the HDFS is started, run the following command on the HDFS client to check whether all directories and files in the HDFS root directory are correctly copied:

```
hdfs fsck /
```

Check the fsck command output.

- If the following information is displayed, no file is lost or damaged, and data replication is successful. No further action is required.
The filesystem under path '/' is HEALTHY
- If other information is displayed, some files are lost or damaged. In this case, check whether [8.4](#) is correct and run the `hdfs fsck Name of the damaged file -delete` command.

Changing the storage directory of a single DataNode instance

- Step 12** Choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS** > **Instance**. Select the HDFS instance whose storage directory needs to be modified, and choose **More** > **Stop Instance**.
- Step 13** Log in to the DataNode node as user **root**, and perform the following operations:
1. Create a target directory.
For example, to create a target directory `${BIGDATA_DATA_HOME}/hadoop/data3/dn`, run the following commands:

```
mkdir ${BIGDATA_DATA_HOME}/hadoop/data3 and mkdir $  
{BIGDATA_DATA_HOME}/hadoop/data3/dn
```
 2. Mount the target directory to the new disk.
For example, mount `${BIGDATA_DATA_HOME}/hadoop/data3` to the new disk.

3. Modify permissions on the new directory.
For example, to create a target directory `${BIGDATA_DATA_HOME}/hadoop/data3/dn`, run the following commands:

```
chmod 700 ${BIGDATA_DATA_HOME}/hadoop/data3/dn -R and chown omm:wheel ${BIGDATA_DATA_HOME}/hadoop/data3/dn -R
```
4. Copy the data to the target directory.
For example, if the old directory is `${BIGDATA_DATA_HOME}/hadoop/data1/dn` and the target directory is `${BIGDATA_DATA_HOME}/hadoop/data3/dn`, run the following command:

```
cp -af ${BIGDATA_DATA_HOME}/hadoop/data1/dn/* ${BIGDATA_DATA_HOME}/hadoop/data3/dn
```

Step 14 On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Service** > **HDFS** > **Instance**. Click the specified DataNode instance and go to the **Configurations** page.

Change the value of `dfs.datanode.data.dir` from the default value `%{@auto.detect.datapart.dn}` to the new target directory, for example, `${BIGDATA_DATA_HOME}/hadoop/data3/dn`.

For example, the original data storage directories are `/srv/BigData/hadoop/data1,/srv/BigData/hadoop/data2`. To migrate data from the `/srv/BigData/hadoop/data1` directory to the newly created `/srv/BigData/hadoop/data3` directory, replace the whole parameter with `/srv/BigData/hadoop/data2,/srv/BigData/hadoop/data3`.

Step 15 Click **Save**, and then click **OK**.

Operation succeeded is displayed. click **Finish**.

Step 16 Choose **More** > **Restart Instance** to restart the DataNode instance.

----End

9.8 Configuring HDFS Directory Permission

Scenario

The permission for some HDFS directories is **777** or **750** by default, which brings potential security risks. You are advised to modify the permission for the HDFS directories after the HDFS is installed to increase user security.

Procedure

Log in to the HDFS client as the administrator and run the following command to modify the permission for the `/user` directory.

The permission is set to **1777**, that is, **1** is added to the original permission. This indicates that only the user who creates the directory can delete it.

```
hdfs dfs -chmod 1777 /user
```

To ensure security of the system file, you are advised to harden the security for non-temporary directories. The following directories are examples:

- /user:777
- /mr-history:777
- /mr-history/tmp:777
- /mr-history/done:777
- /user/mapred:755

9.9 Configuring NFS

Scenario

NOTE

This section applies to MRS 3.x or later.

Before deploying a cluster, you can deploy a Network File System (NFS) server based on requirements to store NameNode metadata to enhance data reliability.

If the NFS server has been deployed and NFS services are configured, you can follow operations in this section to configure NFS on the cluster. These operations are optional.

Procedure

- Step 1** Check the permission of the shared NFS directories on the NFS server to ensure that the server can access NameNode in the MRS cluster.
- Step 2** Log in to the active NameNode as user **root**.
- Step 3** Run the following commands to create a directory and assign it write permissions:

```
mkdir ${BIGDATA_DATA_HOME}/namenode-nfs
chown omm:wheel ${BIGDATA_DATA_HOME}/namenode-nfs
chmod 750 ${BIGDATA_DATA_HOME}/namenode-nfs
```

- Step 4** Run the following command to mount the NFS to the active NameNode:

```
mount -t nfs -o rsize=8192,wsiz=8192,soft,nolock,timeo=3,intr IP address of the NFS server.Shared directory ${BIGDATA_DATA_HOME}/namenode-nfs
```

For example, if the IP address of the NFS server is **192.168.0.11** and the shared directory is **/opt/Hadoop/NameNode**, run the following command:

```
mount -t nfs -o rsize=8192,wsiz=8192,soft,nolock,timeo=3,intr
192.168.0.11:/opt/Hadoop/NameNode ${BIGDATA_DATA_HOME}/namenode-
nfs
```

- Step 5** Perform **Step 2** to **Step 4** on the standby NameNode.

NOTE

The names of the shared directories (for example, **/opt/Hadoop/NameNode**) created on the NFS server by the active and standby NameNodes must be different.

- Step 6** Log in to FusionInsight Manager, and choose **Cluster > Name of the desired cluster > Service > HDFS > Configuration > All Configurations**.

- Step 7** In the search box, search for **dfs.namenode.name.dir**, add **\$ {BIGDATA_DATA_HOME}/namenode-nfs** to **Value**, and click **Save**. Separate paths with commas (,).
- Step 8** Click **OK**. On the **Dashboard** tab page, choose **More > Restart Service** to restart the service.
- End

9.10 Planning HDFS Capacity

In HDFS, DataNode stores user files and directories as blocks, and file objects are generated on the NameNode to map each file, directory, and block on the DataNode.

The file objects on the NameNode require certain memory capacity. The memory consumption linearly increases as more file objects generated. The number of file objects on the NameNode increases and the objects consume more memory when the files and directories stored on the DataNode increase. In this case, the existing hardware may not meet the service requirement and the cluster is difficult to be scaled out.

Capacity planning of the HDFS that stores a large number of files is to plan the capacity specifications of the NameNode and DataNode and to set parameters according to the capacity plans.

Capacity Specifications

- NameNode capacity specifications

Each file object on the NameNode corresponds to a file, directory, or block on the DataNode.

A file uses at least one block. The default size of a block is **134,217,728**, that is, 128 MB, which can be set in the **dfs.blocksize** parameter. By default, a file whose size is less than 128 MB occupies only one block. If the file size is greater than 128 MB, the number of occupied blocks is the file size divided by 128 MB (Number of occupied blocks = File size/128). The directories do not occupy any blocks.

Based on **dfs.blocksize**, the number of file objects on the NameNode is calculated as follows:

Table 9-7 Number of NameNode file objects

Size of a File	Number of File Objects
< 128 MB	1 (File) + 1 (Block) = 2
> 128 MB (for example, 128 GB)	1 (File) + 1,024 (128 GB/128 MB = 1,024 blocks) = 1,025

The maximum number of file objects supported by the active and standby NameNodes is 300,000,000 (equivalent to 150,000,000 small files).

dfs.namenode.max.objects specifies the number of file objects that can be

generated in the system. The default value is **0**, which indicates that the number of generated file objects is not limited.

- DataNode capacity specifications

In HDFS, blocks are stored on the DataNode as copies. The default number of copies is **3**, which can be set in the **dfs.replication** parameter.

The number of blocks stored on all DataNode role instances in the cluster can be calculated based on the following formula: Number of HDFS blocks x 3
Average number of saved blocks = Number of HDFS blocks x 3/Number of DataNodes

Table 9-8 DataNode specifications

Item	Specifications
Maximum number of block copies supported by a DataNode instance	5,000,000
Maximum number of block copies supported by a disk on a DataNode instance	500,000
Minimum number of disks required when the number of block copies supported by a DataNode instance reaches the maximum	10

Table 9-9 Number of DataNodes

Number of HDFS Blocks	Minimum Number of DataNode Roles
10,000,000	$10,000,000 * 3 / 5,000,000 = 6$
50,000,000	$50,000,000 * 3 / 5,000,000 = 30$
100,000,000	$100,000,000 * 3 / 5,000,000 = 60$

Setting Memory Parameters

- Configuration rules of the NameNode JVM parameter

Default value of the NameNode JVM parameter **GC_OPTS**:

```
-Xms2G -Xmx4G -XX:NewSize=128M -XX:MaxNewSize=256M -
XX:MetaspaceSize=128M -XX:MaxMetaspaceSize=128M -XX:
+UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled -
XX:CMSInitiatingOccupancyFraction=65 -XX:+PrintGCDetails -
Dsun.rmi.dgc.client.gcInterval=0x7FFFFFFFFFFFFFFFE -
Dsun.rmi.dgc.server.gcInterval=0x7FFFFFFFFFFFFFFFE -XX:-
OmitStackTraceInFastThrow -XX:+PrintGCDateStamps -XX:
+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -XX:GCLogFileSize=1M -
Djdk.tls.ephemeralDHKeySize=2048
```

The number of NameNode files is proportional to the used memory size of the NameNode. When file objects change, you need to change **-Xms2G** -

Xmx4G -XX:NewSize=128M --XX:MaxNewSize=256M in the default value. The following table lists the reference values.

Table 9-10 NameNode JVM configuration

Number of File Objects	Reference Value
10,000,000	-Xms6G -Xmx6G -XX:NewSize=512M - XX:MaxNewSize=512M
20,000,000	-Xms12G -Xmx12G -XX:NewSize=1G - XX:MaxNewSize=1G
50,000,000	-Xms32G -Xmx32G -XX:NewSize=3G - XX:MaxNewSize=3G
100,000,000	-Xms64G -Xmx64G -XX:NewSize=6G - XX:MaxNewSize=6G
200,000,000	-Xms96G -Xmx96G -XX:NewSize=9G - XX:MaxNewSize=9G
300,000,000	-Xms164G -Xmx164G -XX:NewSize=12G - XX:MaxNewSize=12G

- Configuration rules of the DataNode JVM parameter

Default value of the DataNode JVM parameter **GC_OPTS**:

-Xms2G -Xmx4G -XX:NewSize=128M -XX:MaxNewSize=256M - XX:MetaspaceSize=128M -XX:MaxMetaspaceSize=128M -XX: +UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled - XX:CMSInitiatingOccupancyFraction=65 -XX:+PrintGCDetails - Dsun.rmi.dgc.client.gcInterval=0x7FFFFFFFFFFFFFFE - Dsun.rmi.dgc.server.gcInterval=0x7FFFFFFFFFFFFFFE -XX:- OmitStackTraceInFastThrow -XX:+PrintGCDateStamps -XX: +UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -XX:GCLogFileSize=1M - Djdk.tls.ephemeralDHKeySize=2048

The average number of blocks stored in each DataNode instance in the cluster is: Number of HDFS blocks x 3/Number of DataNodes. If the average number of blocks changes, you need to change **-Xms2G -Xmx4G - XX:NewSize=128M -XX:MaxNewSize=256M** in the default value. The following table lists the reference values.

Table 9-11 DataNode JVM configuration

Average Number of Blocks in a DataNode Instance	Reference Value
2,000,000	-Xms6G -Xmx6G -XX:NewSize=512M - XX:MaxNewSize=512M
5,000,000	-Xms12G -Xmx12G -XX:NewSize=1G - XX:MaxNewSize=1G

Xmx specifies memory which corresponds to the threshold of the number of DataNode blocks, and each GB memory supports a maximum of 500,000 DataNode blocks. Set the memory as required.

Viewing the HDFS Capacity Status

- NameNode information
For versions earlier than MRS 3.x: Log in to the MRS console, and choose **Components > HDFS > NameNode (Active)**. Click **Overview** and check the number of file objects, files, directories, or blocks in the HDFS in **Summary**.
For MRS 3.x or later: Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > HDFS > NameNode(Active)**, and click **Overview** to view information like the number of file objects, files, directories, and blocks in HDFS in **Summary** area.
- DataNode information
For versions earlier than MRS 3.x: Log in to the MRS console and choose **Components > HDFS > NameNode (Active)**. Click **DataNodes** and check the number of blocks of all DataNodes that report alarms.
For MRS 3.x or later: Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > HDFS > NameNode(Active)**, and click **DataNodes** to view the number of blocks on all DataNodes that report alarms.
- Alarm information
Check whether the alarms whose IDs are 14007, 14008, and 14009 are generated and change the alarm thresholds as required.

9.11 Configuring ulimit for HBase and HDFS

Symptom

When you open an HDFS file, an error occurs due to the limit on the number of file handles. Information similar to the following is displayed.

```
IOException (Too many open files)
```

Procedure

You can contact the MRS cluster administrator to add file handles for each user. This is a configuration on the OS instead of HBase or HDFS. It is recommended that configure the number of file handles based on the service traffic of HBase and HDFS and the rights of each user. If a user performs a large number of operations frequently on the HDFS that has large service traffic, set the number of file handles of this user to a large value.

Step 1 Log in to the OSs of all nodes or clients in the cluster as user **root**, and go to the **/etc/security** directory.

Step 2 Run the following command to edit the **limits.conf** file:

vi limits.conf

Add the following information to the file.

```
hdfs - nofile 32768
hbase - nofile 32768
```

hdfs and **hbase** indicate the usernames of the OSs that are used during the services.

NOTE

- Only user **root** has the rights to edit the **limits.conf** file.
- If this modification does not take effect, check whether other nofile values exist in the **/etc/security/limits.d** directory. Such values may overwrite the values set in the **/etc/security/limits.conf** file.
- If a user needs to perform operations on HBase, set the number of file handles of this user to a value greater than **10000**. If a user needs to perform operations on HDFS, set the number of file handles of this user based on the service traffic. It is recommended that the value not be too small. If a user needs to perform operations on both HBase and HDFS, set the number of file handles of this user to a large value, such as **32768**.

Step 3 Run the following command to check the limit on the number of file handles of a user:

```
su - user_name
```

```
ulimit -n
```

The limit on the number of file handles of this user is displayed as follows.

```
8194
```

```
----End
```

9.12 Balancing DataNode Capacity

Scenario

NOTE

This section applies to MRS 3.x or later.

In the HDFS cluster, unbalanced disk usage among DataNodes may occur, for example, when new DataNodes are added to the cluster. Unbalanced disk usage may result in multiple problems. For example, MapReduce applications cannot make full use of local computing advantages, network bandwidth usage between data nodes cannot be optimal, or node disks cannot be used. Therefore, you need to periodically check and maintain DataNode data balance.

HDFS provides a capacity balancing program Balancer. By running Balancer, you can balance the HDFS cluster and ensure that the difference between the disk usage of each DataNode and that of the HDFS cluster does not exceed the threshold. DataNode disk usage before and after balancing is shown in [Figure 9-1](#) and [Figure 9-2](#), respectively.

Figure 9-1 DataNode disk usage before balancing

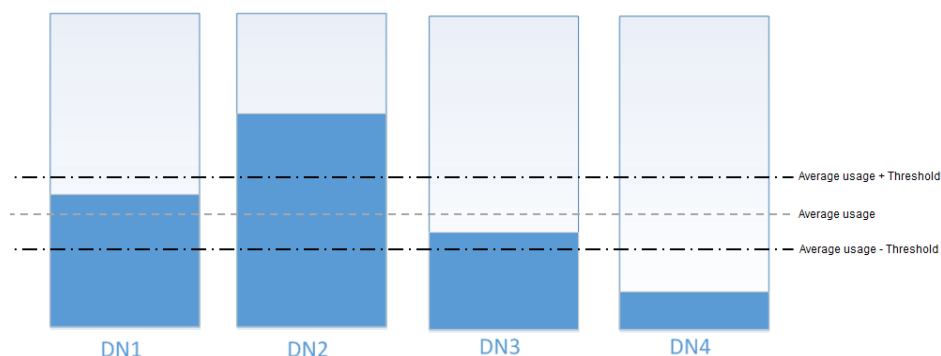
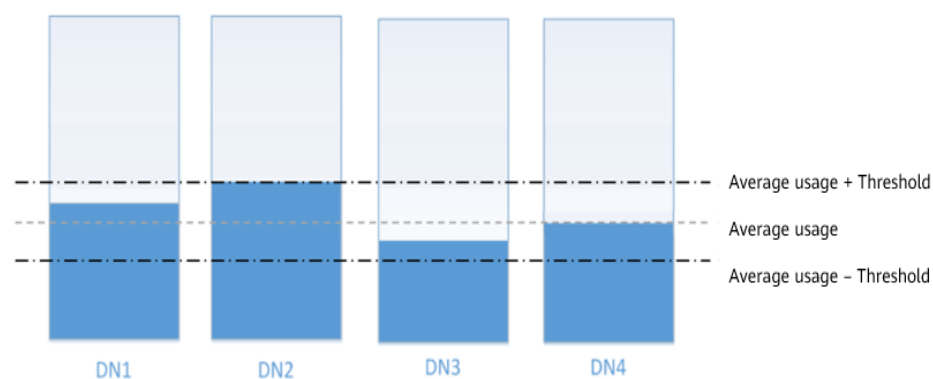


Figure 9-2 DataNode disk usage after balancing



The time of the balancing operation is affected by the following two factors:

1. Total amount of data to be migrated:
The data volume of each DataNode must be greater than $(\text{Average usage} - \text{Threshold}) \times \text{Average data volume}$ and less than $(\text{Average usage} + \text{Threshold}) \times \text{Average data volume}$. If the actual data volume is less than the minimum value or greater than the maximum value, imbalance occurs. The system sets the largest deviation volume on all DataNodes as the total data volume to be migrated.
2. Balancer migration is performed in sequence in iteration mode. The amount of data to be migrated in each iteration does not exceed 10 GB, and the usage of each iteration is recalculated.

Therefore, for a cluster, you can estimate the time consumed by each iteration (by observing the time consumed by each iteration recorded in balancer logs) and divide the total data volume by 10 GB to estimate the task execution time.

The balancer can be started or stopped at any time.

Impact on the System

- The balance operation occupies network bandwidth resources of DataNodes. Perform the operation during maintenance based on service requirements.
- The balance operation may affect the running services if the bandwidth traffic (the default bandwidth control is 20 MB/s) is reset or the data volume is increased.

Prerequisites

The client has been installed.

Procedure

- Step 1** Log in to the node where the client is installed as a client installation user. Run the following command to switch to the client installation directory, for example, `/opt/client`:

```
cd /opt/client
```

 **NOTE**

If the cluster is in normal mode, run the `su - omm` command to switch to user `omm`.

- Step 2** Run the following command to configure environment variables:

```
source bigdata_env
```

- Step 3** If the cluster is in security mode, run the following command to authenticate the HDFS identity:

```
kinit hdfs
```

- Step 4** Determine whether to adjust the bandwidth control.

- If yes, go to [Step 5](#).
- If no, go to [Step 6](#).

- Step 5** Run the following command to change the maximum bandwidth of Balancer, and then go to [Step 6](#).

```
hdfs dfsadmin -setBalancerBandwidth <bandwidth in bytes per second>
```

<bandwidth in bytes per second> indicates the bandwidth control value, in bytes. For example, to set the bandwidth control to 20 MB/s (the corresponding value is 20971520), run the following command:

```
hdfs dfsadmin -setBalancerBandwidth 20971520
```

 **NOTE**

- The default bandwidth control is 20 MB/s. This value is applicable to the scenario where the current cluster uses the 10GE network and services are being executed. If the service idle time window is insufficient for balance maintenance, you can increase the value of this parameter to shorten the balance time, for example, to 209715200 (200 MB/s).
- The value of this parameter depends on the networking. If the cluster load is high, you can change the value to 209715200 (200 MB/s). If the cluster is idle, you can change the value to 1073741824 (1 GB/s).
- If the bandwidth of the DataNodes cannot reach the specified maximum bandwidth, modify the HDFS parameter `dfs.datanode.balance.max.concurrent.moves` on FusionInsight Manager, and change the number of threads for balancing on each DataNode to **32** and restart the HDFS service.

- Step 6** Run the following command to start the balance task:

```
bash /opt/client/HDFS/hadoop/sbin/start-balancer.sh -threshold <threshold of balancer>
```

-threshold specifies the deviation value of the DataNode disk usage, which is used for determining whether the HDFS data is balanced. When the difference between the disk usage of each DataNode and the average disk usage of the entire HDFS cluster is less than this threshold, the system considers that the HDFS cluster has been balanced and ends the balance task.

For example, to set deviation rate to 5%, run the following command:

```
bash /opt/client/HDFS/hadoop/sbin/start-balancer.sh -threshold 5
```

NOTE

- The preceding command executes the task in the background. You can query related logs in the **hadoop-root-balancer-host name.out log** file in the **/opt/client/HDFS/hadoop/logs** directory of the host.
- To stop the balance task, run the following command:
bash /opt/client/HDFS/hadoop/sbin/stop-balancer.sh
- If only data on some nodes needs to be balanced, you can add the **-include** parameter in the script to specify the nodes to be migrated. You can run commands to view the usage of different parameters.
- **/opt/client** is the client installation directory. If the directory is inconsistent, replace it.
- If the command fails to be executed and the error information **Failed to APPEND_FILE /system/balancer.id** is displayed in the log, run the following command to forcibly delete **/system/balancer.id** and run the **start-balancer.sh** script again:
hdfs dfs -rm -f /system/balancer.id

Step 7 If the following information is displayed, the balancing is complete and the system automatically exits the task:

```
Apr 01, 2016 01:01:01 PM Balancing took 23.3333 minutes
```

After you run the script in **Step 6**, the **hadoop-root-balancer-Host name.out log** file is generated in the client installation directory **/opt/client/HDFS/hadoop/logs**. You can view the following information in the log:

- Time Stamp
- Bytes Already Moved
- Bytes Left To Move
- Bytes Being Moved

----End

Related Tasks

Enable automatic execution of the balance task

Step 1 Log in to FusionInsight Manager.

Step 2 Choose **Cluster > Name of the desired cluster > Services > HDFS > Configurations**, select **All Configurations**, search for the following parameters, and change the parameter values.

- **dfs.balancer.auto.enable** indicates whether to enable automatic balance task execution. The default value **false** indicates that automatic balance task execution is disabled. The value **true** indicates that automatic execution is enabled.

- **dfs.balancer.auto.cron.expression** indicates the task execution time. The default value **0 1 * * 6** indicates that the task is executed at 01:00 every Saturday. This parameter is valid only when the automatic execution is enabled.

Table 9-12 describes the expression for modifying this parameter. * indicates consecutive time segments.

Table 9-12 Parameters in the execution expression

Column	Description
1	Minute. The value ranges from 0 to 59.
2	Hour. The value ranges from 0 to 23.
3	Date. The value ranges from 1 to 31.
4	Month. The value ranges from 1 to 12.
5	Week. The value ranges from 0 to 6. 0 indicates Sunday.

- **dfs.balancer.auto.stop.cron.expression** indicates the task ending time. The default value is empty, indicating that the running balance task is not automatically stopped. For example, **0 5 * * 6** indicates that the balance task is stopped at 05:00 every Saturday. This parameter is valid only when the automatic execution is enabled.

Table 9-12 describes the expression for modifying this parameter. * indicates consecutive time segments.

Step 3 Running parameters of the balance task that is automatically executed are shown in **Table 9-13**.

Table 9-13 Running parameters of the automatic balancer

Parameter	Parameter description	Default Value
dfs.balancer.auto.threshold	Specifies the balancing threshold of the disk capacity percentage. This parameter is valid only when dfs.balancer.auto.enable is set to true .	10
dfs.balancer.auto.exclude.datanodes	Specifies the list of DataNodes on which automatic disk balancing is not required. This parameter is valid only when dfs.balancer.auto.enable is set to true .	The value is left blank by default.
dfs.balancer.auto.bandwidthPerSec	Specifies the maximum bandwidth (MB/s) of each DataNode for load balancing.	20

Parameter	Parameter description	Default Value
dfs.balancer.auto.maxIdleIterations	Specifies the maximum number of consecutive idle iterations of Balancer. An idle iteration is an iteration without moving blocks. When the number of consecutive idle iterations reaches the maximum number, the balance task ends. The value -1 indicates infinity.	5
dfs.balancer.auto.maxDataNodesNum	Controls the number of DataNodes that perform automatic balance tasks. Assume that the value of this parameter is N . If N is greater than 0, data is balanced between N DataNodes with the highest percentage of remaining space and N DataNodes with the lowest percentage of remaining space. If N is 0, data is balanced among all DataNodes in the cluster.	5

Step 4 Click **Save** to make configurations take effect. You do not need to restart the HDFS service.

Go to the `/var/log/Bigdata/hdfs/nn/hadoop-omm-balancer-Host name.log` file to view the task execution logs saved in the active NameNode.

----End

9.13 Configuring Replica Replacement Policy for Heterogeneous Capacity Among DataNodes

Scenario

By default, NameNode randomly selects a DataNode to write files. If the disk capacity of some DataNodes in a cluster is inconsistent (the total disk capacity of some nodes is large and of some nodes is small), the nodes with small disk capacity will be fully written. To resolve this problem, change the default disk selection policy for data written to DataNode to the available space block policy. This policy increases the probability of writing data blocks to the node with large available disk space. This ensures that the node usage is balanced when disk capacity of DataNodes is inconsistent.

Impact on the System

The disk selection policy is changed to `org.apache.hadoop.hdfs.server.blockmanagement.AvailableSpaceBlockPlacem`

entPolicy. It is proven that the HDFS file write performance optimizes by 3% after the modification.

 NOTE

The default replica storage policy of the NameNode is as follows:

1. First replica: stored on the node where the client resides.
2. Second replica: stored on DataNodes of the remote rack.
3. Third replica: stored on different nodes of the same rack for the node where the client resides.

If there are more replicas, randomly store them on other DataNodes.

The replica selection mechanism

(**org.apache.hadoop.hdfs.server.blockmanagement.AvailableSpaceBlockPlacementPolicy**) is as follows:

1. First replica: stored on the DataNode where the client resides (the same as the default storage policy).
2. Second replica:
 - When selecting a storage node, select two data nodes that meet the requirements.
 - Compare the disk usages of the two DataNodes. If the difference is smaller than 5%, store the replicas to the first node.
 - If the difference exceeds 5%, there is a 60% probability (specified by **dfs.namenode.available-space-block-placement-policy.balanced-space-preference-fraction** and default value is **0.6**) that the replica is written to the node whose disk space usage is low.
3. As for the storage of the third replica and subsequent replicas, refer to that of the second replica.

Prerequisites

The total disk capacity deviation of DataNodes in the cluster cannot exceed 100%.

Procedure

- Step 1** Go to the **All Configurations** page of HDFS by referring to [Modifying Cluster Service Configuration Parameters](#).
- Step 2** Modify the disk selection policy parameters when HDFS writes data. Search for the **dfs.block.replicator.classname** parameter and change its value to **org.apache.hadoop.hdfs.server.blockmanagement.AvailableSpaceBlockPlacementPolicy**.
- Step 3** Save the modified configuration. Restart the expired service or instance for the configuration to take effect.

----End

9.14 Configuring the Number of Files in a Single HDFS Directory

Scenario

Generally, multiple services are deployed in a cluster, and the storage of most services depends on the HDFS file system. Different components such as Spark

and Yarn or clients are constantly writing files to the same HDFS directory when the cluster is running. However, the number of files in a single directory in HDFS is limited. Users must plan to prevent excessive files in a single directory and task failure.

You can set the number of files in a single directory using the **dfs.namenode.fs-limits.max-directory-items** parameter in HDFS.

Procedure

- Step 1** Go to the **All Configurations** page of HDFS by referring to [Modifying Cluster Service Configuration Parameters](#).
- Step 2** Search for the configuration item **dfs.namenode.fs-limits.max-directory-items**.

Table 9-14 Parameter description

Parameter	Description	Default Value
dfs.namenode.fs-limits.max-directory-items	Maximum number of items in a directory Value range: 1 to 6,400,000	1048576

- Step 3** Set the maximum number of files that can be stored in a single HDFS directory. Save the modified configuration. Restart the expired service or instance for the configuration to take effect.

 **NOTE**

Plan data storage in advance based on time and service type categories to prevent excessive files in a single directory. You are advised to use the default value, which is about 1 million pieces of data in a single directory.

----End

9.15 Configuring the Recycle Bin Mechanism

Scenario

On HDFS, deleted files are moved to the recycle bin (trash can) so that the data deleted by mistake can be restored.

You can set the time threshold for storing files in the recycle bin. Once the file storage duration exceeds the threshold, it is permanently deleted from the recycle bin. If the recycle bin is cleared, all files in the recycle bin are permanently deleted.

Configuration Description

If a file is deleted from HDFS, the file is saved in the trash space rather than cleared immediately. After the aging time is due, the deleted file becomes an aging file and will be cleared based on the system mechanism or manually cleared by users.

Parameter portal:

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-15 Parameter description

Parameter	Description	Default Value
fs.trash.interval	Trash collection time, in minutes. If data in the trash station exceeds the time, the data will be deleted. Value range: 1440 to 259200	2880
fs.trash.checkpoint.interval	Interval between trash checkpoints, in minutes. The value must be less than or equal to the value of fs.trash.interval . The checkpoint program creates a checkpoint every time it runs and removes the checkpoint created fs.trash.interval minutes ago. For example, the system checks whether aging files exist every 10 minutes and deletes aging files if any. Files that are not aging are stored in the checkpoint list waiting for the next check. If this parameter is set to 0, the system does not check aging files and all aging files are saved in the system. Value range: 0 to <i>fs.trash.interval</i> NOTE It is not recommended to set this parameter to 0 because aging files will use up the disk space of the cluster.	60

9.16 Setting Permissions on Files and Directories

Scenario

HDFS allows users to modify the default permissions of files and directories. The default mask provided by the HDFS for creating file and directory permissions is **022**. If you have special requirements for the default permissions, you can set configuration items to change the default permissions.

Configuration Description

Parameter portal:

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-16 Parameter description

Parameter	Description	Default Value
fs.permissions.umask-mode	<p>This umask value (user mask) is used when the user creates files and directories in the HDFS on the clients. This parameter is similar to the file permission mask on Linux.</p> <p>The parameter value can be in octal or in symbolic, for example, 022 (octal, the same as u=rwx,g=r-x,o=r-x in symbolic), or u=rwx,g=rwx,o= (symbolic, the same as 007 in octal).</p> <p>NOTE The octal mask is opposite to the actual permission value. You are advised to use the symbol notation to make the description clearer.</p>	022

9.17 Setting the Maximum Lifetime and Renewal Interval of a Token

Scenario

In security mode, users can flexibly set the maximum token lifetime and token renewal interval in HDFS based on cluster requirements.

Configuration Description

Navigation path for setting parameters:

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-17 Parameter description

Parameter	Description	Default Value
dfs.namenode.delegation.token.max-lifetime	This parameter is a server parameter. It specifies the maximum lifetime of a token. Unit: milliseconds. Value range: 10,000 to 10,000,000,000,000	604,800,000
dfs.namenode.delegation.token.renew-interval	This parameter is a server parameter. It specifies the maximum lifetime to renew a token. Unit: milliseconds. Value range: 10,000 to 10,000,000,000,000	86,400,000

9.18 Configuring the Damaged Disk Volume

Scenario

In the open source version, if multiple data storage volumes are configured for a DataNode, the DataNode stops providing services by default if one of the volumes is damaged. You can change the value of **dfs.datanode.failed.volumes.tolerated** to specify the number of damaged disk volumes that are allowed. If the number of damaged volumes does not exceed the threshold, DataNode continues to provide services.

Configuration Description

Navigation path for setting parameters:

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-18 Parameter description

Parameter	Description	Default Value
dfs.datanode.failed.volumes.tolerated	Specifies the number of damaged volumes that are allowed before the DataNode stops providing services. By default, there must be at least one valid volume. The value -1 indicates that the minimum value of a valid volume is 1 . The value greater than or equal to 0 indicates the number of damaged volumes that are allowed.	Versions earlier than MRS 3.x: 0 MRS 3.x or later: -1

9.19 Configuring Encrypted Channels

Scenario

Encrypted channel is an encryption protocol of remote procedure call (RPC) in HDFS. When a user invokes RPC, the user's login name will be transmitted to RPC through RPC head. Then RPC uses Simple Authentication and Security Layer (SASL) to determine an authorization protocol (Kerberos and DIGEST-MD5) to complete RPC authorization. When users deploy security clusters, they need to use encrypted channels and configure the following parameters. For details about the secure Hadoop RPC, visit https://hadoop.apache.org/docs/r3.1.1/hadoop-project-dist/hadoop-common/SecureMode.html#Data_Encryption_on_RPC.

Configuration Description

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-19 Parameter description

Parameter	Description	Default Value
hadoop.rpc.protection	<p>NOTICE</p> <ul style="list-style-type: none"> The setting takes effect only after the service is restarted. Rolling restart is not supported. After the setting, you need to download the client configuration again. Otherwise, the HDFS cannot provide the read and write services. <p>Whether the RPC channels of each module in Hadoop are encrypted. The channels include:</p> <ul style="list-style-type: none"> RPC channels for clients to access HDFS RPC channels between modules in HDFS, for example, RPC channels between DataNode and NameNode RPC channels for clients to access Yarn RPC channels between NodeManager and ResourceManager RPC channels for Spark to access Yarn and HDFS RPC channels for MapReduce to access Yarn and HDFS RPC channels for HBase to access HDFS <p>NOTE</p> <p>You can set this parameter on the HDFS component configuration page. The parameter setting takes effect globally, that is, the setting of whether the RPC channel is encrypted takes effect on all modules in Hadoop.</p> <p>There are three encryption modes.</p> <ul style="list-style-type: none"> authentication: This is the default value in normal mode. In this mode, data is directly transmitted without encryption after being authenticated. This mode ensures performance but has security risks. integrity: Data is transmitted without encryption or authentication. To ensure data security, exercise caution when using this mode. privacy: This is the default value in security mode, indicating that data is transmitted after authentication and encryption. This mode reduces the performance. 	<ul style="list-style-type: none"> Security mode: privacy Normal mode: authentication

9.20 Reducing the Probability of Abnormal Client Application Operation When the Network Is Not Stable

Scenario

Clients probably encounter running errors when the network is not stable. Users can adjust the following parameter values to improve the running efficiency.

Configuration Description

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-20 Parameter description

Parameter	Description	Default Value
ha.health-monitor.rpc-timeout.ms	Timeout interval during the NameNode health check performed by ZKFC. Increasing this value can prevent dual active NameNodes and reduce the probability of application running exceptions on clients. Unit: millisecond. Value range: 30,000 to 3,600,000	180,000
ipc.client.connect.max.retries.on.timeouts	Number of retry times when the socket connection between a server and a client times out. Value range: 1 to 256	45
ipc.client.connect.timeout	Timeout interval of the socket connection between a client and a server. Increasing the value of this parameter increases the timeout interval for setting up a connection. Unit: millisecond. Value range: 1 to 3,600,000	20,000

9.21 Configuring the NameNode Blacklist

Scenario

 **NOTE**

This section applies to MRS 3.x or later.

In the existing default DFSclient failover proxy provider, if a NameNode in a process is faulty, all HDFS client instances in the same process attempt to connect

to the NameNode again. As a result, the application waits for a long time and timeout occurs.

When clients in the same JVM process connect to the NameNode that cannot be accessed, the system is overloaded. The NameNode blacklist is equipped with the MRS cluster to avoid this problem.

In the new Blacklisting DFSClient failover provider, the faulty NameNode is recorded in a list. The DFSClient then uses the information to prevent the client from connecting to such NameNodes again. This function is called NameNode blacklisting.

For example, there is a cluster with the following configurations:

```
namenode: nn1, nn2
```

```
dfs.client.failover.connection.retries: 20
```

Processes in a single JVM: 10 clients

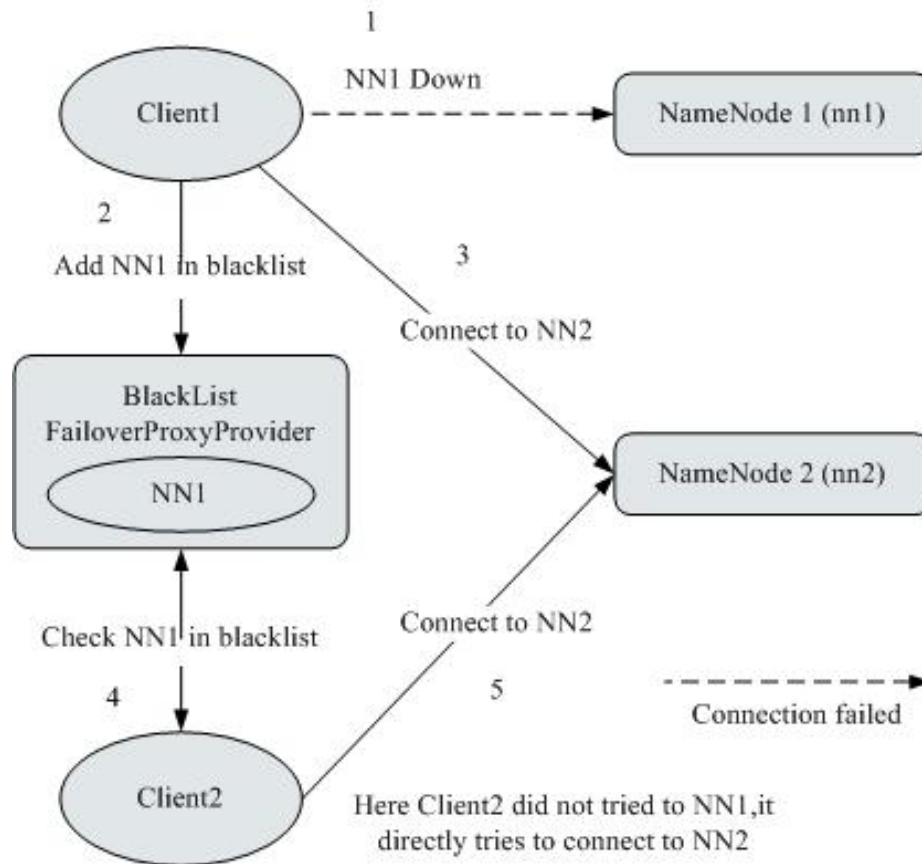
In the preceding cluster, if the active **nn1** cannot be accessed, client1 will retry the connection for 20 times. Then, a failover occurs, and client1 will connect to **nn2**. In the same way, other clients also connect to **nn2** when the failover occurs after retrying the connection to **nn1** for 20 times. Such process prolongs the fault recovery of NameNode.

In this case, the NameNode blacklisting adds **nn1** to the blacklist when client1 attempts to connect to the active **nn1** which is already faulty. Therefore, other clients will avoid trying to connect to **nn1** but choose **nn2** directly.

NOTE

If, at any time, all NameNodes are added to the blacklist, the content in the blacklist will be cleared, and the client attempts to connect to the NameNodes based on the initial NameNode list. If any fault occurs again, the NameNode is still added to the blacklist.

Figure 9-3 NameNode blacklisting working principle



Configuration Description

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-21 NameNode blacklisting parameters

Parameter	Description	Default Value
dfs.client.failover.proxy.provider. [nameservice ID]	Client Failover proxy provider class which creates the NameNode proxy using the authenticated protocol. Set this parameter to org.apache.hadoop.hdfs.server.namenode.ha.BlackListingFailoverProxyProvider . You can configure the observer NameNode to process read requests.	org.apache.hadoop.hdfs.server.namenode.ha.AdaptiveFailoverProxyProvider

9.22 Optimizing HDFS NameNode RPC QoS

Scenarios

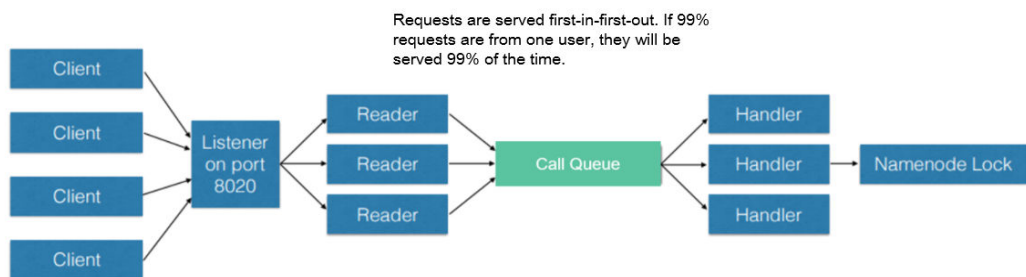
 NOTE

This section applies to MRS 3.x or later.

Several finished Hadoop clusters are faulty because the NameNode is overloaded and unresponsive.

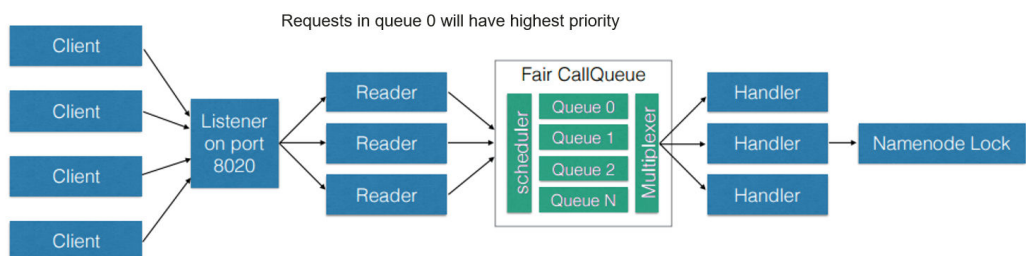
Such problem is caused by the initial design of Hadoop: In Hadoop, the NameNode functions as an independent part and in its namespace coordinates various HDFS operations, including obtaining the data block location, listing directories, and creating files. The NameNode receives HDFS operations, regards them as RPC calls, and places them in the FIFO call queue for read threads to process. Requests in FIFO call queue are served first-in first-out. However, users who perform more I/O operations are served more time than those performing fewer I/O operations. In this case, the FIFO is unfair and causes the delay.

Figure 9-4 NameNode request processing based on the FIFO call queue



The unfair problem and delaying mentioned before can be improved by replacing the FIFO queue with a new type of queue called FairCallQueue. In this way, FAIR queues assign incoming RPC calls to multiple queues based on the scale of the caller's call. The scheduling module tracks the latest calls and assigns a higher priority to users with a smaller number of calls.

Figure 9-5 NameNode request processing based on FAIRCallQueue



Configuration Description

- FairCallQueue ensures quality of service (QoS) by internally adjusting the order in which RPCs are invoked.

This queue consists of the following parts:

- a. DecayRpcScheduler: used to provide priority values from 0 to N (the value 0 indicates the highest priority).
- b. Multi-level queues (located in the FairCallQueue): used to ensure that queues are invoked in order of priority.
- c. Multi-channel converters (provided with Weighted Round Robin Multiplexer): used to provide logic control for queue selection.

After the FairCallQueue is configured, the control module determines the sub-queue to which the received invoking is allocated. The current scheduling module is DecayRpcScheduler, which only continuously tracks the priority numbers of various calls and periodically reduces these numbers.

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-22 FairCallQueue parameters

Parameter	Description	Default Value
<code>ipc.<port>.callqueue.impl</code>	Specifies the queue implementation class. You need to run the org.apache.hadoop.ipc.FairCallQueue command to enable the QoS feature.	<code>java.util.concurrent.LinkedBlockingQueue</code>

- RPC BackOff

Backoff is one of the FairCallQueue functions. It requires the client to retry operations (such as creating, deleting, and opening a file) after a period of time. When the backoff occurs, the RCP server throws RetriableException. The FairCallQueue performs backoff in either of the following cases:

- The queue is full, that is, there are many client calls in the queue.
- The queue response time is longer than the threshold time (specified by the **ipc.<port>.decay-scheduler.backoff.responsetime.thresholds** parameter).

Table 9-23 RPC Backoff configuration

Parameter	Description	Default Value
<code>ipc.<port>.backoff.enable</code>	Specifies whether to enable the backoff. When the current application contains a large number of user callings, the RPC request is blocked if the connection limit of the operating system is not reached. Alternatively, when the RPC or NameNode is heavily loaded, some explicit exceptions can be thrown back to the client based on certain policies. The client can understand these exceptions and perform exponential rollback, which is another implementation of the <code>RetryInvocationHandler</code> class.	false
<code>ipc.<port>.decay-scheduler.backoff.response-time.enable</code>	Indicate whether to enable the backoff based on the average queue response time.	false
<code>ipc.<port>.decay-scheduler.backoff.response-time.thresholds</code>	Configure the response time threshold for each queue. The response time threshold must match the number of priorities (the value of <code>ipc.<port>.faircallqueue.priority-levels</code>). Unit: millisecond	10000,20000,30000,40000

 **NOTE**

- `<port>` indicates the RPC port configured on the NameNode.
- The backoff function based on the response time takes effect only when `ipc.<port>.backoff.enable` is set to **true**.

9.23 Optimizing HDFS DataNode RPC QoS

Scenario

When the speed at which the client writes data to the HDFS is greater than the disk bandwidth of the DataNode, the disk bandwidth is fully occupied. As a result, the DataNode does not respond. The client can back off only by canceling or restoring the channel, which results in write failures and unnecessary channel recovery operations.

NOTE

This section applies to MRS 3.x or later.

Configuration

The new configuration parameter **dfs.pipeline.ecn** is introduced. When this configuration is enabled, the DataNode sends a signal from the write channel when the write channel is overloaded. The client may perform backoff based on the blocking signal to prevent the system from being overloaded. This configuration parameter is introduced to make the channel more stable and reduce unnecessary cancellation or recovery operations. After receiving the signal, the client backs off for a period of time (5,000 ms), and then adjusts the backoff time based on the related filter (the maximum backoff time is 50,000 ms).

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-24 DN ECN configuration

Parameter	Description	Default Value
dfs.pipeline.ecn	After configuration, the DataNode can send blocking notifications to the client.	false

9.24 Configuring Reserved Percentage of Disk Usage on DataNodes

Scenario

When the Yarn local directory and DataNode directory are on the same disk, the disk with larger capacity can run more tasks. Therefore, more intermediate data is stored in the Yarn local directory.

Currently, you can set **dfs.datanode.du.reserved** to configure the absolute value of the reserved disk space on DataNodes. A small value cannot meet the

requirements of a disk with large capacity. However, configuring a large value for a disk with same capacity wastes a lot of disk space.

To avoid this problem, a new parameter **dfs.datanode.du.reserved.percentage** is introduced to configure the reserved percentage of the disk space.

 **NOTE**

- If **dfs.datanode.du.reserved.percentage** and **dfs.datanode.du.reserved** are configured at the same time, the larger value of the reserved disk space calculated using the two parameters is used as the reserved space of the data nodes.
- You are advised to set **dfs.datanode.du.reserved** or **dfs.datanode.du.reserved.percentage** based on the actual disk space.

Configuration Description

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-25 Parameter description

Parameter	Description	Default Value
dfs.datanode.du.reserved.percentage	Indicates the percentage of the reserved disk space on DataNodes. The DataNode permanently reserves the disk space calculated using this percentage. The value is an integer ranging from 0 to 100.	10

9.25 Configuring HDFS NodeLabel

Scenario

You need to configure the nodes for storing HDFS file data blocks based on data features. You can configure a label expression to an HDFS directory or file and assign one or more labels to a DataNode so that file data blocks can be stored on specified DataNodes.

If the label-based data block placement policy is used for selecting DataNodes to store the specified files, the DataNode range is specified based on the label expression. Then proper nodes are selected from the specified range.

 **NOTE**

This section applies to MRS 3.x or later.

After cross-AZ HA is enabled for a single cluster, the HDFS NodeLabel function cannot be configured.

- Scenario 1: DataNodes partitioning scenario
Scenario description:

When different application data is required to run on different nodes for separate management, label expressions can be used to achieve separation of different services, storing specified services on corresponding nodes.

By configuring the NodeLabel feature, you can perform the following operations:

- Store data in **/Hbase** to DN1, DN2, DN3, and DN4.
- Store data in **/Spark** to DN5, DN6, DN7, and DN8.

Figure 9-6 DataNode partitioning scenario



NOTE

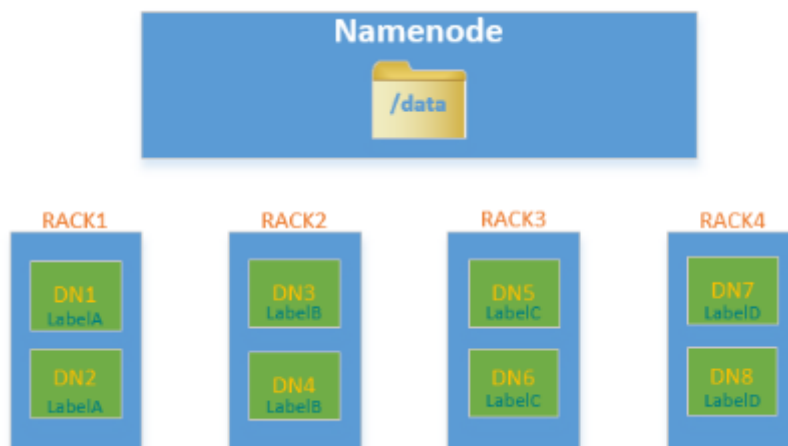
- Run the **hdfs nodelabel -setLabelExpression -expression 'LabelA[fallback=NONE]' -path /Hbase** command to set an expression for the **Hbase** directory. As shown in **Figure 9-6**, the data block replicas of files in the **/Hbase** directory are placed on the nodes labeled with the **LabelA**, that is, DN1, DN2, DN3, and DN4. Similarly, run the **hdfs nodelabel -setLabelExpression -expression 'LabelB[fallback=NONE]' -path /Spark** command to set an expression for the **Spark** directory. Data block replicas of files in the **/Spark** directory can be placed only on nodes labeled with **LabelB**, that is, DN5, DN6, DN7, and DN8.
 - For details about how to set labels for a data node, see [Configuration Description](#).
 - If multiple racks are available in one cluster, it is recommended that DataNodes of these racks should be available under each label, to ensure reliability of data block placement.
- Scenario 2: Specifying replica location when there are multiple racks

Scenario description:

In a heterogeneous cluster, customers need to allocate certain nodes with high availability to store important commercial data. Label expressions can be used to specify replica location so that the replica can be placed on a high reliable node.

Data blocks in the **/data** directory have three replicas by default. In this case, at least one replica is stored on a node of RACK1 or RACK2 (nodes of RACK1 and RACK2 are high reliable), and the other two are stored separately on the nodes of RACK3 and RACK4.

Figure 9-7 Scenario example



NOTE

Run the `hdfs nodelabel -setLabelExpression -expression 'LabelA|| LabelB[fallback=NONE],LabelC,LabelD' -path /data` command to set an expression for the `/data` directory.

When data is to be written to the `/data` directory, at least one data block replica is stored on a node labeled with the LabelA or LabelB, and the other two data block replicas are stored separately on the nodes labeled with the LabelC and LabelD.

Configuration Description

- DataNode label configuration

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-26 Parameter description

Parameter	Description	Default Value
dfs.block.replicator.classname	Used to configure the DataNode policy of HDFS. To enable the NodeLabel function, set this parameter to org.apache.hadoop.hdfs.server.blockmanagement.BlockPlacementPolicyWithNodeLabel .	org.apache.hadoop.hdfs.server.blockmanagement.AvailableSpaceBlockPlacementPolicy

Parameter	Description	Default Value
host2tags	Used to configure a mapping between a DataNode host and a label. The host name can be configured with an IP address extension expression (for example, 192.168.1.[1-128] or 192.168.[2-3].[1-128]) or a regular expression (for example, /datanode-[123]/ or /datanode-\d{2}/) starting and ending with a slash (/). The label configuration name cannot contain the following characters: = / \ Note: The IP address must be a service IP address.	-

 NOTE

- The **host2tags** configuration item is described as follows:
Assume there are 20 DataNodes which range from dn-1 to dn-20 in a cluster and the IP addresses of clusters range from 10.1.120.1 to 10.1.120.20. The value of **host2tags** can be represented in either of the following methods:
Regular expression of the host name
/dn-\d/ = label-1 indicates that the labels corresponding to dn-1 to dn-9 are label-1, that is, dn-1 = label-1, dn-2 = label-1, ..., dn-9 = label-1.
/dn-((1[0-9]\$)|(20\$))/ = label-2 indicates that the labels corresponding to dn-10 to dn-20 are label-2, that is, dn-10 = label-2, dn-11 = label-2, ...dn-20 = label-2.
IP address range expression
10.1.120.[1-9] = label-1 indicates that the labels corresponding to 10.1.120.1 to 10.1.120.9 are label-1, that is, 10.1.120.1 = label-1, 10.1.120.2 = label-1, ..., and 10.1.120.9 = label-1.
10.1.120.[10-20] = label-2 indicates that the labels corresponding to 10.1.120.10 to 10.1.120.20 are label-2, that is, 10.1.120.10 = label-2, 10.1.120.11 = label-2, ..., and 10.1.120.20 = label-2.
- Label-based data block placement policies are applicable to capacity expansion and reduction scenarios.
A newly added DataNode will be assigned a label if the IP address of the DataNode is within the IP address range in the **host2tags** configuration item or the host name of the DataNode matches the host name regular expression in the **host2tags** configuration item.
For example, the value of **host2tags** is **10.1.120.[1-9] = label-1**, but the current cluster has only three DataNodes: 10.1.120.1 to 10.1.120.3. If DataNode 10.1.120.4 is added for capacity expansion, the DataNode is labeled as label-1. If the 10.1.120.3 DataNode is deleted or out of the service, no data block will be allocated to the node.
- Set label expressions for directories or files.
 - On the HDFS parameter configuration page, configure **path2expression** to configure the mapping between HDFS directories and labels. If the configured HDFS directory does not exist, the configuration can succeed. When a directory with the same name as the HDFS directory is created manually, the configured label mapping relationship will be inherited by the directory within 30 minutes. After a labeled directory is deleted, a

- new directory with the same name as the deleted one will inherit its mapping within 30 minutes.
- For details about configuring items using commands, see the **hdfs nodelabel -setLabelExpression** command.
- To set label expressions using the Java API, invoke the **setLabelExpression(String src, String labelExpression)** method using the instantiated object `NodeLabelFileSystem`. *src* indicates a directory or file path on HDFS, and **labelExpression** indicates the label expression.
- After the NodeLabel is enabled, you can run the **hdfs nodelabel -listNodeLabels** command to view the label information of each DataNode.

Block Replica Location Selection

Nodelabel supports different placement policies for replicas. The expression **label-1,label-2,label-3** indicates that three replicas are respectively placed in DataNodes containing label-1, label-2, and label-3. Different replica policies are separated by commas (,).

If you want to place two replicas in DataNode with label-1, set the expression as follows: **label-1 [replica=2],label-2,label-3**. In this case, if the default number of replicas is 3, two nodes with label-1 and one node with label-2 are selected. If the default number of replicas is 4, two nodes with label-1, one node with label-2, and one node with label-3 are selected. Note that the number of replicas is the same as that of each replica policy from left to right. However, the number of replicas sometimes exceeds the expressions. If the default number of replicas is 5, the extra replica is placed on the last node, that is, the node labeled with label-3.

When the ACLs function is enabled and the user does not have the permission to access the labels used in the expression, the DataNode with the label is not selected for the replica.

Deletion of Redundant Block Replicas

If the number of block replicas exceeds the value of **dfs.replication** (number of file replicas specified by the user), HDFS will delete redundant block replicas to ensure cluster resource usage.

The deletion rules are as follows:

- Preferentially delete replicas that do not meet any expression.
For example: The default number of file replicas is **3**.
The label expression of **/test** is **LA[replica=1],LB[replica=1],LC[replica=1]**.
The file replicas of **/test** are distributed on four nodes (D1 to D4), corresponding to labels (LA to LD).
D1:LA
D2:LB
D3:LC
D4:LD
Then, block replicas on node D4 will be deleted.
- If all replicas meet the expressions, delete the redundant replicas which are beyond the number specified by the expression.
For example: The default number of file replicas is **3**.

The label expression of **/test** is **LA[replica=1],LB[replica=1],LC[replica=1]**.

The file replicas of **/test** are distributed on the following four nodes, corresponding to the following labels.

```
D1:LA
D2:LA
D3:LB
D4:LC
```

Then, block replicas on node D1 or D2 will be deleted.

- If a file owner or group of a file owner cannot access a label, preferentially delete the replica from the DataNode mapped to the label.

Example of label-based block placement policy

Assume that there are six DataNodes, namely, dn-1, dn-2, dn-3, dn-4, dn-5, and dn-6 in a cluster and the corresponding IP address range is 10.1.120.[1-6]. Six directories must be configured with label expressions. The default number of block replicas is **3**.

- The following provides three expressions of the DataNode label in **host2labels** file. The three expressions have the same function.

- Regular expression of the host name

```
/dn-[1456]/ = label-1,label-2
/dn-[26]/ = label-1,label-3
/dn-[3456]/ = label-1,label-4
/dn-5/ = label-5
```

- IP address range expression

```
10.1.120.[1-6] = label-1
10.1.120.1 = label-2
10.1.120.2 = label-3
10.1.120.[3-6] = label-4
10.1.120.[4-6] = label-2
10.1.120.5 = label-5
10.1.120.6 = label-3
```

- Common host name expression

```
/dn-1/ = label-1, label-2
/dn-2/ = label-1, label-3
/dn-3/ = label-1, label-4
/dn-4/ = label-1, label-2, label-4
/dn-5/ = label-1, label-2, label-4, label-5
/dn-6/ = label-1, label-2, label-3, label-4
```

- The label expressions of the directories are set as follows:

```
/dir1 = label-1
/dir2 = label-1 && label-3
/dir3 = label-2 || label-4[replica=2]
/dir4 = (label-2 || label-3) && label-4
/dir5 = !label-1
/sdir2.txt = label-1 && label-3[replica=3,fallback=NONE]
/dir6 = label-4[replica=2],label-2
```

NOTE

For details about the label expression configuration, see the **hdfs nodelabel - setLabelExpression** command.

The file data block storage locations are as follows:

- Data blocks of files in the **/dir1** directory can be stored on any of the following nodes: dn-1, dn-2, dn-3, dn-4, dn-5, and dn-6.
- Data blocks of files in the **/dir2** directory can be stored on the dn-2 and dn-6 nodes. The default number of block replicas is **3**. The expression

matches only two DataNodes. The third replica will be stored on one of the remaining nodes in the cluster.

- Data blocks of files in the **/dir3** directory can be stored on any three of the following nodes: dn-1, dn-3, dn-4, dn-5, and dn-6.
- Data blocks of files in the **/dir4** directory can be stored on the dn-4, dn-5, and dn-6 nodes.
- Data blocks of files in the **/dir5** directory do not match any DataNode and will be stored on any three nodes in the cluster, which is the same as the default block selection policy.
- For the data blocks of the **/sdir2.txt** file, two replicas are stored on the dn-2 and dn-6 nodes. The left one is not stored in the node because **fallback=NONE** is enabled.
- Data blocks of the files in the **/dir6** directory are stored on the two nodes with label-4 selected from dn-3, dn-4, dn-5, and dn-6 and another node with label-2. If the specified number of file replicas in the **/dir6** directory is more than 3, the extra replicas will be stored on a node with label-2.

Restrictions

In configuration files, **key** and **value** are separated by equation signs (=), colons (:), and whitespace. Therefore, the host name of the **key** cannot contain these characters because these characters may be considered as separators.

9.26 Configuring HDFS Mover

Scenario

Mover is a new data migration tool whose working mode is similar to that of the HDFS Balancer. Mover can redistribute data in the cluster based on the configured data storage policy.

Use Mover to periodically check whether the specified HDFS file or directory in the HDFS file system meets the preset storage policy. If not, migrate data to make them meet the policy.

NOTE

This section applies to MRS 3.x or later.

Configuration Description

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-27 Parameter description

Parameter	Description	Default Value
dfs.mover.auto.enable	Specifies whether to enable the data replica migration function. This function supports multiple modes. The default value is false , indicating that this function is disabled.	false
dfs.mover.auto.cron.expression	Specifies the CRON expression for HDFS automatic data migration, and is used to control the start time of data migration. This parameter is valid only when dfs.mover.auto.enable is set to true . The default value is 0 * * * * , indicating that the task is executed on the hour. For details about CRON expression, see Table 9-28 .	0 * * * *
dfs.mover.auto.hdfsfiles_or_dirs	Specifies HDFS file and directory lists that implement automatic replica migration in specified clusters. Multiple values are separated by space. This parameter is valid only when dfs.mover.auto.enable is set to true .	-

Table 9-28 CRON expressions

Column	Description
1	Minute. The value ranges from 0 to 59.
2	Hour. The value ranges from 0 to 23.
3	Date. The value ranges from 1 to 31.
4	Month. The value ranges from 1 to 12.
5	Week. The value ranges from 0 to 6. 0 indicates Sunday.

Use Restrictions

Run the command on the HDFS client to enable the mover function. The command format is as follows:

hdfs mover -p *<Full path or directory path of an HDFS file >*

NOTE

Users running this command on the client must have the **supergroup** permission. You can use the system user **hdfs** of the HDFS service. Alternatively, you can create a user with the **supergroup** permission in the cluster and then run the command.

9.27 Using HDFS AZ Mover

Scenario

AZ Mover is a copy migration tool used to move copies to meet the new AZ policies set on the directory. It can be used to migrate copies from one AZ policy to another. AZ Mover instructs NameNode to move copies based on a new AZ policy. If the NameNode refuses to delete the old copies, the new policy may not be met. For example, the copies are marked as outdated.

Restrictions

- Changing the policy name to **LOCAL_AZ** is the same as that to **ONE_AZ** because the client location cannot be determined when the uploaded file is written.
- Mover cannot determine the AZ status. As a result, the copy may be moved to the abnormal AZ and depends on NameNode for further processing.
- Mover depends on whether the number of DataNodes in each AZ meets the minimum requirement. If the AZ Mover is executed in an AZ with a small number of DataNodes, the result may be different from the expected result.
- Mover only meets the AZ-level policies and does not guarantee to meet the basic block placement policy (BPP).
- Mover does not support the change of replication factors. If the number of copies in the new AZ is different from that in the old AZ, an exception occurs.
- If a policy similar to **EC:AZ1** is configured and then changed to another policy such as **EC:AZ2**, the Mover execution result will not meet the expectation. To write the EC file to a single AZ, you are advised to use the default policy, for example, **EC:ONE_AZ** or **EC:LOCAL_AZ**.

Procedure

Step 1 Run the following command to go to the client installation directory.

```
cd /opt/client
```

Step 2 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 3 If the cluster is in security mode, the user must have the read permission on the source directory or file and the write permission on the destination directory, and run the following command to authenticate the user: In normal mode, skip user authentication.

```
kinit Component service user
```

Step 4 Create a directory and set an AZ policy.

Run the following command to create a directory.

```
hdfs dfs -mkdir <path>
```


Run the following command to set the AZ policy (**azexpression** indicates the AZ policy):

```
hdfs dfsadmin -setAZExpression <path> <azexpression>
```

Run the following command to view the AZ policy:

```
hdfs dfsadmin -getAZExpression <path>
```

Step 5 Upload files to the directory.

```
hdfs dfs -put <localfile> <hdfs-path>
```

Step 6 Delete the old policy from the directory and set a new policy.

Run the following command to clear the old policy:

```
hdfs dfsadmin -clearAZExpression <path>
```

Run the following command to configure a new policy:

```
hdfs dfsadmin -setAZExpression <path> <azexpression>
```

Step 7 Run the **azmover** command to make the copy distribution meet the new AZ policy.

```
hdfs azmover -p /targetDirecotry
```

----End

9.28 Configuring HDFS DiskBalancer

Scenario

DiskBalancer is an online disk balancer that balances disk data on running DataNodes based on various indicators. It works in the similar way of the HDFS Balancer. The difference is that HDFS Balancer balances data between DataNodes, while HDFS DiskBalancer balances data among disks on a single DataNode.

Data among disks may be unevenly distributed if a large number of files have been deleted from a cluster running for a long time, or disk capacity expansion is performed on a node in the cluster. Uneven data distribution may deteriorate the concurrent read/write performance of the HDFS, or cause service failure due to inappropriate HDFS write policies. In this case, the data density among disks on a node needs to be balanced to prevent heterogeneous small disks from becoming the performance bottleneck of the node.

NOTE

This section applies to MRS 3.x or later.

Configuration Description

Go to the **All Configurations** page of HDFS and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Table 9-29 Parameter description

Parameter	Description	Default Value
dfs.disk.balancer.auto.enabled	Indicates whether to enable the HDFS DiskBalancer function. The default value is false , indicating that this function is disabled.	false
dfs.disk.balancer.auto.cron.expression	CRON expression of the HDFS disk balancing operation, which is used to control the start time of the balancing operation. This parameter is valid only when dfs.disk.balancer.auto.enabled is set to true . The default value is 0 1 * * 6 , indicating that tasks are executed at 01:00 every Saturday. For details about cron expression, see Table 9-30 . The default value indicates that the DiskBalancer check is executed at 01:00 every Saturday.	0 1 * * 6
dfs.disk.balancer.max.disk.throughputInMBperSec	Specifies the maximum disk bandwidth that can be used for disk data balancing. The unit is MB/s, and the default value is 10 . Set this parameter based on the actual disk conditions of the cluster.	10
dfs.disk.balancer.max.disk.errors	Specifies the maximum number of errors that are allowed in a specified movement process. If the value exceeds this threshold, the movement fails.	5
dfs.disk.balancer.block.tolerance.percent	Specifies the difference threshold between the data storage capacity and perfect status of each disk during data balancing among disks. For example, the ideal data storage capacity of each disk is 1 TB, and this parameter is set to 10 . When the data storage capacity of the target disk reaches 900 GB, the storage status of the disk is considered as perfect. Value range: 1 to 100.	10
dfs.disk.balancer.plan.threshold.percent	Specifies the data density difference that is allowed between two disks during disk data balancing. If the absolute value of the data density difference between any two disks exceeds the threshold, data balancing is required. Value range: 1 to 100.	10
dfs.disk.balancer.top.nodes.number	Specifies the top <i>N</i> nodes whose disk data needs to be balanced in the cluster.	5

To use this function, set **dfs.disk.balancer.auto.enabled** to **true** and configure a proper CRON expression. Set other parameters based on the cluster status.

Table 9-30 CRON expressions

Column	Description
1	Minute. The value ranges from 0 to 59.
2	Hour. The value ranges from 0 to 23.
3	Date. The value ranges from 1 to 31.
4	Month. The value ranges from 1 to 12.
5	Week. The value ranges from 0 to 6. 0 indicates Sunday.

Use Restrictions

1. Data can only be moved between disks of the same type. For example, data can only be moved between SSDs or between DISKS.
2. Enabling this function occupies disk I/O resources and network bandwidth resources of involved nodes. Enable this function in off-peak hours.
3. The DataNodes specified by the **dfs.disk.balancer.top.nodes.number** parameter is frequently calculated. Therefore, set the parameter to a small value.
4. Commands for using the DiskBalancer function on the HDFS client are as follows:

Table 9-31 DiskBalancer commands

Syntax	Description
<code>hdfs diskbalancer -report -top <N></code>	Set <i>N</i> to an integer greater than 0. This command can be used to query the top <i>N</i> nodes that require disk data balancing in the cluster.
<code>hdfs diskbalancer -plan <Hostname IP Address></code>	This command can be used to generate a JSON file based on the DataNode. The file contains information about the source disk, target disk, and blocks to be moved. In addition, this command can be used to specify other parameters such as the network bandwidth.
<code>hdfs diskbalancer -query <Hostname: \$dfs.datanode.ipc.port></code>	The default port number of the cluster is 9867. This command is used to query the running status of the DiskBalancer task on the current node.

Syntax	Description
<code>hdfs diskbalancer -execute <planfile></code>	In this command, planfile indicates the JSON file generated in the second command. Use the absolute path.
<code>hdfs diskbalancer -cancel <planfile></code>	This command is used to cancel the running planfile. Use the absolute path.

 NOTE

- Users running this command on the client must have the **supergroup** permission. You can use the system user **hdfs** of the HDFS service. Alternatively, you can create a user with the **supergroup** permission in the cluster and then run the command.
- Only formats and usage of commands are provided in [Table 9-31](#). For more parameters to be configured for each command, run the `hdfs diskbalancer -help <command>` command to view detailed information.
- When you troubleshoot performance problems during the cluster O&M, check whether the HDFS disk balancing occurs in the event information of the cluster. If yes, check whether DiskBalancer is enabled in the cluster.
- After the automatic DiskBalancer function is enabled, the ongoing task stops only after the current data balancing is complete. The task cannot be canceled during the balancing.
- You can manually specify certain nodes for data balancing on the client.

9.29 Configuring HDFS EC Storage

Scenario

To ensure data reliability, HDFS stores three replicas by default, that is, when data is written, size of the space occupied is three times the size of the data. As a result, a large amount of space is wasted. To address this issue, HDFS introduces erasure coding (EC), a mature technology that has been applied to RAID disk arrays.

EC divides a file into small data blocks (blocks of the size of 128 KB, 256 KB, and 1 MB), stores these data blocks on multiple DataNodes, and stores the parity blocks generated by the EC encoding and decoding algorithm on other DataNodes. If a DataNode that stores data blocks is unavailable, the lost data block can be reversely calculated based on the parity blocks and other data blocks stored on other DataNodes.

NOTE

Different EC algorithms, data block sizes, number of data blocks, and number of parity blocks can form different EC policies. The following uses the **RS-6-3-1024k** policy as an example to describe the definition of the EC policy:

1. The Reed Solomon (RS) encoding and decoding algorithm is used.
2. **6** DataNodes are used to store data blocks.
3. **3** DataNodes are used to store parity blocks.
4. A maximum of **3** blocks can be lost.
5. The size of each file block is **1024 kB** (equal to 1 MB).
6. If the size of stored file is 100 MB, the total amount of data written to the DataNode is $(1+3/6) \times 100 \text{ MB} = 150 \text{ MB}$.
 1. The total size of data blocks is the file size: 100 MB.
 2. The total size of parity blocks is $3/6 \times 100 \text{ MB} = 50 \text{ MB}$.

The differences between the storage using the three-copy policy and the storage using the EC policy are as follows:

Figure 9-8 Common storage policy

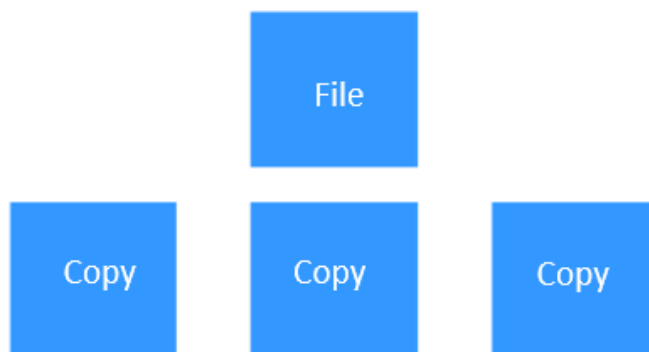
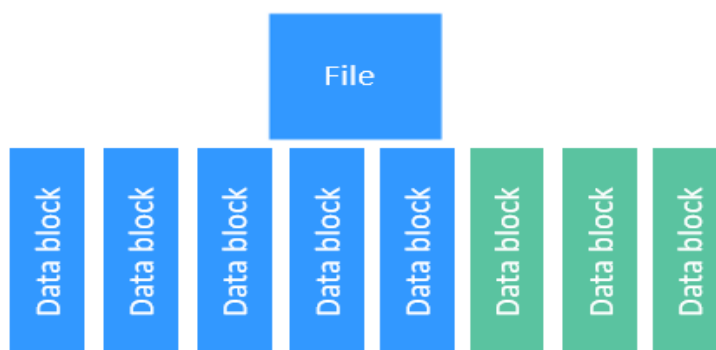


Figure 9-9 EC storage policy



By using the storage with the EC policy, a large amount of storage space can be saved without compromising the reliability. For example, to achieve the same reliability of three-copy policy, the storage space with only 1.5 times of the original file size is required. In this way, half of the storage space is saved.

 NOTE

- HDFS has the following built-in EC encoding and decoding algorithms: XOR, RS-LEGACY (the old version of Reed-Solomon, which has poor performance and is not recommended), and RS (the new version of Reed-Solomon with the performance five times that of RS-LEGACY).
- HDFS has the following built-in EC policies: RS-3-2-1024k, RS-6-3-1024k, RS-10-4-1024k, RS-LEGACY-6-3-1024k, and XOR-2-1-1024k.
- HDFS allows the user-defined EC policy by running commands on the client.
- An EC policy can be configured only for directories, not for files.
- After an EC policy is configured for a directory, all directories that are not configured with the EC policy in that directory will inherit the EC policy.

Problems and restrictions of applying the EC policy to HDFS:

- Files are distributed to multiple DataNodes during storing. Therefore, files cannot be read locally. As a result, data is de-affinity, causing serious data skew.
- When a file is written, calculation is needed to generate a parity block or verify the data block by a parity block, which consumes more CPU resources and increases complexity.
- When the DataNode is unavailable, lost data blocks need to be obtained by calculation, which consumes more CPU resources, increases the complexity, and lengthen the recovery time.
- When reading and writing files, the client needs to be connected with all related read/write data blocks and parity blocks at the same time.
- Currently, two basic FileSystem interfaces are not supported: **append()** and **truncate()**. When either of them is invoked, an exception is displayed. When **concat()** of the files with different EC policies is invoked, an exception is also displayed.

Based on the preceding restrictions and problems, EC feature is applicable to scenarios where a large amount of data is written at a time but not applicable to the scenarios with the following files:

- A large number of small files
- Files with small data volume
- Files that have been written for a long time after being opened
- Files that need to be modified by invoking the **append()** or **truncate()** interface

Impact on the System

Compared with the default replica storage, the EC storage has the following impacts on the system:

- NameNodes, DataNodes, and clients consume more CPU resources, memory, and network resources.
- If a DataNode is abnormal, it takes a longer time to restore data on the DataNode and reduces the read performance of related files.

To prevent these impacts, you are advised to perform the following operations for the cluster that will deploy EC storage based on the service load:

- Increase the values of memory parameters (mainly the value of **-Xmx**) in **GC_OPTS** of NameNode and DataNode.
- Increase the values of memory parameters of the HDFS client and upper-layer components that depend on HDFS.
- Increase the value of **dfs.datanode.max.transfer.threads** of the DataNode.
- If the number of DataNodes in the cluster is less than 20, you are advised to change the value of **dfs.namenode.redundancy.considerLoad** to **false**.

Prerequisites

- The HDFS client is installed and available. For details, see [Using the HDFS Client](#).
- If the cluster is in security mode, create a user that belongs to **supergroup** group and run the **kinit user** command to log in to Kerberos.
- If the cluster is in common mode, run the **export HADOOP_USER_NAME=omm** command.

Operation Command Description

- To list all EC codec algorithms supported by the current cluster, run the **hdfs ec -listCodecs** command. Example:

```
[root@10-219-254-200 ~]#hdfs ec -listCodecs
Erasure Coding Codecs: Codec [Coder List]
RS [RS_NATIVE, RS_JAVA]
RS-LEGACY [RS-LEGACY_JAVA]
XOR [XOR_NATIVE, XOR_JAVA]
```

- To list all EC policies in the current cluster, run the **hdfs ec -listPolicies** command. Example:

```
[root@10-219-254-200 ~]#hdfs ec -listPolicies
Erasure Coding Policies:
ErasureCodingPolicy=[Name=RS-10-4-1024k, Schema=[ECSchema=[Codec=rs, numDataUnits=10, numParityUnits=4]], CellSize=1048576, Id=5], State=DISABLED
ErasureCodingPolicy=[Name=RS-3-2-1024k, Schema=[ECSchema=[Codec=rs, numDataUnits=3, numParityUnits=2]], CellSize=1048576, Id=2], State=DISABLED
ErasureCodingPolicy=[Name=RS-6-3-1024k, Schema=[ECSchema=[Codec=rs, numDataUnits=6, numParityUnits=3]], CellSize=1048576, Id=1], State=ENABLED
ErasureCodingPolicy=[Name=RS-LEGACY-6-3-1024k, Schema=[ECSchema=[Codec=rs-legacy, numDataUnits=6, numParityUnits=3]], CellSize=1048576, Id=3], State=DISABLED
ErasureCodingPolicy=[Name=XOR-2-1-1024k, Schema=[ECSchema=[Codec=xor, numDataUnits=2, numParityUnits=1]], CellSize=1048576, Id=4], State=ENABLED
```

- To enable a specified EC policy, run the **hdfs ec -enablePolicy -policy <policyName>** command. Example:

```
[root@10-219-254-200 ~]#hdfs ec -enablePolicy -policy RS-3-2-1024k
Erasure coding policy RS-3-2-1024k is enabled
```

NOTICE

Only the EC policy in the **ENABLED** state can be set for a directory.

- To disable a specified EC policy, run the **hdfs ec -disablePolicy -policy <policyName>** command. Example:

```
[root@10-219-254-200 ~]#hdfs ec -disablePolicy -policy RS-3-2-1024k
Erasure coding policy RS-3-2-1024k is disabled
```

- To add a customized EC policy, run the **hdfs ec -addPolicies -policyFile <Configuration file path>** command.

 NOTE

1. First compile the configuration file of the customized EC policy. (For details, see *<HDFS client installation directory >/HDFS/hadoop/etc/hadoop/user_ec_policies.xml.template* sample.) Content of the sample is as follows:

```
<?xml version="1.0"?>
<configuration>
<!-- The version of EC policy XML file format, it must be an integer -->
<layoutversion>1</layoutversion>
<schemas>
<!-- schema id is only used to reference internally in this document -->
<schema id="XORk2m1">
<!-- The combination of codec, k, m and options as the schema ID, defines
a unique schema, for example 'xor-2-1'. schema ID is case insensitive -->
<!-- codec with this specific name should exist already in this system -->
<codec>xor</codec>
<k>2</k>
<m>1</m>
<options> </options>
</schema>
<schema id="RSk12m4">
<codec>rs</codec>
<k>12</k>
<m>4</m>
<options> </options>
</schema>
<schema id="RS-legacyk12m4">
<codec>rs-legacy</codec>
<k>12</k>
<m>4</m>
<options> </options>
</schema>
</schemas>
<policies>
<policy>
<!-- the combination of schema ID and cellsize(in unit k) defines a unique
policy, for example 'xor-2-1-256k', case insensitive -->
<!-- schema is referred by its id -->
<schema>XORk2m1</schema>
<!-- cellsize must be an positive integer multiple of 1024(1k) -->
<!-- maximum cellsize is defined by 'dfs.namenode.ec.policies.max.cellsize' property -->
<cellsize>131072</cellsize>
</policy>
<policy>
<schema>RS-legacyk12m4</schema>
<cellsize>262144</cellsize>
</policy>
</policies>
</configuration>
```

2. The customized EC policy is in the **DISABLED** state by default and can be used only after being enabled.

Example:

```
[root@10-219-254-200 ~]#hdfs ec -addPolicies -policyFile /opt/client/HDFS/hadoop/etc/hadoop/
user_ec_policies.xml.template
2018-12-31 17:50:17,666 INFO util.ECPolicyLoader: Loading EC policy file /opt/client/HDFS/
hadoop/etc/hadoop/user_ec_policies.xml.template
Add ErasureCodingPolicy XOR-2-1-128k succeed.
Add ErasureCodingPolicy RS-LEGACY-12-4-256k succeed.
```

- To remove a specified EC policy, run the **hdfs ec -removePolicy -policy <policy>** command. Example:

```
[root@10-219-254-200 ~]#hdfs ec -removePolicy -policy XOR-2-1-128k
Erasure coding policy XOR-2-1-128k is removed
```

- To set an EC policy of a directory, run the **hdfs ec -setPolicy -path <path> [-policy <policy>] [-replicate]** command. Example:


```
[root@10-219-254-200 ~]#hdfs ec -setPolicy -path /test -policy RS-6-3-1024k
Set RS-6-3-1024k erasure coding policy on /test
Warning: setting erasure coding policy on a non-empty directory will not automatically convert
existing files to RS-6-3-1024k erasure coding policy
```

 **NOTE**

- After an EC policy is set for a directory, the storage mode of original files in the directory remains unchanged. The newly created files are stored according to the EC policy.
- The **-replicate** parameter is used to set a three-copy storage policy instead of an EC policy for a directory.
- **-replicate** and **-policy <policyName>** cannot be used at the same time.
- To get the EC policy of a specified directory, run the **hdfs ec -getPolicy -path <path>** command. Example:

```
[root@10-219-254-200 ~]#hdfs ec -getPolicy -path /test
RS-6-3-1024k
```
- To disable the EC policy of a specified directory, run the **hdfs ec -unsetPolicy -path <path>** command. Example:

```
[root@10-219-254-200 ~]#hdfs ec -unsetPolicy -path /test
Unset erasure coding policy from /test
Warning: unsetting erasure coding policy on a non-empty directory will not automatically convert
existing files to replicated data.
```

 **NOTE**

When the EC policy of a directory is disabled, it inherits the EC policy of the parent directory if the parent directory has an EC policy.

Suggestions on Using EC for Upper-Level Services

According to the current situation of EC storage, the following upper-level services are recommended to use EC: Yarn, MapReduce, HBase, Hive, and Spark2x.

The HDFS directories of each service that can use the EC storage are listed in [Table 9-32](#):

Table 9-32 Directories of services that can use EC storage

Service	HDFS directory	Parameter	Description
Yarn	/tmp/logs	yarn.nodemanager.remote-app-log-dir	Directory for storing container log files.
	/tmp/archived	yarn.nodemanager.remote-app-log-archive-dir	Directory for storing archived task log files.
MapReduce	/mr-history/tmp	-	Directory for storing logs generated by MapReduce jobs.
	/mr-history/done	-	Directory for storing logs managed by MR JobHistory Server.
HBase	/hbase/.tmp	-	Temporary directory for creating tables.

Service	HDFS directory	Parameter	Description
	/hbase/archive	-	Archive directory.
	/hbase/data	-	Table data directory.
Hive	/user/hive/warehouse	hive.metastore.warehouse.dir	Hive data warehouse directory.
	/tmp/hive-scratch	hive.exec.scratchdir	Temporary data directory run by Hive jobs.
Spark2x	/user/hive/warehouse	hive.metastore.warehouse.dir	Default directory for storing table data.

The EC policy cannot be used for data directories listed in [Table 9-33](#). Otherwise, services fail to run. If the EC policy is configured for these directories, HDFS automatically cancels the EC policy and sets the replica policy for these directories.

Table 9-33 Directories of services that cannot use EC

Service	HDFS directory	Parameter	Description
Public	/tmp	-	HDFS temporary directory for storing temporary files.
	/user	-	HDFS user directory for storing user data.
HBase	/hbase	hbase.data.rootdir	HBase data root directory.
	/hbase/.hbase-snapshot	-	Table Snapshot directory.
	/hbase/.hbck	-	Directory for storing hfiles (such as out-of-bound hfiles) that fail the hbck check.
	/hbase/MasterProcWALs	-	Directory for storing Proc logs.
	/hbase/WALs	-	Directory for storing WALs.
	/hbase/corrupt	-	Directory for storing damaged HFiles.
	/hbase/oldWALs	-	Directory to which data in WALs is migrated.

Service	HDFS directory	Parameter	Description
Spark2x	/user/spark2x	spark.yarn.stagingDir	Directory for storing temporary files generated during Spark2x application running.
	/tmp/spark2x/sparkhive-scratch	hive.exec.scratchdir	Directory for storing temporary files generated during SQL execution on JDBCServer.
	/spark2xJobHistory2x	spark.eventLog.dir	Directory for storing event logs.
	/user/spark2x/jars	-	Directory for storing Spark2x JAR files.
CarbonData	/user/hive/warehouse/carbon.store/	carbon.storelocation	Directory for storing data files.
Flink	/flink	-	Directory for storing Flink running snapshot information for task restoration.

 **NOTE**

You can specify the list of HDFS directories for which the EC policy cannot be used by specifying **dfs.no.ec.dirs**. Use commons (,) to separate multiple values.

EC and Other Storage Policies

EC supports the following storage policies: **HOT**, **COLD**, and **ALL_SSD**.

If other storage policies (such as **WARM**, **ONE_SSD**, **PROVIDED**, and **LAZY_PERSIST**) are configured, the default storage policy (**HOT**) will be used.

 **NOTE**

Information about all blocks needs to be obtained for reading and writing EC files. To ensure the consistency of the read and write time of all blocks, the blocks need to be stored on the disks with the same performance. Only **HOT**, **COLD**, and **ALL_SSD** storage policies meet this condition.

Impact of EC on DistCp

If the source file of DistCp contains an EC file, different parameters need to be added based on different requirements.

- To preserve the EC policy of a file during the replication process, the **-preserveec** parameter needs to be added to the DistCp command.

- If the file is rewritten to a destination directory based on the EC policy (or replication policy) of the destination directory but not of the source file, the **-skipcrccheck** parameter needs to be added to the DistCp command.

 **CAUTION**

If neither of the two parameters is added, the DistCp command task fails due to different parity values of the files with different EC policies.

Conversion Between Copy Files and EC Files

Convert a replica file into an EC file. (For example, convert all files in the **/src** directory into the EC files with the **RS-6-3-1024k** policy.)

- Step 1** Run the following command to create a temporary directory (for example, **/tmp/convert_tmp_dir**, which must be a directory that does not exist) to temporarily store converted data.

```
hdfs dfs -mkdir /tmp/convert_tmp_dir
```

- Step 2** Run the following command to enable the EC policy **RS-6-3-1024k**. (If it has been enabled, skip this step.)

```
hdfs ec -enablePolicy -policy RS-6-3-1024k
```

- Step 3** Run the following command to set the EC policy of **/tmp/convert_tmp_dir** to **RS-6-3-1024k**.

```
hdfs ec -setPolicy -path /tmp/convert_tmp_dir -policy RS-6-3-1024k
```

- Step 4** Run the following DistCp command to copy files. In this case, all files are written into the temporary directory based on the **RS-6-3-1024k** policy.

```
hadoop distcp -numListstatusThreads 40 -update -delete -skipcrccheck -pbugpaxt /src /tmp/convert_tmp_dir
```

- Step 5** Run the following command to move **/src** to **/src-to-delete**.

```
hdfs dfs -mv /src /src-to-delete
```

- Step 6** Run the following command to move **/tmp/convert_tmp_dir** to **/src**.

```
hdfs dfs -mv /tmp/convert_tmp_dir /src
```

- Step 7** Run the following command to delete **/src-to-delete**.

```
hdfs dfs -rm -r -f /src-to-delete
```

 **NOTE**

Deleting a file is a high-risk operation. Ensure that the file is no longer needed before performing this operation.

----End

Convert an EC file into a replica file. (For example, convert all files in the **/src** directory into replica files.)

Step 1 Run the following command to create a temporary directory (for example, `/tmp/convert_tmp_dir`, which must be a directory that does not exist) to temporarily store converted data.

```
hdfs dfs -mkdir /tmp/convert_tmp_dir
```

Step 2 Run the following command to set `/tmp/convert_tmp_dir` to a replica policy.

```
hdfs ec -setPolicy -path /tmp/convert_tmp_dir -replicate
```

Step 3 Run the following DistCp command to copy files. In this case, all files are written into the temporary directory based on the replica policy.

```
hadoop distcp -numListstatusThreads 40 -update -delete -skipcrccheck -pbugpaxt /src /tmp/convert_tmp_dir
```

Step 4 Run the following command to move `/src` to `/src-to-delete`.

```
hdfs dfs -mv /src /src-to-delete
```

Step 5 Run the following command to move `/tmp/convert_tmp_dir` to `/src`.

```
hdfs dfs -mv /tmp/convert_tmp_dir /src
```

Step 6 Run the following command to delete `/src-to-delete`.

```
hdfs dfs -rm -r -f /src-to-delete
```

----End

9.30 Configuring the Observer NameNode to Process Read Requests

Scenario

In an HDFS cluster configured with HA, the active NameNode processes all client requests, and the standby NameNode reserves the latest metadata and block location information. However, in this architecture, the active NameNode is the bottleneck of client request processing. This bottleneck is more obvious in clusters with a large number of requests.

To address this issue, a new NameNode is introduced: an observer NameNode. Similar to the standby NameNode, the observer NameNode also reserves the latest metadata information and block location information. In addition, the observer NameNode can process read requests from clients in the same way as the active NameNode. In typical HDFS clusters with many read requests, the observer NameNode can be used to process read requests, reducing the active NameNode load and improving the cluster capability of processing requests.

NOTE

This section applies to MRS 3.x or later.

Impact on the System

- The active NameNode load can be reduced and the capability of HDFS cluster processing requests can be improved, which is especially obvious for large clusters.
- The client application configuration needs to be updated.

Prerequisites

- The HDFS cluster has been installed, the active and standby NameNodes are running properly, and the HDFS service is normal.
- The `/${BIGDATA_DATA_HOME}/namenode` partition has been created on the node where the observer NameNode is to be installed.

Procedure

The following steps describe how to configure the observer NameNode of a hacluster and enable it to process read requests. If there are multiple pairs of NameServices in the cluster and they are all in use, perform the following steps to configure the observer NameNode for each pair.

Step 1 Log in to FusionInsight Manager.

Step 2 Choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS** > **NameService Management**.

Step 3 Click **Add** next to **hacluster**.

Step 4 On the **Add NameNode** page, set **NameNode type** to **Observer** and click **Next**.

Step 5 On the **Assign Role** page, select the planned host, add the observer NameNode, and click **Next**.

NOTE

A maximum of five observer NameNodes can be added to each pair of NameServices.

Step 6 On the configuration page, configure the storage directory and port number of the NameNode as planned and click **Next**.

Step 7 Confirm the information, click **Submit**, and wait until the installation of the observer NameNode is complete.

Step 8 Restart the upper-layer components that depend on HDFS, update the client application configuration, and restart the client application.

----End

9.31 Performing Concurrent Operations on HDFS Files

Scenario

Performing this operation can concurrently modify file and directory permissions and access control tools in a cluster.

 **NOTE**

This section applies to MRS 3.x or later.

Impact on the System

Performing concurrent file modification operations in a cluster has adverse impacts on the cluster performance. Therefore, you are advised to do so when the cluster is idle.

Prerequisites

- The HDFS client or clients including HDFS has been installed. For example, the installation directory is **/opt/client**.
- Service component users are created by the MRS cluster administrator as required. In security mode, machine-machine users need to download the keytab file. A human-machine user needs to change the password upon the first login. (This operation is not required in normal mode.)

Procedure

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client installation directory:

```
cd /opt/client
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 If the cluster is in security mode, the user executing the DistCp command must belong to the **supergroup** group and run the following command to perform user authentication. In normal mode, user authentication is not required.

```
kinit Component service user
```

Step 5 Increase the JVM size of the client to prevent out of memory (OOM). (32 GB is recommended for 100 million files.)

 **NOTE**

The HDFS client exits abnormally and the error message "java.lang.OutOfMemoryError" is displayed after the HDFS client command is executed.

This problem occurs because the memory required for running the HDFS client exceeds the preset upper limit (128 MB by default). You can change the memory upper limit of the client by modifying **CLIENT_GC_OPTS** in *<Client installation path>/HDFS/component_env*. For example, if you want to set the upper limit to 1 GB, run the following command:

```
CLIENT_GC_OPTS="-Xmx1G"
```

After the modification, run the following command to make the modification take effect:

```
source <Client installation path>/bigdata_env
```

Step 6 Run the concurrent commands shown in the following table.

Command	Description	Function
hdfs quickcmds [-t threadsNumber] [-p principal] [-k keytab] -setrep <rep> <path> ...	<p>threadsNumber indicates the number of concurrent threads. The default value is the number of vCPUs of the local host.</p> <p>principal indicates the Kerberos user.</p> <p>keytab indicates the Keytab file.</p> <p>rep indicates the number of replicas.</p> <p>path indicates the HDFS directory.</p>	Used to concurrently set the number of copies of all files in a directory.
hdfs quickcmds [-t threadsNumber] [-p principal] [-k keytab] -chown [owner][: [group]] <path> ...	<p>threadsNumber indicates the number of concurrent threads. The default value is the number of vCPUs of the local host.</p> <p>principal indicates the Kerberos user.</p> <p>keytab indicates the Keytab file.</p> <p>owner indicates the owner.</p> <p>group indicates the group to which the user belongs.</p> <p>path indicates the HDFS directory.</p>	Used to concurrently set the owner group of all files in the directory.
hdfs quickcmds [-t threadsNumber] [-p principal] [-k keytab] -chmod <mode> <path> ...	<p>threadsNumber indicates the number of concurrent threads. The default value is the number of vCPUs of the local host.</p> <p>principal indicates the Kerberos user.</p> <p>keytab indicates the Keytab file.</p> <p>mode indicates the permission (for example, 754).</p> <p>path indicates the HDFS directory.</p>	Used to concurrently set permissions for all files in a directory.
hdfs quickcmds [-t threadsNumber] [-p principal] [-k keytab] -setfacl [{-b -k} {-m -x} <acl_spec>] <path> ...] [--set <acl_spec> <path> ...]	<p>threadsNumber indicates the number of concurrent threads. The default value is the number of vCPUs of the local host.</p> <p>principal indicates the Kerberos user.</p> <p>keytab indicates the Keytab file.</p> <p>acl_spec indicates the ACL list separated by commas (,).</p> <p>path indicates the HDFS directory.</p>	Used to concurrently set ACL information for all files in a directory.

----End

9.32 Introduction to HDFS Logs

Log Description

Log path: The default path of HDFS logs is `/var/log/Bigdata/hdfs/Role name`.

- NameNode: `/var/log/Bigdata/hdfs/nn` (run logs) and `/var/log/Bigdata/audit/hdfs/nn` (audit logs)
- DataNode: `/var/log/Bigdata/hdfs/dn` (run logs) and `/var/log/Bigdata/audit/hdfs/dn` (audit logs)
- ZKFC: `/var/log/Bigdata/hdfs/zkfc` (run logs) and `/var/log/Bigdata/audit/hdfs/zkfc` (audit logs)
- JournalNode: `/var/log/Bigdata/hdfs/jn` (run logs) and `/var/log/Bigdata/audit/hdfs/jn` (audit logs)
- Router: `/var/log/Bigdata/hdfs/router` (run logs) and `/var/log/Bigdata/audit/hdfs/router` (audit logs)
- HttpFS: `/var/log/Bigdata/hdfs/httpfs` (run logs) and `/var/log/Bigdata/audit/hdfs/httpfs` (audit logs)

Log archive rule: The automatic HDFS log compression function is enabled. By default, when the size of logs exceeds 100 MB, logs are automatically compressed into a log file named in the following format: `<Original log file name>-<yyyy-mm-dd_hh-mm-ss.[ID].log.zip`. A maximum of 100 latest compressed files are reserved. The number of compressed files can be configured on Manager.

Table 9-34 HDFS log list

Type	Name	Description
Run log	hadoop-<SSH_USER>-<process_name>-<hostname>.log	HDFS system log, which records most of the logs generated when the HDFS system is running.
	hadoop-<SSH_USER>-<process_name>-<hostname>.out	Log that records the HDFS running environment information.
	hadoop.log	Log that records the operation of the Hadoop client.

Type	Name	Description
	hdfs-period-check.log	Log that records scripts that are executed periodically, including automatic balancing, data migration, and JournalNode data synchronization detection.
	<process_name>-<SSH_USER>-<DATE>-<PID>-gc.log	Garbage collection log file
	postinstallDetail.log	Work log before the HDFS service startup and after the installation.
	hdfs-service-check.log	Log that records whether the HDFS service starts successfully.
	hdfs-set-storage-policy.log	Log that records the HDFS data storage policies.
	cleanupDetail.log	Log that records the cleanup logs about the uninstallation of the HDFS service.
	prestartDetail.log	Log that records cluster operations before the HDFS service startup.
	hdfs-recover-fsimage.log	Recovery log of the NameNode metadata.
	datanode-disk-check.log	Log that records the disk status check during the cluster installation and use.
	hdfs-availability-check.log	Log that check whether the HDFS service is available.
	hdfs-backup-fsimage.log	Backup log of the NameNode metadata.
	startDetail.log	Detailed log that records the HDFS service startup.

Type	Name	Description
	hdfs-blockplacement.log	Log that records the placement policy of HDFS blocks.
	upgradeDetail.log	Upgrade logs.
	hdfs-clean-acls-java.log	Log that records the clearing of deleted roles' ACL information by HDFS.
	hdfs-haCheck.log	Run log that checks whether the NameNode in active or standby state has obtained scripts.
	<process_name>-jvmpause.log	Log that records JVM pauses during process running.
	hadoop-<SSH_USER>-balancer-<hostname>.log	Run log of HDFS automatic balancing.
	hadoop-<SSH_USER>-balancer-<hostname>.out	Log that records information of the environment where HDFS executes automatic balancing.
	hdfs-switch-namenode.log	Run log that records the HDFS active/standby switchover.
	hdfs-router-admin.log	Run log of the mount table management operation
Tomcat logs	hadoop-omm-host1.out, httpfs-catalina.<DATE>.log, httpfs-host-manager.<DATE>.log, httpfs-localhost.<DATE>.log, httpfs-manager.<DATE>.log, localhost_access_web_log.log	Tomcat run log
Audit log	hdfs-audit-<process_name>.log ranger-plugin-audit.log	Audit log that records the HDFS operations (such as creating, deleting, modifying and querying files).
	SecurityAuth.audit	HDFS security audit log.

Log Level

Table 9-35 lists the log levels supported by HDFS. The log levels include FATAL, ERROR, WARN, INFO, and DEBUG. Logs of which the levels are higher than or equal to the set level will be printed by programs. The higher the log level is set, the fewer the logs are recorded.

Table 9-35 Log levels

Level	Description
FATAL	Indicates the critical error information about system running.
ERROR	Indicates the error information about system running.
WARN	Indicates that the current event processing exists exceptions.
INFO	Indicates that the system and events are running properly.
DEBUG	Indicates the system and system debugging information.

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page of HDFS by referring to [Modifying Cluster Service Configuration Parameters](#).
- Step 2** On the left menu bar, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.

 **NOTE**

The configurations take effect immediately without restarting the service.

----End

Log Formats

The following table lists the HDFS log formats.

Table 9-36 Log formats

Type	Format	Example
Run log	<yyyy-MM-dd HH:mm:ss,SSS> <Log level> <Name of the thread that generates the log> <Message in the log> <Location where the log event occurs>	2015-01-26 18:43:42,840 INFO IPC Server handler 40 on 8020 Rolling edit logs org.apache.hadoop.hdfs.server.namenode.FSEditLog.rollEditLog(FSEditLog.java:1096)
Audit log	<yyyy-MM-dd HH:mm:ss,SSS> <Log level> <Name of the thread that generates the log> <Message in the log> <Location where the log event occurs>	2015-01-26 18:44:42,607 INFO IPC Server handler 32 on 8020 allowed=true ugi=hbase (auth:SIMPLE) ip=/10.177.112.145 cmd=getfileinfo src=/hbase/WALs/hghoulaslx410,16020,1421743096083/hghoulaslx410%2C16020%2C1421743096083.1422268722795 dst=null perm=null org.apache.hadoop.hdfs.server.namenode.FSNameSystem\$DefaultAuditLogger.logAuditMessage(FSNamesystem.java:7950)

9.33 HDFS Performance Tuning

9.33.1 Improving Write Performance

Scenario

Improve the HDFS write performance by modifying the HDFS attributes.

 **NOTE**

This section applies to MRS 3.x or later.

Procedure

Navigation path for setting parameters:

On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS** > **Configurations** and select **All Configurations**. Enter a parameter name in the search box.

Table 9-37 Parameters for improving HDFS write performance

Parameter	Description	Default Value
dfs.datanode.drop.cache.behind.reads	<p>Specifies whether to enable a DataNode to automatically clear all data in the cache after the data in the cache is transferred to the client.</p> <p>If it is set to true, the cached data is discarded. The parameter needs to be configured on the DataNode.</p> <p>You are advised to set it to true if data is repeatedly read only a few times, so that the cache can be used by other operations. You are advised to set it to false if data is repeatedly read many times to enhance the reading speed.</p>	false
dfs.client-write-packet-size	<p>Specifies the size of the client write packet. When the HDFS client writes data to the DataNode, the data will be accumulated until a packet is generated. Then, the packet is transmitted over the network. This parameter specifies the size (unit: byte) of the data packet to be transmitted, which can be specified by each job.</p> <p>In the 10-Gigabit network, you can increase the value of this parameter to enhance the transmission throughput.</p>	262144

9.33.2 Improving Read Performance Using Client Metadata Cache

Scenario

Improve the HDFS read performance by using the client to cache the metadata for block locations.

NOTE

This function is recommended only for reading files that are not modified frequently. Because the data modification done on the server side by some other client is invisible to the cache client, which may cause the metadata obtained from the cache to be outdated.

This section applies to MRS 3.x or later.

Procedure

Navigation path for setting parameters:

On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS** > **Configurations**, select **All Configurations**, and enter the parameter name in the search box.

Table 9-38 Parameter configuration

Parameter	Description	Default Value
dfs.client.metadata.cache.enabled	Enables or disables the client to cache the metadata for block locations. Set this parameter to true and use it along with the dfs.client.metadata.cache.pattern parameter to enable the cache.	false
dfs.client.metadata.cache.pattern	Indicates the regular expression pattern of the path of the file to be cached. Only the metadata for block locations of these files is cached until the metadata expires. This parameter is valid only when dfs.client.metadata.cache.enabled is set to true . Example: /test.* indicates that all files whose paths start with /test are read. NOTE <ul style="list-style-type: none"> To ensure consistency, configure a specific mode to cache only files that are not frequently modified by other clients. The regular expression pattern verifies only the path of the URI, but not the schema and authority in the case of the Fully Qualified path. 	-
dfs.client.metadata.cache.expiry.sec	Indicates the duration for caching metadata. The cache entry becomes invalid after its caching time exceeds this duration. Even metadata that is frequently used during the caching process can become invalid. Time suffixes s/m/h can be used to indicate second, minute, and hour, respectively. NOTE If this parameter is set to 0s , the cache function is disabled.	60s
dfs.client.metadata.cache.max.entries	Indicates the maximum number of non-expired data items that can be cached at a time.	65536

 NOTE

Call `DFSClient#clearLocatedBlockCache()` to completely clear the client cache before it expires.

The sample usage is as follows:

```
FileSystem fs = FileSystem.get(conf);
DistributedFileSystem dfs = (DistributedFileSystem) fs;
DFSClient dfsClient = dfs.getClient();
dfsClient.clearLocatedBlockCache();
```

9.33.3 Improving the Connection Between the Client and NameNode Using Current Active Cache

Scenario

When HDFS is deployed in high availability (HA) mode with multiple NameNode instances, the HDFS client needs to connect to each NameNode in sequence to determine which is the active NameNode and use it for client operations.

Once the active NameNode is identified, its details can be cached and shared to all clients running on the client host. In this way, each new client first tries to load the details of the active NameNode from the cache and save the RPC call to the standby NameNode, which can help a lot in abnormal scenarios, for example, when the standby NameNode cannot be connected for a long time.

When a fault occurs and the other NameNode is switched to the active state, the cached details are updated to the information about the current active NameNode.

 NOTE

This section applies to MRS 3.x or later.

Procedure

Navigation path for setting parameters:

On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS** > **Configurations**, select **All Configurations**, and enter the parameter name in the search box.

Table 9-39 Configuration parameters

Parameter	Description	Default Value
dfs.client.failover.proxy.provider [nameservice ID]	Client Failover proxy provider class which creates the NameNode proxy using the authenticated protocol. If this parameter is set to org.apache.hadoop.hdfs.server.namenode.ha.BlackListingFailoverProxyProvider , you can use the NameNode blacklist feature on the HDFS client. If this parameter is set to org.apache.hadoop.hdfs.server.namenode.ha.ObserverReadProxyProvider , you can configure the observer NameNode to process read requests.	org.apache.hadoop.hdfs.server.namenode.ha.AdaptiveFailoverProxyProvider
dfs.client.failover.activeinfo.share.flag	Specifies whether to enable the cache function and share the detailed information about the current active NameNode with other clients. Set it to true to enable the cache function.	false
dfs.client.failover.activeinfo.share.path	Specifies the local directory for storing the shared files created by all clients in the host. If a cache area is to be shared by different users, the directory must have required permissions (for example, creating, reading, and writing cache files in the specified directory).	/tmp
dfs.client.failover.activeinfo.share.io.timeout.sec	(Optional) Used to control timeout. The cache file is locked when it is being read or written, and if the file cannot be locked within the specified time, the attempt to read or update the caches will be abandoned. The unit is second.	5

 **NOTE**

The cache files created by the HDFS client are reused by other clients, and thus these files will not be deleted from the local system. If this function is disabled, you may need to manually clear the data.

9.34 FAQ

9.34.1 NameNode Startup Is Slow

Question

The NameNode startup is slow when it is restarted immediately after a large number of files (for example, 1 million files) are deleted.

Answer

It takes time for the DataNode to delete the corresponding blocks after files are deleted. When the NameNode is restarted immediately, it checks the block information reported by all DataNodes. If a deleted block is found, the NameNode generates the corresponding INFO log information, as shown below:

```
2015-06-10 19:25:50,215 | INFO | IPC Server handler 36 on 25000 | BLOCK* processReport: blk_1075861877_2121067 on node 10.91.8.218:9866 size 10249 does not belong to any file | org.apache.hadoop.hdfs.server.blockmanagement.BlockManager.processReport(BlockManager.java:1854)
```

A log is generated for each deleted block. A file may contain one or more blocks. Therefore, after startup, the NameNode spends a large amount of time printing logs when a large number of files are deleted. As a result, the NameNode startup becomes slow.

To address this issue, the following operations can be performed to speed up the startup:

1. After a large number of files are deleted, wait until the DataNode deletes the corresponding blocks and then restart the NameNode.
You can run the *hdfs dfsadmin -report* command to check the disk space and check whether the files have been deleted.
2. If a large number of the preceding logs are generated, you can change the NameNode log level to **ERROR** so that the NameNode stops printing such logs.

After the NameNode is restarted, change the log level back to **INFO**. You do not need to restart the service after changing the log level.

9.34.2 DataNode Is Normal but Cannot Report Data Blocks

Question

The DataNode is normal, but cannot report data blocks. As a result, the existing data blocks cannot be used.

Answer

This error may occur when the number of data blocks in a data directory exceeds four times the upper limit (4 x 1 MB). And the DataNode generates the following error logs:

```
2015-11-05 10:26:32,936 | ERROR | DataNode:[[[DISK]file:/srv/BigData/hadoop/data1/dn/]] heartbeating to vm-210/10.91.8.210:8020 | Exception in BPOfferService for Block pool BP-805114975-10.91.8.210-1446519981645 (Datanode Uuid bcada350-0231-413b-bac0-8c65e906c1bb) service to vm-210/10.91.8.210:8020 | BPServiceActor.java:824 java.lang.IllegalStateException:com.google.protobuf.InvalidProtocolBufferException:Protocol message was
```

```

too large.May
be malicious.Use CodedInputStream.setSizeLimit() to increase the size limit. at
org.apache.hadoop.hdfs.protocol.BlockListAsLongs$BufferDecoder$1.next(BlockListAsLongs.java:369)
at org.apache.hadoop.hdfs.protocol.BlockListAsLongs$BufferDecoder$1.next(BlockListAsLongs.java:347) at
org.apache.hadoop.hdfs.
protocol.BlockListAsLongs$BufferDecoder.getBlockListAsLongs(BlockListAsLongs.java:325) at
org.apache.hadoop.hdfs.protocolPB.DatanodeProtocolClientSideTranslatorPB.
blockReport(DatanodeProtocolClientSideTranslatorPB.java:190) at
org.apache.hadoop.hdfs.server.datanode.BPServiceActor.blockReport(BPServiceActor.java:473)
at org.apache.hadoop.hdfs.server.datanode.BPServiceActor.offerService(BPServiceActor.java:685) at
org.apache.hadoop.hdfs.server.datanode.BPServiceActor.run(BPServiceActor.java:822)
at java.lang.Thread.run(Thread.java:745) Caused
by:com.google.protobuf.InvalidProtocolBufferException:Protocol message was too large.May be
malicious.Use CodedInputStream.setSizeLimit()
to increase the size limit. at
com.google.protobuf.InvalidProtocolBufferException.sizeLimitExceeded(InvalidProtocolBufferException.java:
110) at com.google.protobuf.CodedInputStream.refillBuffer(CodedInputStream.java:755)
at com.google.protobuf.CodedInputStream.readRawByte(CodedInputStream.java:769) at
com.google.protobuf.CodedInputStream.readRawVarint64(CodedInputStream.java:462) at
com.google.protobuf.
CodedInputStream.readInt64(CodedInputStream.java:363) at
org.apache.hadoop.hdfs.protocol.BlockListAsLongs$BufferDecoder$1.next(BlockListAsLongs.java:363)

```

The number of data blocks in the data directory is displayed as **Metric**. You can monitor its value through **http://<datanode-ip>:<http-port>/jmx**. If the value is greater than four times the upper limit (4 x 1 MB), you are advised to configure multiple drives and restart HDFS.

Recovery procedure:

1. Configure multiple data directories on the DataNode.

For example, configure multiple directories on the DataNode where only the / **data1/datadir** directory is configured:

```
<property> <name>dfs.datanode.data.dir</name> <value>/data1/datadir</value> </property>
```

Configure as follows:

```
<property> <name>dfs.datanode.data.dir</name> <value>/data1/datadir/,/data2/datadir,/data3/
datadir</value> </property>
```

NOTE

You are advised to configure multiple data directories on multiple disks. Otherwise, performance may be affected.

2. Restart the HDFS.
3. Perform the following operation to move the data to the new data directory:


```
mv /data1/datadir/current/finalized/subdir1 /data2/datadir/current/finalized/
subdir1
```
4. Restart the HDFS.

9.34.3 HDFS WebUI Cannot Properly Update Information About Damaged Data

Question

1. When errors occur in the **dfs.datanode.data.dir** directory of DataNode due to the permission or disk damage, HDFS WebUI does not display information about damaged data.
2. After errors are restored, HDFS WebUI does not timely remove related information about damaged data.

Answer

1. DataNode checks whether the disk is normal only when errors occur in file operations. Therefore, only when a data damage is detected and the error is reported to NameNode, NameNode displays information about the damaged data on HDFS WebUI.
2. After errors are fixed, you need to restart DataNode. During restarting DataNode, all data states are checked and damaged data information is uploaded to NameNode. Therefore, after errors are fixed, damaged data information is not displayed on the HDFS WebUI only by restarting DataNode.

9.34.4 Why Does the Distcp Command Fail in the Secure Cluster, Causing an Exception?

Question

Why distcp command fails in the secure cluster with the following error displayed?

Client side exception

```
Invalid arguments: Unexpected end of file from server
```

Server side exception

```
javax.net.ssl.SSLException: Unrecognized SSL message, plaintext connection?
```

Answer

The preceding error may occur if **webhdfs://** is used in the distcp command. The reason is that the big data cluster uses the HTTPS mechanism, that is, **dfs.http.policy** is set to **HTTPS_ONLY** in **core-site.xml** file. To avoid the error, replace **webhdfs://** with **swebhdfs://** in the file.

For example:

```
./hadoop distcp swwebhdfs://IP:PORT/testfile hdfs://IP:PORT/testfile1
```

9.34.5 Why Does DataNode Fail to Start When the Number of Disks Specified by dfs.datanode.data.dir Equals dfs.datanode.failed.volumes.tolerated?

Question

If the number of disks specified by **dfs.datanode.data.dir** is equal to the value of **dfs.datanode.failed.volumes.tolerated**, DataNode startup will fail.

Answer

By default, the failure of a single disk will cause the HDFS DataNode process to shut down, which results in the NameNode scheduling additional replicas for each block that is present on the DataNode. This causes needless replications of blocks that reside on disks that have not failed.

To prevent this, you can configure DataNodes to tolerate the failure of **dfs.data.dir** directories; use the **dfs.datanode.failed.volumes.tolerated** parameter in **hdfs-**

site.xml. For example, if the value for this parameter is 3, the DataNode will only shut down after four or more data directories have failed. This value is respected on DataNode startup.

When we are configuring tolerate volumes which should be always less than the configured volumes or else we can keep this as -1 which is equal to n-1 (where n is number of disks) then DataNode will not be shut down.

9.34.6 Why Does an Error Occur During DataNode Capacity Calculation When Multiple data.dir Are Configured in a Partition?

Question

DataNode capacity count incorrect if several data.dir configured in one disk partition.

Answer

Currently calculation will be done based on the disk like **df** command in linux. Ideally user should not configure multiple directories for same disk which will be huge impact on performance where all data will go to one disk.

Hence it is always better to configure like below:

For example:

if the machine is having disks like following:

```
host-4:~ # df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1       352G  11G  324G   4% /
udev            190G  252K  190G   1% /dev
tmpfs           190G   72K  190G   1% /dev/shm
/dev/sdb1       2.7T   74G  2.5T   3% /data1
/dev/sdc1       2.7T   75G  2.5T   3% /data2
/dev/sdd1       2.7T   73G  2.5T   3% /data
```

Suggested way of configuration:

```
<property>
<name>dfs.datanode.data.dir</name>
<value>/data1/datadir1,/data2/datadir1,/data3/datadir1</value>
</property>
```

Following is not recommended:

```
<property>
<name>dfs.datanode.data.dir</name>
<value>/data1/datadir1,/data2/datadir1,/data3/datadir1,/data1/datadir2,/data1/datadir3,/data2/datadir2,/
data2/datadir3,/data3/datadir2,/data3/datadir3</value>
</property>
```

9.34.7 Standby NameNode Fails to Be Restarted When the System Is Powered off During Metadata (Namespace) Storage

Question

When the standby NameNode is powered off during metadata (namespace) storage, it fails to be started and the following error information is displayed.

```
2015-12-04 11:49:12,121 | ERROR | main | Failed to load image from FS
ImageFile(file=/srv/BigData/namenode/current/fsimage_0000000000000096
080,
cpktTxId=0000000000000096080) | FSImage.java:685
java.io.IOException: Invalid MD5 file /srv/BigData/namenode/current/f
simage_0000000000000096080.md5:
the content " " does not match the expecte
d pattern.
at org.apache.hadoop.hdfs.util.MD5FileUtils.readStoredMd5(MD5FileUtil
s.java:92)
at org.apache.hadoop.hdfs.util.MD5FileUtils.readStoredMd5ForFile(MD5F
ileUtils.java:109)
at org.apache.hadoop.hdfs.server.namenode.FSImage.loadFSImage(FSImage
.java:975)
at org.apache.hadoop.hdfs.server.namenode.FSImage.loadFSImageFile(FSI
mage.java:744)
at org.apache.hadoop.hdfs.server.namenode.FSImage.loadFSImage(FSImage
.java:682)
at org.apache.hadoop.hdfs.server.namenode.FSImage.recoverTransitionRe
ad(FSImage.java:300)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFSImage(FS
Namesystem.java:968)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFromDisk(F
SNamesystem.java:675)
at org.apache.hadoop.hdfs.server.namenode.NameNode.loadNamesystem(Nam
eNode.java:625)
at org.apache.hadoop.hdfs.server.namenode.NameNode.initialize(NameNod
e.java:685)
at org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.ja
va:889)
at org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.ja
va:872)
at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(Nam
eNode.java:1580)
at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java
:1654)
```

Answer

When the standby NameNode is powered off during metadata (namespace) storage, it fails to be started and the MD5 file is damaged. Remove the damaged fsimage and start the standby NameNode to rectify the fault. After the rectification, the standby NameNode loads the previous fsimage and reproduces all edits.

Recovery procedure:

1. Run the following command to remove the damaged fsimage:
**rm -rf \${BIGDATA_DATA_HOME}/namenode/current/
fsimage_0000000000000096**
2. Start the standby NameNode.

9.34.8 Why Data in the Buffer Is Lost If a Power Outage Occurs During Storage of Small Files

Question

Why data in the buffer is lost if a power outage occurs during storage of small files?

Answer

Because of a power outage, the blocks in the buffer are not written to the disk immediately after the write operation is completed. To enable synchronization of blocks to the disk, set **dfs.datanode.synconclose** to **true** in the **hdfs-site.xml** file.

By default, **dfs.datanode.synconclose** is set to **false**. This improves the performance but can cause a buffer data loss in the case of a power outage, and therefore, it is recommended that **dfs.datanode.synconclose** be set to **true** even if this may affect the performance. You can determine whether to enable the synchronization function based on your actual situation.

9.34.9 Why Does Array Border-crossing Occur During FileInputFormat Split?

Question

When HDFS calls the FileInputFormat getSplit method, the ArrayIndexOutOfBoundsException: 0 appears in the following log:

```
java.lang.ArrayIndexOutOfBoundsException: 0
at org.apache.hadoop.mapred.FileInputFormat.identifyHosts(FileInputFormat.java:708)
at org.apache.hadoop.mapred.FileInputFormat.getSplitHostsAndCachedHosts(FileInputFormat.java:675)
at org.apache.hadoop.mapred.FileInputFormat.getSplits(FileInputFormat.java:359)
at org.apache.spark.rdd.HadoopRDD.getPartitions(HadoopRDD.scala:210)
at org.apache.spark.rdd.RDD$$anonfun$partitions$2.apply(RDD.scala:239)
at org.apache.spark.rdd.RDD$$anonfun$partitions$2.apply(RDD.scala:237)
at scala.Option.getOrElse(Option.scala:120)
at org.apache.spark.rdd.RDD.partitions(RDD.scala:237)
at org.apache.spark.rdd.MapPartitionsRDD.getPartitions(MapPartitionsRDD.scala:35)
```

Answer

The elements of each block correspondent frame are as below: `/default/rack0/;/default/rack0/datanodeip:port`.

The problem is due to a block damage or loss, making the block correspondent machine ip and port become null. Use **hdfs fsck** to check the file blocks health state when this problem occurs, and remove damaged block or restore the missing block to re-computing the task.

9.34.10 Why Is the Storage Type of File Copies DISK When the Tiered Storage Policy Is LAZY_PERSIST?

Question

When the storage policy of the file is set to **LAZY_PERSIST**, the storage type of the first replica should be **RAM_DISK**, and the storage type of other replicas should be **DISK**.

But why is the storage type of all copies shown as **DISK** actually?

Answer

When a user writes into a file whose storage policy is **LAZY_PERSIST**, three replicas are written one by one. The first replica is preferentially written into the DataNode where the client is located. The storage type of all replicas is **DISK** in the following scenarios:

- If the DataNode where the client is located does not have the RAM disk, the first replica is written into the disk of the DataNode where the client is located, and other replicas are written into the disks of other nodes.
- If the DataNode where the client is located has the RAM disk, and the value of **dfs.datanode.max.locked.memory** is not specified or smaller than the value of **dfs.blocksize**, the first replica is written into the disk of the DataNode where the client is located, and other replicas are written into the disks of other nodes.

9.34.11 The HDFS Client Is Unresponsive When the NameNode Is Overloaded for a Long Time

Question

When the NameNode node is overloaded (100% of the CPU is occupied), the NameNode is unresponsive. The HDFS clients that are connected to the overloaded NameNode fail to run properly. However, the HDFS clients that are newly connected to the NameNode will be switched to a backup NameNode and run properly.

Answer

The default configuration must be used (as described in [Table 9-40](#)) when the error preceding described occurs: the **keep alive** mechanism is enabled for the RPC connection between the HDFS client and the NameNode. The **keep alive** mechanism will keep the HDFS client waiting for the response from server and prevent the connection from being out timed, causing the unresponsiveness of the HDFS client.

Perform the following operations to the unresponsive HDFS client:

- Leave the HDFS client waiting. Once the CPU usage of the node where NameNode locates drops, the NameNode will obtain CPU resources and the HDFS client will receive a response.

- If you do not want to leave the HDFS client running, restart the application where the HDFS client locates to reconnect the HDFS client to another idle NameNode.

Procedure:

Configure the following parameters in the **core-site.xml** file on the client.

Table 9-40 Parameter description

Parameter	Description	Default Value
ipc.client.ping	<p>If the ipc.client.ping parameter is configured to true, the HDFS client will wait for the response from the server and periodically send the ping message to avoid disconnection caused by tcp timeout.</p> <p>If the ipc.client.ping parameter is configured to false, the HDFS client will set the value of ipc.ping.interval as the timeout time. If no response is received within that time, timeout occurs.</p> <p>To avoid the unresponsiveness of HDFS when the NameNode is overloaded for a long time, you are advised to set the parameter to false.</p>	true
ipc.ping.interval	<p>If the value of ipc.client.ping is true, ipc.ping.interval indicates the interval between sending the ping messages.</p> <p>If the value of ipc.client.ping is false, ipc.ping.interval indicates the timeout time for connection.</p> <p>To avoid the unresponsiveness of HDFS when the NameNode is overloaded for a long time, you are advised to set the parameter to a large value, for example 900000 (unit ms) to avoid timeout when the server is busy.</p>	60000

9.34.12 Can I Delete or Modify the Data Storage Directory in DataNode?

Question

- In DataNode, the storage directory of data blocks is specified by **dfs.datanode.data.dir**. Can I modify **dfs.datanode.data.dir** to modify the data storage directory?
- Can I modify files under the data storage directory?

Answer

During the system installation, you need to configure the **dfs.datanode.data.dir** parameter to specify one or more root directories.

- During the system installation, you need to configure the `dfs.datanode.data.dir` parameter to specify one or more root directories.
- Exercise caution when modifying `dfs.datanode.data.dir`. You can configure this parameter to add a new data root directory.
- Do not modify or delete data blocks in the storage directory. Otherwise, the data blocks will lose.

 **NOTE**

Similarly, do not delete the storage directory, or modify or delete data blocks under the directory using the following parameters:

- `dfs.namenode.edits.dir`
- `dfs.namenode.name.dir`
- `dfs.journalnode.edits.dir`

9.34.13 Blocks Miss on the NameNode UI After the Successful Rollback

Question

Why are some blocks missing on the NameNode UI after the rollback is successful?

Answer

This problem occurs because blocks with new IDs or genstamps may exist on the DataNode. The block files in the DataNode may have different generation flags and lengths from those in the rollback images of the NameNode. Therefore, the NameNode rejects these blocks in the DataNode and marks the files as damaged.

Scenarios:

1. Before an upgrade:
Client A writes some data to file X. (Assume A bytes are written.)
2. During an upgrade:
Client A still writes data to file X. (The data in the file is A + B bytes.)
3. After an upgrade:
Client A completes the file writing. The final data is A + B bytes.
4. Rollback started:
The status will be rolled back to the status before the upgrade. That is, file X in NameNode will have A bytes, but block files in DataNode will have A + B bytes.

Recovery procedure:

1. Obtain the list of damaged files from NameNode web UI or run the following command to obtain:
`hdfs fsck <filepath> -list-corruptfileblocks`
2. Run the following command to delete unnecessary files:

hdfs fsck <corrupt file path> - delete

NOTE

Deleting a file is a high-risk operation. Ensure that the files are no longer needed before performing this operation.

3. For the required files, run the **fsck** command to obtain the block list and block sequence.
 - In the block sequence table provided, use the block ID to search for the data directory in the DataNode and download the corresponding block from the DataNode.
 - Write all such block files in appending mode based on the sequence to construct the original file.

Example:

File 1--> blk_1, blk_2, blk_3

Create a file by combining the contents of all three block files from the same sequence.

- Delete the old file from HDFS and rewrite the new file.

9.34.14 Why Is "java.net.SocketException: No buffer space available" Reported When Data Is Written to HDFS

Question

Why is an "java.net.SocketException: No buffer space available" exception reported when data is written to HDFS?

This problem occurs when files are written to the HDFS. Check the error logs of the client and DataNode.

The client logs are as follows:

Figure 9-10 Client logs

```

2017-07-05 21:58:06.459 INFO [htable-pool3-t1] ipc.AbstractRpcClient: RPC Server Kerberos principal name for service=ClientService is hbase/hadoop.hadoop123.com@HADOOP12
2017-07-05 21:58:06.893 WARN [main] mapreduce.LoadIncrementalHFiles: Skipping non-directory hdfs://hacluster/HBaseTest/bulkload_output/_SUCCESS
2017-07-05 21:59:13.211 WARN [main] hdfs.BlockReaderFactory: I/O error constructing remote block reader.
java.net.SocketException: No buffer space available
    at sun.nio.ch.Net.connect0(Native Method)
    at sun.nio.ch.Net.connect(Net.java:454)
    at sun.nio.ch.SocketChannelImpl.connect(SocketChannelImpl.java:446)
    at sun.nio.ch.SocketChannelImpl.connect(SocketChannelImpl.java:648)
    at org.apache.hadoop.net.SocketIOWithTimeout.connect(SocketIOWithTimeout.java:192)
    at org.apache.hadoop.net.NetUtils.connect(NetUtils.java:531)
    at org.apache.hadoop.hdfs.DFSClient.newConnectedPeer(DFSClient.java:3345)
    at org.apache.hadoop.hdfs.BlockReaderFactory.nextTcpPeer(BlockReaderFactory.java:789)
    at org.apache.hadoop.hdfs.BlockReaderFactory.getRemoteBlockReaderFromTcp(BlockReaderFactory.java:706)
    at org.apache.hadoop.hdfs.BlockReaderFactory.build(BlockReaderFactory.java:369)
    at org.apache.hadoop.hdfs.DFSInputStream.getBlockReader(DFSInputStream.java:713)
    at org.apache.hadoop.hdfs.DFSInputStream.blockSeekTo(DFSInputStream.java:663)
    at org.apache.hadoop.hdfs.DFSInputStream.readWithStrategy(DFSInputStream.java:919)
    at org.apache.hadoop.hdfs.DFSInputStream.read(DFSInputStream.java:973)
    at java.io.DataInputStream.readFully(DataInputStream.java:195)
    at org.apache.hadoop.hbase.io.hfile.FixedFileTrailer.readFromStream(FixedFileTrailer.java:391)
    at org.apache.hadoop.hbase.io.hfile.HFile.isHFileFormat(HFile.java:578)
    at org.apache.hadoop.hbase.io.hfile.HFile.isHFileFormat(HFile.java:560)
    at org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles.visitBulkHFiles(LoadIncrementalHFiles.java:229)
    at org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles.discoverLoadQueue(LoadIncrementalHFiles.java:281)
    at org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles.prepareFileQueue(LoadIncrementalHFiles.java:452)
    at org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles.doBulkLoad(LoadIncrementalHFiles.java:365)
    at org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles.run(LoadIncrementalHFiles.java:331)
    at org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles.run(LoadIncrementalHFiles.java:1197)
    at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:70)
    at org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles.main(LoadIncrementalHFiles.java:1114)
2017-07-05 21:59:13.215 WARN [main] hdfs.DFSClient: Failed to connect to /192.168.152.128:25009 for block BP-1969348819-192.168.199.5-1497961637591:blk_1107301222_335745
ffer space available
java.net.SocketException: No buffer space available
    at sun.nio.ch.Net.connect0(Native Method)
    at sun.nio.ch.Net.connect(Net.java:454)
    at sun.nio.ch.SocketChannelImpl.connect(SocketChannelImpl.java:648)
    at org.apache.hadoop.net.SocketIOWithTimeout.connect(SocketIOWithTimeout.java:192)
    at org.apache.hadoop.net.NetUtils.connect(NetUtils.java:531)
    at org.apache.hadoop.hdfs.DFSClient.newConnectedPeer(DFSClient.java:3345)

```

DataNode logs are as follows:

```

2017-07-24 20:43:39,269 | ERROR | DataXceiver for client DFSClient_NONMAPREDUCE_996005058_86
at /192.168.164.155:40214 [Receiving block
BP-1287143557-192.168.199.6-1500707719940:blk_1074269754_528941 with io weight 10] |
DataNode{data=FSDataset{dirpath='[/srv/BigData/hadoop/data1/dn/current, /srv/BigData/hadoop/
data2/dn/current, /srv/BigData/hadoop/data3/dn/current, /srv/BigData/hadoop/data4/dn/current, /srv/
BigData/hadoop/data5/dn/current, /srv/BigData/hadoop/data6/dn/current, /srv/BigData/hadoop/data7/dn/
current]'}, localName='192-168-164-155:9866', datanodeUuid='a013e29c-4e72-400c-bc7b-bbbf0799604c',
xmitsInProgress=0}:Exception transferring block
BP-1287143557-192.168.199.6-1500707719940:blk_1074269754_528941 to mirror 192.168.202.99:9866:
java.net.SocketException: No buffer space available | DataXceiver.java:870
2017-07-24 20:43:39,269 | INFO | DataXceiver for client DFSClient_NONMAPREDUCE_996005058_86
at /192.168.164.155:40214 [Receiving block
BP-1287143557-192.168.199.6-1500707719940:blk_1074269754_528941 with io weight 10] | opWriteBlock
BP-1287143557-192.168.199.6-1500707719940:blk_1074269754_528941 received exception
java.net.SocketException: No buffer space available | DataXceiver.java:933
2017-07-24 20:43:39,270 | ERROR | DataXceiver for client DFSClient_NONMAPREDUCE_996005058_86
at /192.168.164.155:40214 [Receiving block
BP-1287143557-192.168.199.6-1500707719940:blk_1074269754_528941 with io weight 10] |
192-168-164-155:9866:DataXceiver error processing WRITE_BLOCK operation src: /192.168.164.155:40214
dst: /192.168.164.155:9866 | DataXceiver.java:304 java.net.SocketException: No buffer space available
at sun.nio.ch.Net.connect0(Native Method)
at sun.nio.ch.Net.connect(Net.java:454)
at sun.nio.ch.Net.connect(Net.java:446)
at sun.nio.ch.SocketChannelImpl.connect(SocketChannelImpl.java:648)
at org.apache.hadoop.net.SocketIOWithTimeout.connect(SocketIOWithTimeout.java:192)
at org.apache.hadoop.net.NetUtils.connect(NetUtils.java:531)
at org.apache.hadoop.net.NetUtils.connect(NetUtils.java:495)
at org.apache.hadoop.hdfs.server.datanode.DataXceiver.writeBlock(DataXceiver.java:800)
at org.apache.hadoop.hdfs.protocol.datatransfer.Receiver.opWriteBlock(Receiver.java:138)
at org.apache.hadoop.hdfs.protocol.datatransfer.Receiver.processOp(Receiver.java:74)
at org.apache.hadoop.hdfs.server.datanode.DataXceiver.run(DataXceiver.java:265)
at java.lang.Thread.run(Thread.java:748)

```

Answer

The preceding problem may be caused by network memory exhaustion.

You can increase the threshold of the network device based on the actual scenario.

Example:

```

[root@xxxx ~]# cat /proc/sys/net/ipv4/neigh/default/gc_thresh*
128
512
1024
[root@xxxx ~]# echo 512 > /proc/sys/net/ipv4/neigh/default/gc_thresh1
[root@xxxx ~]# echo 2048 > /proc/sys/net/ipv4/neigh/default/gc_thresh2
[root@xxxx ~]# echo 4096 > /proc/sys/net/ipv4/neigh/default/gc_thresh3
[root@xxxx ~]# cat /proc/sys/net/ipv4/neigh/default/gc_thresh*
512
2048
4096

```

You can also add the following parameters to the `/etc/sysctl.conf` file. The configuration takes effect even if the host is restarted.

```

net.ipv4.neigh.default.gc_thresh1 = 512
net.ipv4.neigh.default.gc_thresh2 = 2048
net.ipv4.neigh.default.gc_thresh3 = 4096

```

9.34.15 Why are There Two Standby NameNodes After the active NameNode Is Restarted?

Question

Why are there two standby NameNodes after the active NameNode is restarted?

When this problem occurs, check the ZooKeeper and ZooKeeper FC logs. You can find that the sessions used for the communication between the ZooKeeper server and client (ZKFC) are inconsistent. The session ID of the ZooKeeper server is **0x164cb2b3e4b36ae4**, and the session ID of the ZooKeeper FC is **0x144cb2b3e4b36ae4**. Such inconsistency means that the data interaction between the ZooKeeper server and ZKFC fails.

Content of the ZooKeeper log is as follows:

```
2015-04-15 21:24:54,257 | INFO | CommitProcessor:22 | Established session 0x164cb2b3e4b36ae4 with negotiated timeout 45000 for client /192.168.0.117:44586 | org.apache.zookeeper.server.ZooKeeperServer.finishSessionInit(ZooKeeperServer.java:623)
2015-04-15 21:24:54,261 | INFO | NIOServerCxn.Factory:192-168-0-114/192.168.0.114:2181 | Successfully authenticated client: authenticationID=hdfs/hadoop@<System domain name>; authorizationID=hdfs/hadoop@<System domain name>. | org.apache.zookeeper.server.auth.SaslServerCallbackHandler.handleAuthorizeCallback(SaslServerCallbackHandler.java:118)
2015-04-15 21:24:54,261 | INFO | NIOServerCxn.Factory:192-168-0-114/192.168.0.114:2181 | Setting authorizedID: hdfs/hadoop@<System domain name> | org.apache.zookeeper.server.auth.SaslServerCallbackHandler.handleAuthorizeCallback(SaslServerCallbackHandler.java:134)
2015-04-15 21:24:54,261 | INFO | NIOServerCxn.Factory:192-168-0-114/192.168.0.114:2181 | adding SASL authorization for authorizationID: hdfs/hadoop@<System domain name> | org.apache.zookeeper.server.ZooKeeperServer.processSasl(ZooKeeperServer.java:1009)
2015-04-15 21:24:54,262 | INFO | ProcessThread(sid:22 cport:-1): | Got user-level KeeperException when processing sessionid:0x164cb2b3e4b36ae4 type:create cxid:0x3 zxid:0x20009fafc txntype:-1 reqpath:n/a Error Path:/hadoop-ha/hacluster/ActiveStandbyElectorLock Error:KeeperErrorCode = NodeExists for /hadoop-ha/hacluster/ActiveStandbyElectorLock | org.apache.zookeeper.server.PrepareRequestProcessor.pRequest(PrepareRequestProcessor.java:648)
```

Content of the ZKFC log is as follows:

```
2015-04-15 21:24:54,237 | INFO | main-SendThread(192-168-0-114:2181) | Socket connection established to 192-168-0-114/192.168.0.114:2181, initiating session | org.apache.zookeeper.ClientCnxn.$SendThread.primeConnection(ClientCnxn.java:854)
2015-04-15 21:24:54,257 | INFO | main-SendThread(192-168-0-114:2181) | Session establishment complete on server 192-168-0-114/192.168.0.114:2181, sessionid = 0x144cb2b3e4b36ae4, negotiated timeout = 45000 | org.apache.zookeeper.ClientCnxn.$SendThread.onConnected(ClientCnxn.java:1259)
2015-04-15 21:24:54,260 | INFO | main-EventThread | EventThread shut down | org.apache.zookeeper.ClientCnxn.$EventThread.run(ClientCnxn.java:512)
2015-04-15 21:24:54,262 | INFO | main-EventThread | Session connected. | org.apache.hadoop.ha.ActiveStandbyElector.processWatchEvent(ActiveStandbyElector.java:547)
2015-04-15 21:24:54,264 | INFO | main-EventThread | Successfully authenticated to ZooKeeper using SASL. | org.apache.hadoop.ha.ActiveStandbyElector.processWatchEvent(ActiveStandbyElector.java:573)
```

Answer

- Cause Analysis
After the active NameNode restarts, the temporary node **/hadoop-ha/hacluster/ActiveStandbyElectorLock** created on ZooKeeper is deleted. After the standby NameNode receives that information that the **/hadoop-ha/hacluster/ActiveStandbyElectorLock** node is deleted, the standby NameNode creates the **/hadoop-ha/hacluster/ActiveStandbyElectorLock** node in ZooKeeper in order to switch to the active NameNode. However, when the standby NameNode connects with ZooKeeper through the client ZKFC, the session ID of ZKFC differs from that of ZooKeeper due to network issues, overload CPU, or overload clusters. In this case, the watcher of the standby NameNode fails to detect that the temporary node has been successfully created, and fails to consider the standby NameNode as the active NameNode. After the original active NameNode restarts, it detects that the **/hadoop-ha/hacluster/ActiveStandbyElectorLock** already exists and becomes the standby NameNode. Therefore, both NameNodes are standby NameNodes.

- Solution
You are advised to restart two ZKFCs of HDFS on FusionInsight Manager.

9.34.16 When Does a Balance Process in HDFS, Shut Down and Fail to be Executed Again?

Question

After I start a Balance process in HDFS, the process is shut down abnormally. If I attempt to execute the Balance process again, it fails again.

Answer

After a Balance process is executed in HDFS, another Balance process can be executed only after the **/system/balancer.id** file is automatically released.

However, if a Balance process is shut down abnormally, the **/system/balancer.id** has not been released when the Balance is executed again, which triggers the **append /system/balancer.id** operation.

- If the time spent on releasing the **/system/balancer.id** file exceeds the soft-limit lease period 60 seconds, executing the Balance process again triggers the append operation, which preempts the lease. The last block is in construction or under recovery status, which triggers the block recovery operation. The **/system/balancer.id** file cannot be closed until the block recovery completes. Therefore, the append operation fails.

After the **append /system/balancer.id** operation fails, the exception message **RecoveryInProgressException** is displayed.

```
org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.protocol.RecoveryInProgressException):  
Failed to APPEND_FILE /system/balancer.id for DFSClient because lease recovery is in progress. Try  
again later.
```

- If the time spent on releasing the **/system/balancer.id** file is within 60 seconds, the original client continues to own the lease and the exception **AlreadyBeingCreatedException** occurs and null is returned to the client. The following exception message is displayed on the client:

```
java.io.IOException: Cannot create any NameNode Connectors.. Exiting...
```

Either of the following methods can be used to solve the problem:

- Execute the Balance process again after the hard-limit lease period expires for 1 hour, when the original client has released the lease.
- Delete the **/system/balancer.id** file before executing the Balance process again.

9.34.17 "This page can't be displayed" Is Displayed When Internet Explorer Fails to Access the Native HDFS UI

Question

Occasionally, Internet Explorer 9, Explorer 10, or Explorer 11 fails to access the native HDFS UI.

Symptom

Internet Explorer 9, Explorer 10, or Explorer 11 fails to access the native HDFS UI, as shown in the following figure.

Turn on TLS 1.0, TLS 1.1, and TLS 1.2 in Advanced settings and try connecting to

Cause

Some Internet Explorer 9, Explorer 10, or Explorer 11 versions fail to handle SSL handshake issues, causing access failure.

Solution

Refresh the page.

9.34.18 NameNode Fails to Be Restarted Due to EditLog Discontinuity

Question

If a JournalNode server is powered off, the data directory disk is fully occupied, and the network is abnormal, the EditLog sequence number on the JournalNode is inconsecutive. In this case, the NameNode restart may fail.

Symptom

The NameNode fails to be restarted. The following error information is reported in the NameNode run logs:

```
2019-11-08 16:30:28,399 | ERROR | main | Failed to start namenode. | NameNode.java:1732
java.io.IOException: There appears to be a gap in the edit log. We expected txid 13698019, but got txid 13698088.
    at org.apache.hadoop.hdfs.server.namenode.MetaRecoveryContext.editLogLoaderPrompt(MetaRecoveryContext.java:94)
    at org.apache.hadoop.hdfs.server.namenode.FSEditLogLoader.loadEditRecords(FSEditLogLoader.java:278)
    at org.apache.hadoop.hdfs.server.namenode.FSEditLogLoader.loadFSEdits(FSEditLogLoader.java:188)
    at org.apache.hadoop.hdfs.server.namenode.FSImage.loadEdits(FSImage.java:924)
    at org.apache.hadoop.hdfs.server.namenode.FSImage.loadFSImage(FSImage.java:771)
    at org.apache.hadoop.hdfs.server.namenode.FSImage.recoverTransitionRead(FSImage.java:331)
    at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFSImage(FSNamesystem.java:1108)
    at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.loadFromDisk(FSNamesystem.java:727)
    at org.apache.hadoop.hdfs.server.namenode.NameNode.loadNamesystem(NameNode.java:638)
    at org.apache.hadoop.hdfs.server.namenode.NameNode.initialize(NameNode.java:700)
    at org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.java:943)
    at org.apache.hadoop.hdfs.server.namenode.NameNode.<init>(NameNode.java:916)
    at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(NameNode.java:1655)
    at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1725)
```

Solution

1. Find the active NameNode before the restart, go to its data directory (you can obtain the directory, such as **/srv/BigData/namenode/current** by checking the configuration item **dfs.namenode.name.dir**), and obtain the sequence number of the latest FsImage file, as shown in the following figure:

```

-rw-----, 1 omm wheel      574 Oct  2 01:12 edits_0000000000013259401-0000000000:
-rw-----, 1 omm wheel      575 Oct  2 01:13 edits_0000000000013259409-0000000000:
-rw-----, 1 omm wheel       42 Oct  2 01:13 edits_0000000000013259417-0000000000:
-rw-----, 1 omm wheel 1048576 Nov  8 16:01 edits_inprogress_0000000000013698088
-rw-----, 1 omm wheel 314803 Nov  8 15:53 fsimage_0000000000013698018
-rw-----, 1 omm wheel      62 Nov  8 15:53 fsimage_0000000000013698018.md5
-rw-----, 1 omm wheel 314803 Nov  8 15:56 fsimage_0000000000013698050
-rw-----, 1 omm wheel      62 Nov  8 15:56 fsimage_0000000000013698050.md5
-rw-----, 1 omm wheel 314803 Nov  8 15:59 fsimage_0000000000013698066
-rw-----, 1 omm wheel      62 Nov  8 15:59 fsimage_0000000000013698066.md5
-rw-----, 1 omm wheel      9 Oct  2 01:13 seen_txid
-rw-----, 1 omm wheel     187 Nov  8 15:59 VERSION

```

2. Check the data directory of each JournalNode (you can obtain the directory such as `/srv/BigData/journalnode/hacluster/current` by checking the value of the configuration item `dfs.journalnode.edits.dir`), and check whether the sequence number starting from that obtained in step 1 is consecutive in edits files. That is, you need to check whether the last sequence number of the previous edits file is consecutive with the first sequence number of the next edits file. (As shown in the following figure, `edits_0000000000013259231-0000000000013259237` and `edits_0000000000013259239-0000000000013259246` are not consecutive.)

```

-rw-----, 1 omm wheel      576 Oct  2 00:41 edits_0000000000013259151-0000000000013259158
-rw-----, 1 omm wheel      575 Oct  2 00:43 edits_0000000000013259159-0000000000013259166
-rw-----, 1 omm wheel      576 Oct  2 00:43 edits_0000000000013259167-0000000000013259174
-rw-----, 1 omm wheel      575 Oct  2 00:45 edits_0000000000013259175-0000000000013259182
-rw-----, 1 omm wheel      575 Oct  2 00:45 edits_0000000000013259183-0000000000013259190
-rw-----, 1 omm wheel      576 Oct  2 00:47 edits_0000000000013259191-0000000000013259198
-rw-----, 1 omm wheel      575 Oct  2 00:48 edits_0000000000013259199-0000000000013259206
-rw-----, 1 omm wheel      575 Oct  2 00:49 edits_0000000000013259207-0000000000013259214
-rw-----, 1 omm wheel      575 Oct  2 00:50 edits_0000000000013259215-0000000000013259222
-rw-----, 1 omm wheel      573 Oct  2 00:51 edits_0000000000013259223-0000000000013259230
-rw-----, 1 omm wheel      571 Oct  2 00:52 edits_0000000000013259231-0000000000013259237
-rw-----, 1 omm wheel      576 Oct  2 00:53 edits_0000000000013259239-0000000000013259246
-rw-----, 1 omm wheel      575 Oct  2 00:54 edits_0000000000013259247-0000000000013259254
-rw-----, 1 omm wheel      576 Oct  2 00:55 edits_0000000000013259255-0000000000013259262
-rw-----, 1 omm wheel      42 Oct  2 00:56 edits_0000000000013259263-0000000000013259264
-rw-----, 1 omm wheel     1107 Oct  2 00:57 edits_0000000000013259265-0000000000013259278
-rw-----, 1 omm wheel      42 Oct  2 00:58 edits_0000000000013259279-0000000000013259280
-rw-----, 1 omm wheel     1109 Oct  2 00:59 edits_0000000000013259281-0000000000013259294
-rw-----, 1 omm wheel      42 Oct  2 01:00 edits_0000000000013259295-0000000000013259296
-rw-----, 1 omm wheel     1299 Oct  2 01:01 edits_0000000000013259297-0000000000013259312
-rw-----, 1 omm wheel      260 Oct  2 01:02 edits_0000000000013259313-0000000000013259316
-rw-----, 1 omm wheel      984 Oct  2 01:03 edits_0000000000013259317-0000000000013259328
-rw-----, 1 omm wheel      572 Oct  2 01:04 edits_0000000000013259329-0000000000013259336
-rw-----, 1 omm wheel      575 Oct  2 01:05 edits_0000000000013259337-0000000000013259344
-rw-----, 1 omm wheel      983 Oct  2 01:06 edits_0000000000013259345-0000000000013259356

```

3. If the edits files are not consecutive, check whether the edits files with the related sequence number exist in the data directories of other JournalNodes or NameNode. If the edits files can be found, copy a consecutive segment to the JournalNode.
4. In this way, all inconsecutive edits files are restored.
5. Restart the NameNode and check whether the restart is successful. If the fault persists, contact Technical support.

10 Using Hive

10.1 Using Hive from Scratch

Hive is a data warehouse framework built on Hadoop. It maps structured data files to a database table and provides SQL-like functions to analyze and process data. It also allows you to quickly perform simple MapReduce statistics using SQL-like statements without the need of developing a specific MapReduce application. It is suitable for statistical analysis of data warehouses.

Background

Suppose a user develops an application to manage users who use service A in an enterprise. The procedure of operating service A on the Hive client is as follows:

Operations on common tables:

- Create the **user_info** table.
- Add users' educational backgrounds and professional titles to the table.
- Query user names and addresses by user ID.
- Delete the user information table after service A ends.

Table 10-1 User information

ID	Name	Gender	Age	Address
12005000201	A	Male	19	City A
12005000202	B	Female	23	City B
12005000203	C	Male	26	City C
12005000204	D	Male	18	City D
12005000205	E	Female	21	City E
12005000206	F	Male	32	City F
12005000207	G	Female	29	City G

ID	Name	Gender	Age	Address
12005000208	H	Female	30	City H
12005000209	I	Male	26	City I
12005000210	J	Female	25	City J

Procedure

Step 1 Download the client configuration file.

- For versions earlier than MRS 3.x, perform the following operations:
 - a. Log in to MRS Manager. For details, see [Accessing Manager](#). Then, choose **Services**.
 - b. Click **Download Client**.
Set **Client Type** to **Only configuration files**, **Download to** to **Server**, and click **OK** to generate the client configuration file. The generated file is saved in the **/tmp/MRS-client** directory on the active management node by default.
- For MRS 3.x or later, perform the following operations:
 - a. Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
 - b. Choose **Cluster** > *Name of the desired cluster* > **Dashboard** > **More** > **Download Client**.
 - c. Download the cluster client.
Set **Select Client Type** to **Configuration Files Only**, select a platform type, and click **OK** to generate the client configuration file which is then saved in the **/tmp/FusionInsight-Client/** directory on the active management node by default.

Step 2 Log in to the active management node of Manager.

- For versions earlier than MRS 3.x, perform the following operations:
 - a. On the MRS console, click **Clusters**, choose **Active Clusters**, and click a cluster name. On the **Nodes** tab, view the node names. The node whose name contains **master1** is the Master1 node, and the node whose name contains **master2** is the Master2 node.
The active and standby management nodes of MRS Manager are installed on Master nodes by default. Because Master1 and Master2 are switched over in active and standby mode, Master1 is not always the active management node of MRS Manager. Run a command in Master1 to check whether Master1 is active management node of MRS Manager. For details about the command, see [Step 2.d](#).
 - b. Log in to the Master1 node using the password as user **root**. For details, see [Logging In to a Cluster](#).
 - c. Run the following commands to switch to user **omm**:
sudo su - root
su - omm

- d. Run the following command to check the active management node of MRS Manager:

```
sh ${BIGDATA_HOME}/om-0.0.1/sbin/status-oms.sh
```

In the command output, the node whose **HAActive** is **active** is the active management node, and the node whose **HAActive** is **standby** is the standby management node. In the following example, **mgtomsdat-sh-3-01-1** is the active management node, and **mgtomsdat-sh-3-01-2** is the standby management node.

```
Ha mode
double
NodeName      HostName      HAVersion      StartTime
HAActive      HAAllResOK    HARunPhase
192-168-0-30  mgtomsdat-sh-3-01-1  V100R001C01    2014-11-18 23:43:02
active      normal        Activated
192-168-0-24  mgtomsdat-sh-3-01-2  V100R001C01    2014-11-21 07:14:02
standby    normal        Deactivated
```

- e. Log in to the active management node as user **root**, for example, node **192-168-0-30**.
- For MRS 3.x or later, perform the following operations:
 - a. Log in to any node where Manager is deployed as user **root**.
 - b. Run the following command to identify the active and standby nodes:

```
sh ${BIGDATA_HOME}/om-server/om/sbin/status-oms.sh
```

In the command output, the value of **HAActive** for the active management node is **active**, and that for the standby management node is **standby**. In the following example, **node-master1** is the active management node, and **node-master2** is the standby management node.

```
HAMode
double
NodeName      HostName      HAVersion      StartTime      HAActive
HAAllResOK    HARunPhase
192-168-0-30  node-master1  V100R001C01    2020-05-01 23:43:02  active
normal        Activated
192-168-0-24  node-master2  V100R001C01    2020-05-01 07:14:02
standby      normal        Deactivated
```

- c. Log in to the primary management node as user **root** and run the following command to switch to user **omm**:

```
sudo su - omm
```

Step 3 Run the following command to go to the client installation directory:

```
cd /opt/client
```

The cluster client has been installed in advance. The following client installation directory is used as an example. Change it based on the site requirements.

Step 4 Run the following command to update the client configuration for the active management node.

```
sh refreshConfig.sh /opt/client Full path of the client configuration file package
```

For example, run the following command:

```
sh refreshConfig.sh /opt/client /tmp/FusionInsight-Client/
FusionInsight_Cluster_1_Services_Client.tar
```

If the following information is displayed, the configurations have been updated successfully.

```
ReFresh components client config is complete.  
Succeed to refresh components client config.
```

 **NOTE**

You can refer to Method 2 in [Updating a Client](#) to perform operations in steps [Step 1](#) to [Step 4](#).

Step 5 Use the client on a Master node.

1. On the active management node, for example, **192-168-0-30**, run the following command to switch to the client directory, for example, **/opt/client**.

```
cd /opt/client
```

2. Run the following command to configure environment variables:

```
source bigdata_env
```

3. If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the current user:

```
kinit MRS cluster user
```

Example: user **kinit hiveuser**

The current user must have the permission to create Hive tables. To create a role with the permission, refer to [Creating a Role](#). To bind the role to the current user, refer to [Creating a User](#). If Kerberos authentication is disabled, skip this step.

4. Run the client command of the Hive component directly.

```
beeline
```

Step 6 Run the Hive client command to implement service A.

Operations on internal tables:

1. Create the **user_info** user information table according to [Table 10-1](#) and add data to it.

```
create table user_info(id string,name string,gender string,age int,addr string);
```

```
insert into table user_info(id,name,gender,age,addr)  
values("12005000201","A","Male",19,"City A");
```

2. Add users' educational backgrounds and professional titles to the **user_info** table.

For example, to add educational background and title information about user 12005000201, run the following command:

```
alter table user_info add columns(education string,technical string);
```

3. Query user names and addresses by user ID.

For example, to query the name and address of user 12005000201, run the following command:

```
select name,addr from user_info where id='12005000201';
```

4. Delete the user information table.

```
drop table user_info;
```

Operations on external partition tables:

Create an external partition table and import data.

1. Create a path for storing external table data.

```
hdfs dfs -mkdir /hive/  
hdfs dfs -mkdir /hive/user_info
```

2. Create a table.

```
create external table user_info(id string,name string,gender string,age  
int,addr string) partitioned by(year string) row format delimited fields  
terminated by ' ' lines terminated by '\n' stored as textfile location '/hive/  
user_info';
```

 NOTE

fields terminated indicates delimiters, for example, spaces.

lines terminated indicates line breaks, for example, \n.

/hive/user_info indicates the path of the data file.

3. Import data.

- a. Execute the insert statement to insert data.

```
insert into user_info partition(year="2018") values  
("12005000201","A","Male",19,"City A");
```

- b. Run the **load data** command to import file data.

- i. Create a file based on the data in [Table 10-1](#). For example, the file name is **txt.log**. Fields are separated by space, and the line feed characters are used as the line breaks.

- ii. Upload the file to HDFS.

```
hdfs dfs -put txt.log /tmp
```

- iii. Load data to the table.

```
load data inpath '/tmp/txt.log' into table user_info partition  
(year='2011');
```

4. Query the imported data.

```
select * from user_info;
```

5. Delete the user information table.

```
drop table user_info;
```

6. Run the following command to exit:

```
!q
```

----End

10.2 Configuring Hive Parameters

Navigation Path

Go to the Hive configurations page by referring to [Modifying Cluster Service Configuration Parameters](#).

Parameter Description

Table 10-2 Hive parameter description

Parameter	Description	Default Value
hive.auto.convert.join	Whether Hive converts common join to mapjoin based on the input file size. When Hive is used to query a join table, whatever the table size is (if the data in the join table is less than 24 MB, it is a small one), set this parameter to false . If this parameter is set to true , new mapjoin cannot be generated when you query a join table.	Possible values are as follows: <ul style="list-style-type: none"> • true • false The default value is true .
hive.default.fileformat	Indicates the default file format used by Hive.	Versions earlier than MRS 3.x: TextFile MRS 3.x or later: RCFile
hive.exec.reducers.max	Indicates the maximum number of reducers in a MapReduce job submitted by Hive.	999
hive.server2.thrift.max.worker.threads	Indicates the maximum number of threads that can be started in the HiveServer internal thread pool.	1,000
hive.server2.thrift.min.worker.threads	Indicates the number of threads started during initialization in the HiveServer internal thread pool.	5
hive.hbase.delete.mode.enabled	Indicates whether to enable the function of deleting HBase records from Hive. If this function is enabled, you can use remove table xx where xxx to delete HBase records from Hive. NOTE This parameter applies to MRS 3.x or later.	true

Parameter	Description	Default Value
hive.metastore.server.min.threads	Indicates the number of threads started by MetaStore for processing connections. If the number of threads is more than the set value, MetaStore always maintains a number of threads that is not lower than the set value, that is, the number of resident threads in the MetaStore thread pool is always higher than the set value.	200
hive.server2.enable.doAs	Indicates whether to simulate client users during sessions between HiveServer2 and other services (such as Yarn and HDFS). If you change the configuration item from false to true , users with only the column permission lose the permissions to access corresponding tables. NOTE This parameter applies to MRS 3.x or later.	true

10.3 Hive SQL

Hive SQL supports all features of Hive-3.1.0. For details, see <https://cwiki.apache.org/confluence/display/hive/languagemanual>.

Table 10-3 describes the extended Hive statements provided by .

Table 10-3 Extended Hive statements

Extended Syntax	Syntax Description	Syntax Example	Example Description
<pre>CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_ name (col_name data_type [COMMENT col_comment], ...) [ROW FORMAT row_format] [STORED AS file_format] STORED BY 'storage.handler.cl ass.name' [WITH SERDEPROPERTIE S (...)] [TBLPROPERTIES ("groupId"=" group1 ","locatorId"="loc ator1")] ...;</pre>	<p>The statement is used to create a Hive table and specify locators on which table data files locate. For details, see Using HDFS Colocation to Store Hive Tables.</p>	<pre>CREATE TABLE tab1 (id INT, name STRING) row format delimited fields terminated by '\t' stored as RCFILE TBLPROPERTIES(" groupId"=" group1 ","locatorId"="loc ator1");</pre>	<p>The statement is used to create table tab1 and specify locator1 on which the table data of tab1 locates.</p>

Extended Syntax	Syntax Description	Syntax Example	Example Description
<pre>CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_name (col_name data_type [COMMENT col_comment], ...) [ROW FORMAT row_format] [STORED AS file_format] STORED BY 'storage.handler.class.name' [WITH SERDEPROPERTIES (...)] ... [TBLPROPERTIES ('column.encode.columns'='col_name1,col_name2' 'column.encode.indices'='col_id1,col_id2', 'column.encode.classname'='encode_classname')]...;</pre>	<p>The statement is used to create a hive table and specify the table encryption column and encryption algorithm. For details, see Using the Hive Column Encryption Function.</p>	<pre>create table encode_test(id INT, name STRING, phone STRING, address STRING) ROW FORMAT SERDE 'org.apache.hadoop p.hive.serde2.lazy. LazySimpleSerDe' WITH SERDEPROPERTIES S ('column.encode.i ndices'='2,3', 'column.encode.cl assname'='org.apa che.hadoop.hive.s erde2.SMS4Rewrit er') STORED AS TEXTFILE;</pre>	<p>The statement is used to create table encode_test and specify that column 2 and column 3 will be encrypted using the org.apache.hadoop.hive.serde2.SMS4Rewriter encryption algorithm class during data insertion.</p>
<pre>REMOVE TABLE hbase_tablename [WHERE where_condition];</pre>	<p>The statement is used to delete data that meets criteria from the Hive on HBase table. For details, see Deleting Single-Row Records from Hive on HBase.</p>	<pre>remove table hbase_table1 where id = 1;</pre>	<p>The statement is used to delete data that meets the criterion of "id = 1" from the table.</p>

Extended Syntax	Syntax Description	Syntax Example	Example Description
<pre>CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_ name (col_name data_type [COMMENT col_comment], ...) [ROW FORMAT row_format] STORED AS inputformat 'org.apache.hado op.hive.contrib.fil eformat.Specifie dDelimiterInput- Format' outputformat 'org.apache.hadoo p.hive ql.io.HiveIg noreKeyTextOutpu tFormat';</pre>	<p>The statement is used to create a hive table and specify that the table supports customized row delimiters. For details, see Customizing Row Separators.</p>	<pre>create table blu(time string, num string, msg string) row format delimited fields terminated by ',' stored as inputformat 'org.apache.hado op.hive.contrib.fil eformat.Specifie dDelimiterInput- Format' outputformat 'org.apache.hadoo p.hive ql.io.HiveIg noreKeyTextOutpu tFormat';</pre>	<p>The statement is used to create table blu and set inputformat to SpecifiedDelimiterInputFormat so that the query row delimiter can be specified during the query.</p>

10.4 Permission Management

10.4.1 Hive Permission

Hive is a data warehouse framework built on Hadoop. It provides basic data analysis services using the Hive query language (HQL), a language like the structured query language (SQL).

MRS supports users, user groups, and roles. Permissions must be assigned to roles and then roles are bound to users or user groups. Users can obtain permissions only by binding a role or joining a group that is bound with a role. For details about Hive authorization, visit <https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Authorization>.

NOTE

- Hive permissions in security mode need to be managed whereas those in normal mode do not.
- MRS 3.x or later supports Ranger. If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. For details, see [Adding a Ranger Access Permission Policy for Hive](#).

Hive Permission Model

To use the Hive component, users must have permissions on Hive databases and tables (including external tables and views). In MRS, the complete Hive permission model is composed of Hive metadata permission and HDFS file permission. The Hive permission model also includes the permission to use databases or tables.

- Hive metadata permission

Similar to traditional relational databases, the Hive database of MRS supports the **CREATE** and **SELECT** permission, and the Hive tables and columns support the **SELECT**, **INSERT**, and **DELETE** permissions. Hive also supports the permissions of **OWNERSHIP** and **Hive Admin Privilege**.

NOTE

The **UPDATE** and **DELETE** operations on Hive tables and columns can be performed only when **ACID** is enabled. **ACID** cannot be enabled in the current version.

- Hive data file permission, also known as HDFS file permission

Hive database and table files are stored in the HDFS. The created databases or tables are saved in the **/user/hive/warehouse** directory of the HDFS by default. The system automatically creates subdirectories named after database names and database table names. To access a database or a table, the corresponding file permissions (read, write, and execute) on the HDFS are required.

NOTE

MRS 3.X supports multiple Hive instances. In the multi-instance scenario, the directory is **/user/hiven n ($n=1-4$)/warehouse**.

To perform various operations on Hive databases or tables, you need to associate the metadata permission with the HDFS file permission. For example, to query Hive data tables, you need to associate the metadata permission **SELECT** and the HDFS file permissions **Read** and **Write**.

To use the role management function of Manager GUI to manage the permissions of Hive databases and tables, you only need to configure the metadata permission, and the system will automatically associate and configure the HDFS file permission. In this way, operations on the interface are simplified, and the efficiency is improved.

Hive Users

MRS provides users and roles to use Hive, such as creating tables, inserting data into tables, and querying tables. Hive defines the **USER** class, corresponding to user instances. Hive defines the **GROUP** class, corresponding to role instances.

You can use Manager to set permissions for Hive users. This method only supports permission setting in roles. A user or user group can obtain the permissions only after a role is bound to the user or user group. Hive users can be granted administrator permissions and permissions to access databases, tables, and columns.

Hive Usage Scenarios and Related Permissions

Creating a database with Hive requires users to join in the **hive** group, without granting a role. Users have all permissions on the databases or tables created by

themselves in Hive or HDFS. They can create tables, select, delete, insert, or update data, and grant permissions to other users to allow them to access the tables and corresponding HDFS directories and files.

A user can access the tables or database only with permissions. The permission required by users varies according to Hive usage scenarios.

Table 10-4 Hive usage scenarios

Typical Scenario	Permission
Using Hive tables, columns, or databases	<p>Permissions required in different scenarios are as follows:</p> <ul style="list-style-type: none"> • To create tables, the CREATE permission is required. • To query data, the SELECT permission is required. • To insert data, the INSERT permission is required. • To delete data, the DELETE permission is required.
Associating and using other components	<p>In addition to Hive permissions, permissions of other components are required in some scenarios, for example:</p> <ul style="list-style-type: none"> • Yarn permissions are required when some HQL statements, such as insert, count, distinct, group by, order by, sort by, and join, are run. You are advised to grant Yarn permissions to the role of each Hive user. • HBase permission is required when Hive over HBase is used, for example, querying HBase table data in Hive.

In some special Hive usage scenarios, you need to configure other types of permission.

Table 10-5 Hive authorization precautions

Scenario	Permission
<p>Creating Hive databases, tables, and external tables, or adding partitions to created Hive tables or external tables when data files specified by Hive users are saved to other HDFS directories except /user/hive/warehouse</p>	<p>The directory must already exist, the Hive user must be the owner of the directory, and the Hive user must have the read, write, and execute permissions on the directory. The user must have the read and write permissions of all the upper-layer directories of the directory. After a MRS cluster administrator grants the Hive permission to the role, the HDFS permission is automatically granted.</p>
<p>Using load to load data from all the files or specified files in a specified directory to Hive tables as a Hive user</p>	<ul style="list-style-type: none"> • The data source is a Linux local disk, the specified directory exists, and the system user omm has read and execute permission of the directory and all its upper-layer directories. The specified file exists, and user omm has read permission of the file and has the read and execute permission of all the upper-layer directories of the file. • The data source is HDFS, the specified directory exists, and the Hive user is the owner of the directory and has read, write, and execute permission on the directory and its subdirectories, and has read and write permission on all its upper-layer directories. The specified file exists, and the Hive user is the owner of the file and has read, write, and execute permission, and has read and execute permission on the file and all its upper-layer directories. <p>NOTE When load is used to import data to a Linux local disk, files must be loaded to the HiveServer on which the command is run and the permission must be modified. You are advised to run the command on a client. The HiveServer to which the client is connected can be found. For example, if the Hive client displays 0: jdbc:hive2://10.172.0.43:21066/>, the IP address of the connected HiveServer is 10.172.0.43.</p>
<p>Creating or deleting functions or modifying any database</p>	<p>The Hive Admin Privilege is required.</p>

Scenario	Permission
Performing operations on all databases and tables in Hive	The user must be added to the supergroup user group and granted Hive Admin Privilege .

10.4.2 Creating a Hive Role

Scenario

This section describes how to create and configure a Hive role on Manager as the MRS cluster administrator. The Hive role can be granted the permissions of the Hive administrator and the permissions to operate Hive table data.

Creating a database with Hive requires users to join in the **hive** group, without granting a role. Users have all permissions on the databases or tables created by themselves in Hive or HDFS. They can create tables, select, delete, insert, or update data, and grant permissions to other users to allow them to access the tables and corresponding HDFS directories and files. The created databases or tables are saved in the **/user/hive/warehouse** directory of the HDFS by default.

NOTE

- A Hive role can be created only in security mode.
- If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. For details, see [Adding a Ranger Access Permission Policy for Hive](#) for MRS 3.x or later that supports Ranger.

Prerequisites

- You have understood the service requirements.
- Log in to FusionInsight Manager.
- The Hive client has been installed.

Procedure

For versions earlier than MRS 3.x, perform the following operations to create a Hive role:

Step 1 Log in to MRS Manager.

Step 2 Choose **System > Permission > Manage Role**.

Step 3 Click **Create Role**, and set **Role Name** and **Description**.

Step 4 Set permissions. For details, see [Table 10-6](#).

- **Hive Admin Privilege:** Hive administrator permissions. If you want to use this permission, run the **set role admin** command to set the permission before running SQL statements.
- **Hive Read Write Privileges:** Hive data table management permission, which is the operation permission to set and manage the data of created tables.

Select the permissions of a database as required. To specify permissions on tables, click the database name and select the permissions of the tables.

 **NOTE**

- Hive role management supports granting of the administrator rights and the rights to access databases, tables, and views.
- The permissions of the Hive administrator do not include the permission to manage HDFS.
- If there are too many tables in the database or too many files in tables, the permission granting may last a while. For example, if a table contains 10,000 files, the permission granting lasts about 2 minutes.

Table 10-6 Setting a role

Scenario	Role Authorization
Setting the Hive administrator permission	<p>In the Permission table, click Hive and select Hive Admin Privilege.</p> <p>NOTE After being bound to the Hive administrator role, perform the following operations during each maintenance operation:</p> <ol style="list-style-type: none"> 1. Log in to the node where the client is installed. For details, see Installing a Client. 2. Run the following command to configure environment variables: For example, if the Hive client installation directory is /opt/hiveclient, run source /opt/hiveclient/bigdata_env. 3. Run the following command to authenticate the user: kinit Hive service user 4. Run the following command to log in to the client tool: beeline 5. Run the following command to update the administrator permissions: set role admin;
Setting the permission to query a table of another user in the default database	<ol style="list-style-type: none"> 1. In the Permission table, choose Hive > Hive Read Write Privileges. 2. In the Permission column of the specified table, select SELECT.
Setting the permission to query a table of another user in the default database	<ol style="list-style-type: none"> 1. In the Permission table, choose Hive > Hive Read Write Privileges. 2. In the Permission column of the specified table, select Insert.

Scenario	Role Authorization
Setting the permission to import data to a table of another user in the default database	<ol style="list-style-type: none"> 1. In the Permission table, choose Hive > Hive Read Write Privileges. 2. In the Permission column of the specified table, select Delete and Insert.
Setting the permission to submit HQL commands to Yarn for execution	<p>The HQL commands used by some services are converted into MapReduce tasks and submitted to Yarn for execution. You need to set the Yarn permissions. For example, the HQL statements to be run use statements, such as insert, count, distinct, group by, order by, sort by, or join.</p> <ol style="list-style-type: none"> 1. In the Permission table, choose Yarn > Scheduler Queue > root. 2. In the Permission column of the default queue, select Submit.

Step 5 Click **OK**, and return to the **Role** page.

Step 6 Choose **System > Manage User > Create User**.

Step 7 Enter the username, set **User Type** to **Human-machine**, set the user password, add a user group bound with the Hive administrator role, bind the new Hive role to the user group, and click **OK**.

Step 8 After the user is created, you can run the SQL statement using the user.

----End

For MRS 3.x or later, perform the following operations to create a Hive role:

Step 1 Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#)

Step 2 Choose **System > Permission > Role**.

Step 3 Click **Create Role**, and set **Role Name** and **Description**.

Step 4 Set **Configure Resource Permission**. For details, see [Table 10-7](#).

- **Hive Admin Privilege:** Hive administrator permission.
- **Hive Read Write Privileges:** Hive data table management permission, which is the operation permission to set and manage the data of created tables.

 NOTE

- Hive role management supports the administrator permission, and the permissions of accessing tables and views, without granting the database permission.
- The permissions of the Hive administrator do not include the permission to manage HDFS.
- If there are too many tables in the database or too many files in tables, the permission granting may last a while. For example, if a table contains 10,000 files, the permission granting lasts about 2 minutes.

Table 10-7 Setting a role

Task	Role Authorization
Setting the Hive administrator permission	<p>In the Configure Resource Permission table, choose <i>Name of the desired cluster</i> > Hive and select Hive Admin Privilege.</p> <p>NOTE After being bound to the Hive administrator role, perform the following operations during each maintenance operation:</p> <ol style="list-style-type: none"> 1. Log in to the node where the Hive client is installed as the client installation user. 2. Run the following command to configure environment variables: For example, if the Hive client installation directory is /opt/hiveclient, run source /opt/hiveclient/bigdata_env. 3. Run the following command to authenticate the user: kinit Hive service user 4. Run the following command to log in to the client tool: beeline 5. Run the following command to update the administrator permissions: set role admin;
Setting the permission to query a table of another user in the default database	<ol style="list-style-type: none"> 1. In the Configure Resource Permission table, choose <i>Name of the desired cluster</i> > Hive > Hive Read Write Privileges. 2. Click the name of the specified database in the database list. Tables in the database are displayed. 3. In the Rights column of the specified table, choose Select.

Task	Role Authorization
Setting the permission to query a table of another user in the default database	<ol style="list-style-type: none"> 1. In the Configure Resource Permission table, choose <i>Name of the desired cluster</i> > Hive > Hive Read Write Privileges. 2. Click the name of the specified database in the database list. Tables in the database are displayed. 3. In the Permission column of the specified table, select INSERT.
Setting the permission to import data to a table of another user in the default database	<ol style="list-style-type: none"> 1. In the Configure Resource Permission table, choose <i>Name of the desired cluster</i> > Hive > Hive Read Write Privileges. 2. Click the name of the specified database in the database list. Tables in the database are displayed. 3. In the Permission column of the specified indexes, select DELETE and INSERT.
Setting the permission to submit HQL commands to Yarn for execution	<p>The HQL commands used by some services are converted into MapReduce tasks and submitted to Yarn for execution. You need to set the Yarn permissions. For example, the HQL statements to be run use statements, such as insert, count, distinct, group by, order by, sort by, or join.</p> <ol style="list-style-type: none"> 1. In the Permission table, choose <i>Name of the desired cluster</i> > Yarn > Scheduling Queue > root. 2. In the Permission column of the default queue, select Submit.

Step 5 Click **OK**, and return to the **Role** page.

----End

10.4.3 Configuring Permissions for Hive Tables, Columns, or Databases

Scenario

You can configure related permissions if you need to access tables or databases created by other users. Hive supports column-based permission control. If a user

needs to access some columns in tables created by other users, the user must be granted the permission for columns. The following describes how to grant table, column, and database permissions to users by using the role management function of MRS Manager.

 **NOTE**

- You can configure permissions for Hive tables, columns, or databases only in security mode.
- MRS 3.x or later supports Ranger. If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. For details, see [Adding a Ranger Access Permission Policy for Hive](#).

Prerequisites

- You have obtained a user account with the MRS cluster administrator permissions, such as **admin**.
- You have created a role, for example, **hrole**, on Manager by referring to instructions in [Creating a Hive Role](#). You do not need to set the Hive permission but need to set the permission to submit the HQL command to Yarn for execution.
- You have created two Hive human-machine users, such as **huser1** and **huser2**, on Manager and added them to the **hive** group. **huser2** has been bound to **hrole**. The **hdb** database has created by user **huser1** and the **htable** table has been created in the database.

Procedure

- Granting Table Permissions
Users have complete permission on the tables created by themselves in Hive and the HDFS. To access the tables created by others, they need to be granted the permission. After the Hive metadata permission is granted, the HDFS permission is automatically granted. The procedure for granting a role the permission of querying, inserting, and deleting **htable** data is as follows:
For versions earlier than MRS 3.x, perform the following operations to grant table permissions:
 - a. On MRS Manager, choose **System > Permission > Manage Role**.
 - b. Locate the row that contains **hrole**, and click **Modify**.
 - c. Choose **Hive > Hive Read Write Privileges**.
 - d. Click the name of the specified database **hdb** in the database list. Table **htable** in the database is displayed.
 - e. In the **Permission** column of the **htable** table, select **Select, Insert, and Delete**.
 - f. Click **OK**.For MRS 3.x or later, perform the following operations to grant table permissions:
 - a. On FusionInsight Manager, choose **System > Permission > Role**.
 - b. Locate the row that contains **hrole**, and click **Modify**.
 - c. Choose *Name of the desired cluster* > **Hive > Hive Read Write Privileges**.

- d. Click the name of the specified database **hdb** in the database list. Table **htable** in the database is displayed.
- e. In the **Permission** column of the **htable** table, select **SELECT**, **INSERT**, and **DELETE**.
- f. Click **OK**.

 **NOTE**

In role management, the procedure for granting a role the permission of querying, inserting, and deleting Hive external table data is the same. After the metadata permission is granted, the HDFS permission is automatically granted.

- **Granting Column Permissions**

Users have all permissions for the tables created by themselves in Hive and HDFS. Users do not have the permission to access the tables created by others. If a user needs to access some columns in tables created by other users, the user must be granted the permission for columns. After the Hive metadata permission is granted, the HDFS permission is automatically granted. The procedure for granting a role the permission of querying and inserting data in **hcol** of **htable** is as follows:

For versions earlier than MRS 3.x, perform the following operations to grant column permissions:

- a. On MRS Manager, choose **System > Permission > Manage Role**.
- b. Locate the row that contains **hrole**, and click **Modify**.
- c. Choose **Hive > Hive Read Write Privileges**.
- d. In the database list, click the specified database **hdb** to display the **htable** table in the database. Click the **htable** table to display the **hcol** column in the table.
- e. In the **Permission** column of the **hcol** column, select **Select** and **Insert**.
- f. Click **OK**.

For MRS 3.x or later, perform the following operations:

- a. On FusionInsight Manager, choose **System > Permission > Role**.
- b. Locate the row that contains **hrole**, and click **Modify**.
- c. Choose *Name of the desired cluster* > **Hive > Hive Read Write Privileges**.
- d. In the database list, click the specified database **hdb** to display the **htable** table in the database. Click the **htable** table to display the **hcol** column in the table.
- e. In the **Permission** column of the **hcol** column, select **SELECT** and **INSERT**.
- f. Click **OK**.

 **NOTE**

In role management, after the metadata permission is granted, the HDFS permission is automatically granted. Therefore, after the column permission is granted, the HDFS ACL permission for all files of the table is automatically granted.

- **Granting Database Permissions**

Users have complete permission on the databases created by themselves in Hive and the HDFS. To access the databases created by others, they need to

be granted the permission. After the Hive metadata permission is granted, the HDFS permission is automatically granted. The procedure for granting a role the permission of querying data and creating tables in database **hdb** is as follows. Other types of database operation permission are not supported.

For versions earlier than MRS 3.x, perform the following database authorization operations:

- a. On MRS Manager, choose **System > Permission > Manage Role**.
- b. Locate the row that contains **hrole**, and click **Modify**.
- c. Choose **Hive > Hive Read Write Privileges**.
- d. In the **Permission** column of the **hdb** database, select **Select** and **Create**.
- e. Click **OK**.

For MRS 3.x or later, perform the following operations to grant database permissions:

- a. On FusionInsight Manager, choose **System > Permission > Role**.
- b. Locate the row that contains **hrole**, and click **Modify**.
- c. Choose *Name of the desired cluster* > **Hive > Hive Read Write Privileges**.
- d. In the **Permission** column of the **hdb** database, select **SELECT** and **CREATE**.
- e. Click **OK**.

 **NOTE**

- Any permission for a table in the database is automatically associated with the HDFS permission for the database directory to facilitate permission management. When any permission for a table is canceled, the system does not automatically cancel the HDFS permission for the database directory to ensure performance. In this case, users can only log in to the database and view table names.
- When the query permission on a database is added to or deleted from a role, the query permission on tables in the database is automatically added to or deleted from the role.

Concepts

Table 10-8 Scenarios of using Hive tables, columns, or databases

Scenario	Required Permission
DESCRIBE TABLE	SELECT
SHOW PARTITIONS	SELECT
ANALYZE TABLE	SELECT and INSERT
SHOW COLUMNS	SELECT
SHOW TABLE STATUS	SELECT
SHOW TABLE PROPERTIES	SELECT
SELECT	SELECT
EXPLAIN	SELECT

Scenario	Required Permission
CREATE VIEW	SELECT, Grant Of Select, and CREATE
SHOW CREATE TABLE	SELECT and Grant Of Select
CREATE TABLE	CREATE
ALTER TABLE ADD PARTITION	INSERT
INSERT	INSERT
INSERT OVERWRITE	INSERT and DELETE
LOAD	INSERT and DELETE
ALTER TABLE DROP PARTITION	DELETE
CREATE FUNCTION	Hive Admin Privilege
DROP FUNCTION	Hive Admin Privilege
ALTER DATABASE	Hive Admin Privilege

10.4.4 Configuring Permissions to Use Other Components for Hive

Scenario

Hive may need to be associated with other components. For example, Yarn permissions are required in the scenario of using HQL statements to trigger MapReduce jobs, and HBase permissions are required in the Hive over HBase scenario. The following describes the operations in the two scenarios.

NOTE

- In security mode, Yarn and HBase permission management is enabled by default. Therefore, Yarn and HBase permissions need to be configured by default.
- In common mode, Yarn and HBase permission management is disabled by default. That is, any user has permissions. Therefore, YARN and HBase permissions does not need to be configured by default. If a user enables the permission management by modifying the Yarn or HBase configurations, the Yarn and HBase permissions then need to be configured.
- MRS 3.x or later supports Ranger. If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. For details, see [Adding a Ranger Access Permission Policy for Hive](#).

Prerequisites

- The Hive client has been installed. For example, the installation directory is `/opt/client`.
- You have obtained a user account with the MRS cluster administrator permissions, such as `admin`.

Procedure

Association with Yarn in MRS Earlier than 3.x

Yarn permissions are required when HQL statements, such as **insert**, **count**, **distinct**, **group by**, **order by**, **sort by**, and **join**, are used to trigger MapReduce jobs. The following uses the procedure for assigning a role the permissions to run the **count** statements in the **thc** table as an example.

- Step 1** Create a role on MRS Manager.
- Step 2** In the **Permission** table, choose **Yarn > Scheduler Queue > root**.
- Step 3** In the **Permission** column of the default queue, select **Submit** and click **OK**.
- Step 4** In the **Permission** table, choose **Hive > Hive Read Write Privileges > default**, select **Select** for **thc**, and click **OK**.

----End

Association with Yarn in MRS 3.x or Later

Yarn permissions are required when HQL statements, such as **insert**, **count**, **distinct**, **group by**, **order by**, **sort by**, and **join**, are used to trigger MapReduce jobs. The following uses the procedure for assigning a role the permissions to run the **count** statements in the **thc** table as an example.

- Step 1** Create a role on FusionInsight Manager.
- Step 2** In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **Yarn > Scheduler Queue > root**.
- Step 3** In the **Permission** column of the **default** queue, select **Submit** and click **OK**.
- Step 4** In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **Hive > Hive Read Write Privileges > default**. Select **SELECT** for table **thc**, and click **OK**.

----End

Hive over HBase Authorization in MRS Earlier than 3.x

After the permissions are assigned, you can use HQL statements that are similar to SQL statements to access HBase tables from Hive. The following uses the procedure for assigning a user the rights to query HBase tables as an example.

- Step 1** On the role management page of MRS Manager, create an HBase role, for example, **hive_hbase_create**, and grant the permission to create HBase tables.

In the **Permission** table, choose **HBase > HBase Scope > global**, select **create** of the namespace **default**, and click **OK**.
- Step 2** On MRS Manager, create a human-machine user, for example, **hbase_creates_user**, add the user to the **hive** group, and bind the **hive_hbase_create** role to the user so that the user can create Hive and HBase tables.
- Step 3** Log in to the node where the client is installed. For details, see [Installing a Client](#).
- Step 4** Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 5 Run the following command to authenticate the user:

```
kinit hbase_creates_user
```

Step 6 Run the following command to go to the shell environment of the Hive client:

```
beeline
```

Step 7 Run the following command to create a table in Hive and HBase, for example, the **thh** table.

```
CREATE TABLE thh(id int, name string, country string) STORED BY  
'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH  
SERDEPROPERTIES("hbase.columns.mapping" = "cf1:id,cf1:name,:key")  
TBLPROPERTIES ("hbase.table.name" = "thh");
```

The created Hive table and the HBase table are stored in the Hive database **default** and the HBase namespace **default**, respectively.

Step 8 On the role management page of MRS Manager, create a role, for example, **hive_hbase_select**, and assign the role the permission to query the Hive table **thh** and the HBase table **thh**.

1. In the **Permission** table, choose **HBase > HBase Scope > global > default**, select **Read** for the **thh** table, and click **OK** to grant the HBase role the permission to query the table.
2. Edit a role. In the **Permission** table, choose **HBase > HBase Scope > global > hbase**. Select **Execute** for **hbase:meta**, and click **OK**.
3. Edit a role. In the **Permission** table, choose **Hive > Hive Read Write Privileges > default**, select **Select** for **thh**, and click **OK**.

Step 9 On MRS Manager, create a human-machine user, for example, **hbase_select_user**, add the user to the **hive** group, and bind the **hive_hbase_select** role to the user so that the user can query Hive and HBase tables.

Step 10 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 11 Run the following command to authenticate users:

```
kinit hbase_select_user
```

Step 12 Run the following command to go to the shell environment of the Hive client:

```
beeline
```

Step 13 Run the following command to use an HQL statement to query HBase table data:

```
select * from thh;
```

```
----End
```

Hive over HBase Authorization in MRS 3.x or Later

After the permissions are assigned, you can use HQL statements that are similar to SQL statements to access HBase tables from Hive. The following uses the procedure for assigning a user the rights to query HBase tables as an example.

Step 1 On the role management page of FusionInsight Manager, create an HBase role, for example, **hive_hbase_create**, and grant the permission to create HBase tables.

In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **HBase** > **HBase Scope** > **global**. Select **Create** of the namespace **default**, and click **OK**.

Step 2 On FusionInsight Manager, create a human-machine user, for example, **hbase_creates_user**, add the user to the **hive** group, and bind the **hive_hbase_create** role to the user so that the user can create Hive and HBase tables.

Step 3 Log in to the node where the client is installed as the client installation user.

Step 4 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 5 Run the following command to authenticate the user:

```
kinit hbase_creates_user
```

Step 6 Run the following command to go to the shell environment of the Hive client:

```
beeline
```

Step 7 Run the following command to create a table in Hive and HBase, for example, the **thh** table.

```
CREATE TABLE thh(id int, name string, country string) STORED BY  
'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH  
SERDEPROPERTIES("hbase.columns.mapping" = "cf1:id,cf1:name,:key")  
TBLPROPERTIES ("hbase.table.name" = "thh");
```

The created Hive table and the HBase table are stored in the Hive database **default** and the HBase namespace **default**, respectively.

Step 8 On the role management page of FusionInsight Manager, create a role, for example, **hive_hbase_select**, and assign the role the permission to query the Hive table **thh** and the HBase table **thh**.

1. In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **HBase** > **HBase Scope** > **global** > **default**. Select **read** of the **thh** table, and click **OK** to grant the table query permission to the HBase role.
2. Edit the role. In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **HBase** > **HBase Scope** > **global** > **hbase**, select **Execute** for **hbase:meta**, and click **OK**.
3. Edit the role. In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **Hive** > **Hive Read Write Privileges** > **default**. Select **SELECT** for the **thh** table, and click **OK**.

Step 9 On FusionInsight Manager, create a human-machine user, for example, **hbase_select_user**, add the user to the **hive** group, and bind the **hive_hbase_select** role to the user so that the user can query Hive and HBase tables.

Step 10 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 11 Run the following command to authenticate users:

```
kinit hbase_select_user
```

Step 12 Run the following command to go to the shell environment of the Hive client:

```
beeline
```

Step 13 Run the following command to use an HQL statement to query HBase table data:

```
select * from thh;  
  
----End
```

10.5 Using a Hive Client

Scenario

This section guides users to use a Hive client in an O&M or service scenario.

Prerequisites

- The client has been installed. For example, the client is installed in the **/opt/hadoopclient** directory. The client directory in the following operations is only an example. Change it to the actual installation directory.
- Service component users are created by the MRS cluster administrator as required. In security mode, machine-machine users need to download the keytab file. A human-machine user must change the password upon the first login.

Using the Hive Client (Versions Earlier Than MRS 3.x)

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 Log in to the Hive client based on the cluster authentication mode.

- In security mode, run the following command to complete user authentication and log in to the Hive client:

```
kinit Component service user  
beeline
```

- In common mode, run the following command to log in to the Hive client. If no component service user is specified, the current OS user is used to log in to the Hive client.

```
beeline -n component service user
```

 NOTE

After a beeline connection is established, you can compile and submit HQL statements to execute related tasks. To run the Catalog client command, you need to run the **!q** command first to exit the beeline environment.

Step 5 Run the following command to execute the HCatalog client command:

```
hcat -e "cmd"
```

cmd must be a Hive DDL statement, for example, **hcat -e "show tables"**.

 NOTE

- To use the HCatalog client, choose **More > Download Client** on the service page to download the clients of all services. This restriction does not apply to the beeline client.
- Due to permission model incompatibility, tables created using the HCatalog client cannot be accessed on the HiveServer client. However, the tables can be accessed on the WebHCat client.
- If you use the HCatalog client in Normal mode, the system performs DDL commands using the current user who has logged in to the operating system.
- Exit the beeline client by running the **!q** command instead of by pressing **Ctrl + c**. Otherwise, the temporary files generated by the connection cannot be deleted and a large number of junk files will be generated as a result.
- If multiple statements need to be entered during the use of beeline clients, separate the statements from each other using semicolons (;) and set the value of **entireLineAsCommand** to **false**.

Setting method: If beeline has not been started, run the **beeline --entireLineAsCommand=false** command. If the beeline has been started, run the **!set entireLineAsCommand false** command.

After the setting, if a statement contains semicolons (;) that do not indicate the end of the statement, escape characters must be added, for example, **select concat_ws('\;', collect_set(col1)) from tbl**.

----End

Using the Hive Client (MRS 3.x or Later)

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 MRS 3.X supports multiple Hive instances. If you use the client to connect to a specific Hive instance in a scenario when multiple Hive instances are installed, run the following command to load the environment variables of the instance. Otherwise, skip this step. For example, load the environment variables of the Hive2 instance.

```
source Hive2/component_env
```

Step 5 Log in to the Hive client based on the cluster authentication mode.

- In security mode, run the following command to complete user authentication and log in to the Hive client:

kinit *Component service user*

beeline

- In common mode, run the following command to log in to the Hive client. If no component service user is specified, the current OS user is used to log in to the Hive client.

beeline -n *component service user*

Step 6 Run the following command to execute the HCatalog client command:

hcat -e "*cmd*"

cmd must be a Hive DDL statement, for example, **hcat -e "show tables"**.

 **NOTE**

- To use the HCatalog client, choose **More > Download Client** on the service page to download the clients of all services. This restriction does not apply to the beeline client.
- Due to permission model incompatibility, tables created using the HCatalog client cannot be accessed on the HiveServer client. However, the tables can be accessed on the WebHCat client.
- If you use the HCatalog client in Normal mode, the system performs DDL commands using the current user who has logged in to the operating system.
- Exit the beeline client by running the **!q** command instead of by pressing **Ctrl + C**. Otherwise, the temporary files generated by the connection cannot be deleted and a large number of junk files will be generated as a result.
- If multiple statements need to be entered during the use of beeline clients, separate the statements from each other using semicolons (;) and set the value of **entireLineAsCommand** to **false**.

Setting method: If beeline has not been started, run the **beeline --entireLineAsCommand=false** command. If the beeline has been started, run the **!set entireLineAsCommand false** command.

After the setting, if a statement contains semicolons (;) that do not indicate the end of the statement, escape characters must be added, for example, **select concat_ws('\;', collect_set(col1)) from tbl.**

----End

Common Hive Client Commands

The following table lists common Hive Beeline commands.

For more commands, see <https://cwiki.apache.org/confluence/display/Hive/HiveServer2+Clients#HiveServer2Clients-BeelineCommands>.

Table 10-9 Common Hive Beeline commands

Command	Description
set <key>=<value>	Sets the value of a specific configuration variable (key). NOTE If the variable name is incorrectly spelled, the Beeline does not display an error.

Command	Description
set	Prints the list of configuration variables overwritten by users or Hive.
set -v	Prints all configuration variables of Hadoop and Hive.
add FILE[S] <filepath> <filepath>*add JAR[S] <filepath> <filepath>*add ARCHIVE[S] <filepath> <filepath>*	Adds one or more files, JAR files, or ARCHIVE files to the resource list of the distributed cache.
add FILE[S] <ivyurl> <ivyurl>* add JAR[S] <ivyurl> <ivyurl>* add ARCHIVE[S] <ivyurl> <ivyurl>*	Adds one or more files, JAR files, or ARCHIVE files to the resource list of the distributed cache using the Ivy URL in the ivy://goup:module:version?query_string format.
list FILE[S]list JAR[S]list ARCHIVE[S]	Lists the resources that have been added to the distributed cache.
list FILE[S] <filepath>*list JAR[S] <filepath>*list ARCHIVE[S] <filepath>*	Checks whether given resources have been added to the distributed cache.
delete FILE[S] <filepath>*delete JAR[S] <filepath>*delete ARCHIVE[S] <filepath>*	Deletes resources from the distributed cache.
delete FILE[S] <ivyurl> <ivyurl>* delete JAR[S] <ivyurl> <ivyurl>* delete ARCHIVE[S] <ivyurl> <ivyurl>*	Delete the resource added using <ivyurl> from the distributed cache.
reload	Enable HiveServer2 to discover the change of the JAR file hive.reloadable.aux.jars.path in the specified path. (You do not need to restart HiveServer2.) Change actions include adding, deleting, or updating JAR files.
dfs <dfs command>	Runs the dfs command.
<query string>	Executes the Hive query and prints the result to the standard output.

10.6 Using HDFS Colocation to Store Hive Tables

Scenario

HDFS Colocation is the data location control function provided by HDFS. The HDFS Colocation API stores associated data or data on which associated operations are performed on the same storage node. Hive supports the HDFS Colocation function. When Hive tables are created, after the locator information is set for table files, data files of related tables are stored on the same storage node when data is inserted into tables using the insert statement (other data import modes are not supported). This ensures convenient and efficient data computing among associated tables. The supported table formats are only TextFile and RCFile.

NOTE

This section applies to MRS 3.x or later.

Procedure

Step 1 Log in to the node where the client is installed as a client installation user.

Step 2 Run the following command to switch to the client installation directory, for example, **opt/client**:

```
cd /opt/client
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 If the cluster is in security mode, run the following command to authenticate the user:

```
kinit MRS username
```

Step 5 Create the *groupid* through the HDFS API.

```
hdfs colocationadmin -createGroup -groupId <groupid> -locatorIds  
<locatorid1>,<locatorid2>,<locatorid3>
```

NOTE

In the preceding command, *<groupid>* indicates the name of the created group. The group created in this example contains three locators. You can define the number of locators as required.

For details about group ID creation and HDFS Colocation, see HDFS description.

Step 6 Run the following command to log in to the Hive client:

```
beeline
```

Step 7 Enable Hive to use colocation.

Assume that **table_name1** and **table_name2** are associated with each other. Run the following statements to create them:

```
CREATE TABLE <[db_name.]table_name1>[(col_name data_type , ...)] [ROW  
FORMAT <row_format>] [STORED AS <file_format>]  
TBLPROPERTIES("groupId"=" <group> ","locatorId"=" <locator1>");
```

```
CREATE TABLE <[db_name.]table_name2> [(col_name data_type , ...)] [ROW  
FORMAT <row_format>] [STORED AS <file_format>]  
TBLPROPERTIES("groupId"=" <group> ","locatorId"=" <locator1>");
```

After data is inserted into **table_name1** and **table_name2** using the insert statement, data files of **table_name1** and **table_name2** are distributed to the same storage position in the HDFS, facilitating associated operations among the two tables.

----End

10.7 Using the Hive Column Encryption Function

Scenario

Hive supports encryption of one or more columns in a table. When creating a Hive table, you can specify the columns to be encrypted and encryption algorithm. When data is inserted into the table using the insert statement, the related columns are encrypted. Column encryption can be performed in HDFS tables of only the TextFile and SequenceFile file formats. Hive column encryption does not support the view and Hive over HBase scenarios.

Hive supports two column encryption algorithms, which can be specified during table creation:

- AES (the encryption class is org.apache.hadoop.hive.serde2.AESRewriter)
- SMS4 (the encryption class is org.apache.hadoop.hive.serde2.SMS4Rewriter)

NOTE

When you import data from a common Hive table into a Hive column encryption table, you are advised to delete the original data from the common Hive table as long as doing this does not affect other services. The reason is that retaining an unencrypted table is a security risk.

Operation Procedure

- Step 1** Specify the column to be encrypted and encryption algorithm when creating a table.

```
create table <[db_name.]table_name> (<col_name1>  
<data_type> ,<col_name2> <data_type> ,<col_name3>  
<data_type> ,<col_name4> <data_type>) ROW FORMAT SERDE  
'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe' WITH  
SERDEPROPERTIES ('column.encode.columns'='<col_name2>,<col_name3>',  
'column.encode.classname'='org.apache.hadoop.hive.serde2.AESRewriter')STO  
RED AS TEXTFILE;
```

Alternatively, use the following statement:

```
create table <[db_name.]table_name> (<col_name1>  
<data_type> ,<col_name2> <data_type> ,<col_name3>
```

```
<data_type>,<col_name4> <data_type>) ROW FORMAT SERDE  
'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe' WITH  
SERDEPROPERTIES ('column.encode.indices'='1,2',  
'column.encode.classname'='org.apache.hadoop.hive.serde2.SMS4Rewriter')  
STORED AS TEXTFILE;
```

 NOTE

- The numbers used to specify encryption columns start from 0. 0 indicates column 1, 1 indicates column 2, and so on.
- When creating a table with encrypted columns, ensure that the directory where the table resides is empty.

Step 2 Insert data into the table using the insert statement.

Assume that the test table exists and contains data.

```
insert into table <table_name> select <col_list> from test;
```

----End

10.8 Customizing Row Separators

Scenario

In most cases, a carriage return character is used as the row delimiter in Hive tables stored in text files, that is, the carriage return character is used as the terminator of a row during queries. However, some data files are delimited by special characters, and not a carriage return character.

MRS Hive allows you to use different characters or character combinations to delimit rows of Hive text data. When creating a table, set **inputformat** to **SpecifiedDelimiterInputFormat**, and set the following parameter before search each time. Then the table data is queried by the specified delimiter.

```
set hive.textinput.record.delimiter="";
```

 NOTE

- The Hue component of the current version does not support the configuration of multiple separators when files are imported to a Hive table.
- This section applies to MRS 3.x or later.

Procedure

Step 1 Specify **inputFormat** and **outputFormat** when creating a table.

```
CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS]  
[db_name.]table_name [(col_name data_type [COMMENT col_comment], ...)]  
[ROW FORMAT row_format] STORED AS inputformat  
'org.apache.hadoop.hive.contrib.fileformat.SpecifiedDelimiterInputFormat'  
outputformat 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
```

Step 2 Specify the delimiter before search.

```
set hive.textinput.record.delimiter='!@!'
```


Hive will use '!@!' as the row delimiter.

----End

10.9 Deleting Single-Row Records from Hive on HBase

Scenario

Due to the limitations of underlying storage systems, Hive does not support the ability to delete a single piece of table data. In Hive on HBase, MRS Hive supports the ability to delete a single piece of HBase table data. Using a specific syntax, Hive can delete one or more pieces of data from an HBase table.

Table 10-10 Permissions required for deleting single-row records from the Hive on HBase table

Cluster Authentication Mode	Required Permission
Security mode	SELECT, INSERT, and DELETE
Common mode	None

Procedure

Step 1 To delete some data from an HBase table, run the following HQL statement:

```
remove table <table_name> where <expression>;
```

In the preceding information, *<expression>* specifies the filter condition of the data to be deleted. *<table_name>* indicates the Hive on HBase table from which data is to be deleted.

----End

10.10 Configuring HTTPS/HTTP-based REST APIs

Scenario

WebHCat provides external REST APIs for Hive. By default, the open-source community version uses the HTTP protocol.

MRS Hive supports the HTTPS protocol that is more secure, and enables switchover between the HTTP protocol and the HTTPS protocol.

NOTE

The security mode supports HTTPS and HTTP, and the common mode supports only HTTP.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Modify the Hive configuration.

- For versions earlier than MRS 3.x: Enter the parameter name in the search box, search for **templeton.protocol.type**, change the parameter value to **HTTPS** or **HTTP**, and restart the Hive service to use the corresponding protocol.
- For MRS 3.x or earlier: Choose **WebHCat > Security**. On the page that is displayed, select **HTTPS** or **HTTP**. After the modification, restart the Hive service to use the corresponding protocol.

----End

10.11 Enabling or Disabling the Transform Function

Scenario

The Transform function is not allowed by Hive of the open source version.

MRS Hive supports the configuration of the Transform function. The function is disabled by default, which is the same as that of the open-source community version.

Users can modify configurations of the Transform function to enable the function. However, security risks exist when the Transform function is enabled.

 **NOTE**

The Transform function can be disabled only in security mode.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Enter the parameter name in the search box, search for **hive.security.transform.disallow**, change the parameter value to **true** or **false**, and restart all HiveServer instances.

 **NOTE**

- If this parameter is set to **true**, the Transform function is disabled, which is the same as that in the open-source community version.
- If this parameter is set to **false**, the Transform function is enabled, which poses security risks.

----End

10.12 Access Control of a Dynamic Table View on Hive

Scenario

This section describes how to create a view on Hive when MRS is configured in security mode, authorize access permissions to different users, and specify that different users access different data.

In the view, Hive can obtain the built-in function **current_user()** of the users who submit tasks on the client and filter the users. This way, authorized users can only access specific data in the view.

 **NOTE**

In normal mode, the **current_user()** function cannot distinguish users who submit tasks on the client. Therefore, the access control function takes effect only for Hive in security mode. If the **current_user()** function is used in the actual service logic, the possible risks must be fully evaluated during the conversion between the security mode and normal mode.

Operation Example

- If the **current_user** function is not used, different views need to be created for different users to access different data.
 - Authorize the view **v1** permission to user **hiveuser1**. The user **hiveuser1** can access data with **type** set to **hiveuser1** in **table1**.
create view v1 as select * from table1 where type='hiveuser1'
 - Authorize the view **v2** permission to user **hiveuser2**. The user **hiveuser2** can access data with **type** set to **hiveuser2** in **table1**.
create view v2 as select * from table1 where type='hiveuser2'
- If the **current_user** function is used, only one view needs to be created. Authorize the view **v** permission to users **hiveuser1** and **hiveuser2**. When user **hiveuser1** queries view **v**, the **current_user()** function is automatically converted to **hiveuser1**. When user **hiveuser2** queries view **v**, the **current_user()** function is automatically converted to **hiveuser2**.

```
create view v as select * from table1 where type=current_user()
```

10.13 Specifying Whether the ADMIN Permissions Is Required for Creating Temporary Functions

Scenario

You must have **ADMIN** permission when creating temporary functions on Hive of the open source community version.

MRS Hive supports the configuration of the function for creating temporary functions with **ADMIN** permission. The function is disabled by default, which is the same as that of the open-source community version.

You can modify configurations of this function. After the function is enabled, you can create temporary functions without **ADMIN** permission. If this parameter is set to **false**, security risks exist.

NOTE

The security mode supports the configuration of whether the ADMIN permission is required for creating temporary functions, but the common mode does not support this function.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Enter the parameter name in the search box, search for **hive.security.temporary.function.need.admin**, change the parameter value to **true** or **false**, and restart all HiveServer instances.

NOTE

- If this parameter is set to **true**, the ADMIN permission is required for creating temporary functions, which is the same as that in the open source community.
- If this parameter is set to **false**, the ADMIN permission is not required for creating temporary functions.

----End

10.14 Using Hive to Read Data in a Relational Database

Scenario

Hive allows users to create external tables to associate with other relational databases. External tables read data from associated relational databases and support Join operations with other tables in Hive.

Currently, the following relational databases can use Hive to read data:

- DB2
- Oracle

NOTE

This section applies to MRS 3.x or later.

Prerequisites

The Hive client has been installed.

Procedure

Step 1 Log in to the node where the Hive client is installed as the Hive client installation user .

Step 2 Run the following command to go to the client installation directory:

```
cd Client installation directory
```

For example, if the client installation directory is **/opt/client**, run the following command:

```
cd /opt/client
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 Check whether the cluster authentication mode is Security.

- If yes, run the following command to authenticate the user:

```
kinit Hive service user
```
- If no, go to [Step 5](#).

Step 5 Run the following command to upload the driver JAR package of the relational database to be associated to an HDFS directory.

```
hdfs dfs -put directory where the JAR package is located HDFS directory to which the JAR is uploaded
```

For example, to upload the Oracle driver JAR package in **/opt** to the **/tmp** directory in HDFS, run the following command:

```
hdfs dfs -put /opt/ojdbc6.jar /tmp
```

Step 6 Create an external table on the Hive client to associate with the relational database, as shown in the following example.

 **NOTE**

If the security mode is used, the user who creates the table must have the **ADMIN** permission. The **ADD JAR** path is subject to the actual path.

```
-- Example of associating with an Oracle Linux 6 database
-- In security mode, set the admin permission.
set role admin;
-- Upload the driver JAR package of the relational database to be associated. The driver JAR packages
vary according to databases.
ADD JAR hdfs:///tmp/ojdbc6.jar;

CREATE EXTERNAL TABLE ora_test
-- The Hive table must have one more column than the database return result. This column is used for
paging query.
(id STRING,rownum string)
STORED BY 'com.qubitproducts.hive.storage.jdbc.JdbcStorageHandler'
TBLPROPERTIES (
-- Relational database table type
"qubit.sql.database.type" = "ORACLE",
-- Connect to the URL of the relational database through JDBC. (The URL formats vary according to
databases.)
"qubit.sql.jdbc.url" = "jdbc:oracle:thin:@//10.163.0.1:1521/mydb",
-- Relational database driver class type
"qubit.sql.jdbc.driver" = "oracle.jdbc.OracleDriver",
-- SQL statement queried in the relational database. The result is returned to the Hive table.
"qubit.sql.query" = "select name from aaa",
-- (Optional) Match the Hive table columns to the relational database table columns.
"qubit.sql.column.mapping" = "id=name",
-- Relational database user
"qubit.sql.dbcp.username" = "test",
-- Relational database password
"qubit.sql.dbcp.password" = "123456");
```

----End

10.15 Supporting Traditional Relational Database Syntax in Hive

Overview

Hive supports the following types of traditional relational database syntax:

- Grouping
- EXCEPT and INTERSECT

Grouping

Syntax description:

- Grouping takes effect only when the Group by statement contains ROLLUP or CUBE.
- The result set generated by CUBE contains all the combinations of values in the selected columns.
- The result set generated by ROLLUP contains the combinations of a certain layer structure in the selected columns.

- Grouping: If a row is added by using the CUBE or ROLLUP operator, the output value of the added row is 1. If the row is not added by using the CUBE or ROLLUP operator, the output value of the added row is 0.

For example, the **table_test** table exists in Hive and the table structure is as follows:

```
+-----+-----+
| table_test.id | table_test.value |
+-----+-----+
| 1             | 10                |
| 1             | 15                |
| 2             | 20                |
| 2             | 5                 |
| 2             | 13                |
+-----+-----+
```

Run the following statement:

```
select id,grouping(id),sum(value) from table_test group by id with rollup;
```

The result is as follows:

```
+-----+-----+-----+
| id | groupingresult | sum |
+-----+-----+-----+
| 1  | 0              | 25  |
| NULL | 1              | 63  |
| 2  | 0              | 38  |
+-----+-----+-----+
```

EXCEPT and INTERSECT

Syntax description:

- EXCEPT returns the difference of two result sets (that is, non-duplicated values return only one query).
- INTERSECT returns the intersection of two result sets (that is, non-duplicated values return by both queries).

For example, two tables **test_table1** and **test_table2** exist in Hive.

The table structure of **test_table1** is as follows:

```
+-----+
| test_table1.id |
+-----+
| 1              |
| 2              |
| 3              |
| 4              |
+-----+
```

The table structure of **test_table2** is as follows:

```
+-----+
| test_table2.id |
+-----+
| 2              |
| 3              |
| 4              |
| 5              |
+-----+
```

- Run the following EXCEPT statement:

```
select id from test_table1 except select id from test_table2;
```

The result is as follows:

```
+-----+--+
|_alias_0.id |
+-----+--+
| 1          |
+-----+--+
```

- Run the following INTERSECT statement:

```
select id from test_table1 intersect select id from test_table2;
```

The result is as follows:

```
+-----+--+
|_alias_0.id |
+-----+--+
| 2          |
| 3          |
| 4          |
+-----+--+
```

10.16 Switching Between Multiple Hive Instances in Hive Editor

Scenario

The Hive Editor (implemented by Hue) on the Hive web page of FusionInsight MRS can adapt to multiple Hive instances. You can perform operations on Hive tables and data in different instances.

NOTE

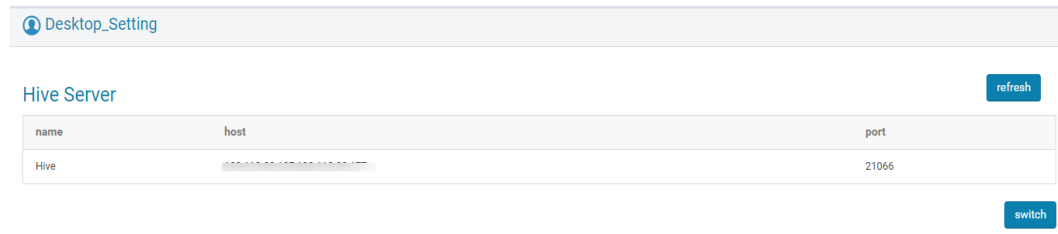
This section applies to MRS 3.x or later.

Prerequisites

- Multiple Hive instances have been installed.
- The Hue component has been installed.

Procedure

- Step 1** Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **Hue**. On the **Dashboard** page, click **Hue (XXX, Active)** on the right of **Hue WebUI** to go to the Hue home page.
- Step 2** In the lower left corner of the Hue page, choose *Username* > **Desktop Setting**. The **Desktop Setting** page is displayed.
- Step 3** Click **switch** in the **Hive Server** table to open the multi-instance list.
- Step 4** Select the required Hive instance and click **save** to save the configuration.
- Step 5** Click **refresh** in the upper right corner of the **Desktop Setting** page to refresh the page and view the instance after the switchover.



name	host	port
Hive	...	21066

----End

10.17 Creating User-Defined Hive Functions

When built-in functions of Hive cannot meet requirements, you can compile user-defined functions (UDFs) and use them for query.

According to implementation methods, UDFs are classified as follows:

- Common UDFs: used to perform operations on a single data row and export a single data row.
- User-defined aggregating functions (UDAFs): used to input multiple data rows and export a single data row.
- User-defined table-generating functions (UDTFs): used to perform operations on a single data row and export multiple data rows.

According to use methods, UDFs are classified as follows:

- Temporary functions: used only in the current session and must be recreated after a session restarts.
- Permanent functions: used in multiple sessions. You do not need to create them every time a session restarts.

NOTE

You need to properly control the memory and thread usage of variables in UDFs. Improper control may cause memory overflow or high CPU usage.

The following uses AddDoublesUDF as an example to describe how to compile and use UDFs.

Function

AddDoublesUDF is used to add two or more floating point numbers. In this example, you can learn how to write and use UDFs.

NOTE

- A common UDF must be inherited from **org.apache.hadoop.hive.ql.exec.UDF**.
- A common UDF must implement at least one **evaluate()**. The evaluate function supports overloading.
- To develop a customized function, you need to add the **hive-exec-3.1.0.jar** dependency package to the project. The package can be obtained from the Hive installation directory.

Sample Code

The following is a UDF code example:

```
package com.xxx.bigdata.hive.example.udf;
import org.apache.hadoop.hive.ql.exec.UDF;

public class AddDoublesUDF extends UDF {
    public Double evaluate(Double... a) {
        Double total = 0.0;
        // Processing logic
        for (int i = 0; i < a.length; i++)
            if (a[i] != null)
                total += a[i];
        return total;
    }
}
```

How to Use

Step 1 Packing programs as **AddDoublesUDF.jar** on the client node, and upload the package to a specified directory in HDFS, for example, **/user/hive_examples_jars**.

Both the user who creates the function and the user who uses the function must have the read permission on the file.

The following are example statements:

```
hdfs dfs -put ./hive_examples_jars /user/hive_examples_jars
```

```
hdfs dfs -chmod 777 /user/hive_examples_jars
```

Step 2 Check the cluster authentication mode.

- In security mode, log in to the beeline client as a user with the Hive management permission and run the following commands:

```
kinit Hive service user
```

```
beeline
```

```
set role admin;
```

- In common mode, run the following command:

```
beeline -n Hive service user
```

Step 3 Define the function in HiveServer. Run the following SQL statement to create a permanent function:

```
CREATE FUNCTION addDoubles AS  
'com.xxx.bigdata.hive.example.udf.AddDoublesUDF' using jar 'hdfs://hacluster/  
user/hive_examples_jars/AddDoublesUDF.jar';
```

addDoubles indicates the function alias that is used for SELECT query.

Run the following statement to create a temporary function:

```
CREATE TEMPORARY FUNCTION addDoubles AS  
'com.xxx.bigdata.hive.example.udf.AddDoublesUDF' using jar 'hdfs://hacluster/  
user/hive_examples_jars/AddDoublesUDF.jar';
```

- *addDoubles* indicates the function alias that is used for SELECT query.
- **TEMPORARY** indicates that the function is used only in the current session with the HiveServer.

Step 4 Run the following SQL statement to use the function on the HiveServer:

```
SELECT addDoubles(1,2,3);
```

 **NOTE**

If an [Error 10011] error is displayed when you log in to the client again, run the **reload function;** command and then use this function.

Step 5 Run the following SQL statement to delete the function from the HiveServer:

```
DROP FUNCTION addDoubles;
```

```
----End
```

Extended Applications

None

10.18 Enhancing beeline Reliability

Scenario

- When the beeline client is disconnected due to network exceptions during the execution of a batch processing task, tasks submitted before beeline is disconnected can be properly executed in Hive. When you start the batch processing task again, the submitted tasks are not executed and tasks that are not executed are executed in sequence.
- When the HiveServer service breaks down due to some reasons during the execution of a batch processing task, Hive enables that the tasks that have been successfully executed are not executed again when the same batch processing task is started again. The execution starts from the task that has not been executed from the time when HiveServer2 breaks down.

 **NOTE**

This section applies to MRS 3.x or later.

Example

1. Beeline is reconnected after being disconnection.

Example:

```
beeline -e "${SQL}" --hivevar batchid=xxxxx
```

2. Beeline kills the running tasks.

Example:

```
beeline -e "" --hivevar batchid=xxxxx --hivevar kill=true
```

3. Log in to the beeline client and start the mechanism of reconnection after disconnection.

Log in to the beeline client and run the **set hivevar:batchid=xxxxx** command.

 NOTE

Instructions:

- `xxxx` indicates the batch ID of tasks submitted in the same batch using the beeline client. Batch IDs can be used to identify the task submission batch. If the batch ID is not contained when a task is submitted, this feature is not enabled.
- If the running SQL script depends on the data timeliness, you are advised not to enable the breakpoint reconnection mechanism. You can use a new batch ID to submit tasks. During reexecution of the scripts, some SQL statements have been executed and are not executed again. As a result, expired data is obtained.
- If some built-in time functions are used in the SQL script, it is recommended that you do not enable the breakpoint reconnection mechanism or the use of a new batch ID for each execution. The reason is the same as above.
- A SQL script contains one or more subtasks. If the logic for deleting and creating temporary tables exist in the SQL script, it is recommended that the logic for deleting temporary tables be placed at the end of the script. If the subtasks executed after the temporary table deletion task fail to be executed and the temporary table is used in the subtasks before the temporary table deletion task, when the SQL script is executed using the same batch ID for the next time, the compilation of the subtasks (excluding the task for creating the temporary table because the creation has been completed and is not executed again, and only compilation is allowed) executed before the temporary table deletion task fails because the temporary has been deleted. In this case, you are advised to use a new batch ID to execute the script.

Parameter description:

- **zk.cleanup.finished.job.interval**: indicates the interval for executing the cleanup task. The default interval is 60 seconds.
- **zk.cleanup.finished.job.outdated.threshold**: indicates the threshold of the node validity period. A node is generated for tasks in the same batch. The threshold is calculated from the end time of the execution of the current batch task. If the time exceeds 60 minutes, the node is deleted.
- **batch.job.max.retry.count**: indicates the maximum number of retry times of a batch task. If the number of retry times of a batch task exceeds the value of this parameter, the task execution record is deleted. The task will be executed from the first task when the task is started next time. The default value is 10.
- **beeline.reconnect.zk.path**: indicates the root node for storing task execution progress. The default value for the Hive service is `/beeline`.

10.19 Viewing Table Structures Using the show create Statement as Users with the select Permission

Scenario

This function is applicable to Hive and Spark2x in MRS 3.x and later.

With this function enabled, if the select permission is granted to a user during Hive table creation, the user can run the **show create table** command to view the table structure.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Choose **HiveServer(Role) > Customization**, add a customized parameter to the **hive-site.xml** parameter file, set **Name** to **hive.allow.show.create.table.in.select.nogrant**, and set **Value** to **true**. Restart all Hive instances after the modification.

Step 3 Determine whether to enable this function on the Spark/Spark2x client.

- If yes, download and install the Spark/Spark2x client again.
- If no, no further action is required.

----End

10.20 Writing a Directory into Hive with the Old Data Removed to the Recycle Bin

Scenario

This function applies to Hive.

After this function is enabled, run the following command to write a directory into Hive: **insert overwrite directory "/path1"** After the operation is successfully performed, the old data is removed to the recycle bin, and the directory cannot be an existing database path in the Hive metastore.

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

- Step 2** Choose **HiveServer(Role) > Customization**, add a customized parameter to the **hive-site.xml** parameter file, set **Name** to **hive.override.directory.move.trash**, and set **Value** to **true**. Restart all Hive instances after the modification.

----End

10.21 Inserting Data to a Directory That Does Not Exist

Scenario

This function applies to Hive.

With this function enabled, run the **insert overwrite directory /path1/path2/path3...** command to write a subdirectory. The permission of the **/path1/path2** directory is 700, and the owner is the current user. If the **/path3** directory does not exist, it is automatically created and data is written successfully.

This function is supported when **hive.server2.enable.doAs** is set to **true** in earlier versions. This version supports the function when **hive.server2.enable.doAs** is set to **false**.

NOTE

The parameter adjustment of this function is the same as that of the custom parameters added in [Writing a Directory into Hive with the Old Data Removed to the Recycle Bin](#).

Procedure

- Step 1** The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

NOTE

- If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)
- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

- Step 2** Choose **HiveServer(Role) > Customization**, add a customized parameter to the **hive-site.xml** parameter file, set **Name** to **hive.override.directory.move.trash**, and set **Value** to **true**. Restart all Hive instances after the modification.

----End

10.22 Creating Databases and Creating Tables in the Default Database Only as the Hive Administrator

Scenario

This function is applicable to Hive and Spark2x for MRS 3.x or later, or Hive and Spark for versions earlier than MRS 3.x.

After this function is enabled, only the Hive administrator can create databases and tables in the default database. Other users can use the databases only after being authorized by the Hive administrator.

NOTE

- After this function is enabled, common users are not allowed to create a database or create a table in the default database. Based on the actual application scenario, determine whether to enable this function.
- Permissions of common users are restricted. In the scenario where common users have been used to perform operations, such as database creation, table script migration, and metadata recreation in an earlier version of database, the users can perform such operations on the database in the condition that this function is disabled temporarily after the database is migrated or after the cluster is upgraded.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Choose **HiveServer(Role) > Customization**, add a customized parameter to the **hive-site.xml** parameter file, set **Name** to **hive.allow.only.admin.create**, and set **Value** to **true**. Restart all Hive instances after the modification.

Step 3 Determine whether to enable this function on the Spark/Spark2x client.

- If yes, download and install the Spark/Spark2x client again.
- If no, no further action is required.

----End

10.23 Disabling of Specifying the location Keyword When Creating an Internal Hive Table

Scenario

This function is applicable to Hive and Spark2x for MRS 3.x or later, or Hive and Spark for versions earlier than MRS 3.x.

After this function is enabled, the **location** keyword cannot be specified when a Hive internal table is created. Specifically, after a table is created, the table path following the location keyword is created in the default **/warehouse** directory and cannot be specified to another directory. If the location is specified when the internal table is created, the creation fails.

NOTE

After this function is enabled, the location keyword cannot be specified during the creation of a Hive internal table. The table creation statement is restricted. If a table that has been created in the database is not stored in the default directory **/warehouse**, the **location** keyword can still be specified when the database creation, table script migration, or metadata recreation operation is performed by disabling this function temporarily.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Choose **HiveServer(Role) > Customization**, add a customized parameter to the **hive-site.xml** parameter file, set **Name** to **hive.internaltable.notallowlocation**, and set **Value** to **true**. Restart all Hive instances after the modification.

Step 3 Determine whether to enable this function on the Spark/Spark2x client.

- If yes, download and install the Spark/Spark2x client again.
- If no, no further action is required.

----End

10.24 Enabling the Function of Creating a Foreign Table in a Directory That Can Only Be Read

Scenario

This function is applicable to Hive and Spark2x for MRS 3.x or later, or Hive and Spark for versions earlier than MRS 3.x.

After this function is enabled, the user or user group that has the read and execute permissions on a directory can create foreign tables in the directory without checking whether the current user is the owner of the directory. In addition, the directory of a foreign table cannot be stored in the default directory `\warehouse`. In addition, do not change the permission of the directory during foreign table authorization.

NOTE

After this function is enabled, the function of the foreign table changes greatly. Based on the actual application scenario, determine whether to enable this function.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Choose **HiveServer(Role) > Customization**, add a customized parameter to the `hive-site.xml` parameter file, set **Name** to `hive.restrict.create.grant.external.table`, and set **Value** to **true**.

Step 3 Choose **MetaStore(Role) > Customization**, add a customized parameter to the `hivemetastore-site.xml` parameter file, set **Name** to `hive.restrict.create.grant.external.table`, and set **Value** to **true**. Restart all Hive instances after the modification.

Step 4 Determine whether to enable this function on the Spark/Spark2x client.

- If yes, download and install the Spark/Spark2x client again.
- If no, no further action is required.

----End

10.25 Authorizing Over 32 Roles in Hive

Scenario

This function applies to Hive.

The number of OS user groups is limited, and the number of roles that can be created in Hive cannot exceed 32. After this function is enabled, more than 32 roles can be created in Hive.

NOTE

- After this function is enabled and the table or database is authorized, roles that have the same permission on the table or database will be combined using vertical bars (|). When the ACL permission is queried, the combined result is displayed, which is different from that before the function is enabled. This operation is irreversible. Determine whether to make adjustment based on the actual application scenario.
- MRS 3.x and later versions support Ranger. If the current component uses Ranger for permission control, you need to configure related policies based on Ranger for permission management. For details, see [Adding a Ranger Access Permission Policy for Hive](#).
- After this function is enabled, a maximum of 512 roles (including **owner**) are supported by default. The number is controlled by the user-defined parameter **hive.supports.roles.max** of MetaStore. You can change the value based on the actual application scenario.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Choose **MetaStore(Role) > Customization**, add a customized parameter to the **hivemetastore-site.xml** parameter file, set **Name** to **hive.supports.over.32.roles**, and set **Value** to **true**. Restart all Hive instances after the modification.

----End

10.26 Restricting the Maximum Number of Maps for Hive Tasks

Scenario

- This function applies to Hive.
- This function is used to limit the maximum number of maps for Hive tasks on the server to avoid performance deterioration caused by overload of the HiveServer service.

Procedure

Step 1 The Hive service configuration page is displayed.

- For versions earlier than MRS 3.x, click the cluster name. On the cluster details page that is displayed, choose **Components > Hive > Service Configuration**, and select **All** from the **Basic** drop-down list.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). And choose **Cluster > Name of the desired cluster > Services > Hive > Configurations > All Configurations**.

Step 2 Choose **MetaStore(Role) > Customization**, add a customized parameter to the **hivemetastore-site.xml** parameter file, set **Name** to **hive.mapreduce.per.task.max.splits**, and set the parameter to a large value. Restart all Hive instances after the modification.

----End

10.27 HiveServer Lease Isolation

Scenario

- This function applies to Hive.
- This function can be enabled to specify specific users to access HiveServer services on specific nodes, achieving HiveServer resource isolation.

NOTE

This section applies to MRS 3.x or later.

Procedure

This section describes how to set lease isolation for user **hiveuser** for existing HiveServer instances.

- Step 1** Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
- Step 2** Choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive** > **HiveServer**.
- Step 3** In the HiveServer list, select the HiveServer for which lease isolation is configured and choose **HiveServer** > **Instance Configurations** > **All Configurations**.
- Step 4** In the upper right corner of the **All Configurations** page, search for **hive.server2.zookeeper.namespace** and specify its value, for example, **hiveserver2_zk**.
- Step 5** Click **Save**. In the dialog box that is displayed, click **OK**.
- Step 6** Choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive**, choose **More** > **Restart Service**, and enter the password to restart the service.
- Step 7** Run the **beeline -u** command to log in to the client and run the following command:

```
beeline -u
"jdbc:hive2://10.5.159.13:2181/?serviceDiscoveryMode=zooKeeper;zooKeeperNameSpace=hiveserver2_zk;ssl.qop=auth-conf;auth=KERBEROS;principal=hive/hadoop.<System domain name>@<System domain name>"
```

In the command, **10.5.159.13** is replaced with the IP address of any ZooKeeper instance, which can be viewed through **Cluster** > *Name of the desired cluster* > **Services** > **ZooKeeper** > **Instance**.

hiveserver2_zk following **zooKeeperNameSpace=** is set to the value of **hive.server2.zookeeper.namespace** in [Step 4](#).

As a result, only the HiveServer whose lease isolation is configured can be logged in.

 **NOTE**

- After this function is enabled, you must run the preceding command during login to access the HiveServer for which lease isolation is configured. If you run the **beeline** command to log in to the client, only the HiveServer that is not isolated by the lease is accessed.
- You can log in to FusionInsight Manager, choose **System** > **Permission** > **Domain and Mutual Trust**, and view the value of **Local Domain**, which is the current system domain name. **hive/hadoop.<system domain name>** is the username. All letters in the system domain name contained in the username are lowercase letters.

----End

10.28 Hive Supporting Transactions

Scenario

Hive supports transactions at the table and partition levels. When the transaction mode is enabled, transaction tables can be incrementally updated, deleted, and read, implementing atomicity, isolation, consistency, and durability of operations on transaction tables.

 **NOTE**

This section applies to MRS 3.x or later.

Introduction to Transaction Features

A transaction is a group of unitized operations. These operations are either executed together or not executed together. A transaction is an inseparable unit of work. The four basic elements of a transaction are usually called ACID features, which are as follows:

- **Atomicity:** A transaction is an inseparable unit of work. All operations in a transaction occur or do not occur together.
- **Consistency:** The database integrity constraints are not damaged before and after a transaction starts.
- **Isolation:** When multiple transactions are concurrently accessed, the transactions are isolated from each other. A transaction does not affect the running of other transactions. The impacts between transactions are as follows: dirty read, non-repeatable read, phantom read, and lost update.
- **Durability:** After a transaction is complete, changes made by the transaction lock to the database are permanently stored in the database.

Characteristics of transaction execution:

- A statement can be written to multiple partitions or tables. If the operation fails, the user cannot see partial write or insert. Even if data is frequently changed, operations can still be quickly performed.
- Hive can automatically compress ACID transaction files without affecting concurrent queries. When querying many small partition files, automatic compression can improve query performance and metadata occupation.
- Read semantics include snapshot isolation. When the read operation starts, the Hive data warehouse is logically locked. The read operation is not affected by any changes that occur during the operation.

Lock Mechanism

Transactions implement the ACID feature through the following two aspects:

- Write-ahead logging ensures atomicity and durability.
- Locking ensures isolation.

Operation	Type of Held Locks
Insert overwrite	If hive.txn.lock.iow is set to true , the exclusive lock is held. If hive.txn.lock.iow is set to false , the semi-shared lock is held.
Insert	Shared lock. When performing this operation, you can perform read and write operations on the current table or partition.

Operation	Type of Held Locks
Update/delete	Semi-shared lock. When this operation is performed, an operation of holding a shared lock can be performed, but an operation of holding an exclusive lock or a semi-shared lock cannot be performed.
Drop	Exclusive lock. You cannot perform any other operations on the current table or partition when performing this operation.

 NOTE

If a conflict caused by the lock mechanism exists in the write operation, the operation that preferentially holds the lock succeeds, and other operations fail.

Procedure

Starting a Transaction

Step 1 Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive** > **Configurations** > **All Configurations** > **MetaStore(Role)** > **Transaction**.

Step 2 Set `metastore.compactor.initiator.on` to `true`.

Step 3 Set `metastore.compactor.worker.threads` to a positive integer.

 NOTE

`metastore.compactor.worker.threads`: Specifies the number of working threads for running the compression program on MetaStore. Set this parameter based on the actual requirements. If the value is too small, the transaction compression task is executed slowly. If the value is too large, the MetaStore execution performance deteriorates.

Step 4 Log in to the Hive client and run the following command to enable the following parameters. For details, see [Using a Hive Client](#).

```
set hive.support.concurrency=true;
```

```
set hive.exec.dynamic.partition.mode=nonstrict;
```

```
set hive.txn.manager=org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
```

Create a transaction table.

Step 5 Run the following command to create a transaction table:

```
CREATE TABLE [IF NOT EXISTS] [db_name.]table_name (col_name data_type  
[COMMENT col_comment], ...) [ROW FORMAT row_format] STORED AS orc .....  
TBLPROPERTIES ('transactional'='true'[, 'groupId'='group1' ... ] );
```

For example:

```
CREATE TABLE acidTbl (a int, b int) STORED AS ORC TBLPROPERTIES  
('transactional'='true');
```

 NOTE

- Currently, the transactions support only the ORC format.
- External tables are not supported.
- Sorted tables are not supported.
- To create a transaction table, you must add the table attribute **transactional='true'**.
- The transaction table can be read and written only in transaction mode.

Use the transaction table.

Step 6 Run commands to use the transaction table. The following uses the **acidTbl** table as an example:

- Insert data into an existing transaction table:

INSERT INTO acidTbl VALUES(1,1);

- Update an existing transaction table:

UPDATE acidTbl SET b = 10 where a = 1;

The content of **acidTbl** is changed to:

```

+-----+-----+
| acidtbl.a | acidtbl.b |
+-----+-----+
| 1         | 10        |
+-----+-----+
1 row selected (0.775 seconds)
    
```

- Merge the old and new transaction tables:

The **acidTbl_update** table contains the following data:

```

+-----+-----+
| acidtbl_update.a | acidtbl_update.b |
+-----+-----+
| 1                 | 20                |
| 2                 | 10                |
+-----+-----+
2 rows selected (0.537 seconds)
    
```

MERGE INTO acidTbl AS a

USING acidTbl_update AS b ON a.a = b.a

WHEN MATCHED THEN UPDATE SET b = b. b

WHEN NOT MATCHED THEN INSERT VALUES (b.a, b.b);

The content of **acidTbl** is changed to:

```

+-----+-----+
| acidtbl.a | acidtbl.b |
+-----+-----+
| 1         | 20        |
| 2         | 10        |
+-----+-----+
2 rows selected (0.666 seconds)
    
```

 NOTE

If "Error evaluating cardinality_violation" is displayed when you run the **merge** command, check whether duplicate connection keys exist or run the **set hive.merge.cardinality.check=false** command to avoid this exception.

- Delete records from the transaction table.

DELETE FROM acidTbl where a = 2;

```

+-----+-----+
| acidtbl.a | acidtbl.b |
+-----+-----+
| 1         | 20        |
+-----+-----+
1 row selected (1.253 seconds)
    
```

Checking the Transaction Execution Status

Step 7 Run the following command to check the transaction execution status:

- Check the lock:
show locks;
 - Check the compression task:
show compactions;
 - Check the task execution status:
show transactions;
 - Interrupt a transaction:
abort transactions *TransactionId*;
- End

Configuring the Compression Function

HDFS does not support in-place file changing. For the new content, HDFS does not provide read consistency either. To provide these features on HDFS, we follow the standard approach used in other data warehouse tools: table or partition data is stored in a set of base files, and new, updated, as well as deleted records are stored in incremental files. Each transaction creates a new set of incremental files to change the table or partition. When read, the base files and the incremental files are merged and the changes of the update or deletion are applied.

Writing a transaction table generates some small files in HDFS. Hive provides major and minor compression policies for combining these small files.

Procedure of Automatic Compression

Step 1 Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive** > **Configurations** > **All Configurations** > **MetaStore(Role)** > **Transaction**.

Step 2 Set the following parameters as required:

Table 10-11 Parameter description

Parameter	Description
hive.compactor.check.interval	Interval of executing compression threads. Unit: second. Default value: 300
hive.compactor.cleaner.run.interval	Interval of executing cleaning threads. Unit: millisecond. Default value: 5,000 .
hive.compactor.delta.num.threshold	Threshold of the number of incremental files that trigger minor compression. Default value: 10

Parameter	Description
hive.compactor.delta.pct.threshold	Ratio threshold of the total size of incremental files (delta) that trigger Major compression to the size of base files. The value 0.1 indicates that Major compression is triggered when the ratio of the total size of delta files to the size of base files is 10%. Default value: 0.1
hive.compactor.max.num.delta	Maximum number of incremental files that the compressor will attempt to process in a single job. Default value: 500
metastore.compactor.initiator.on	Indicates whether to run the startup program thread and cleanup program thread on the MetaStore instance. The value must be true . Default value: false .
metastore.compactor.worker.threads	Number of compression program work threads running on MetaStore. If this parameter is set to 0 , no compression is performed. To use a transaction, you must set this parameter to a positive number on one or more instances of the MetaStore service. Unit: second Default value: 0

Step 3 Log in to the Hive client and perform compression. For details, see [Using a Hive Client](#).

```
CREATE TABLE table_name (
  id int, name string
)
CLUSTERED BY (id) INTO 2 BUCKETS STORED AS ORC
TBLPROPERTIES ("transactional"="true",
  "compactor.mapreduce.map.memory.mb"="2048",          -- Specify the properties of a compression
  "compactorthreshold.hive.compactor.delta.num.threshold"="4", -- If there are more than four incremental
  "compactorthreshold.hive.compactor.delta.pct.threshold"="0.5" -- If the ratio of the incremental file size to
  the basic file size is greater than 50%, deep compression is triggered.
);
```

or

```
ALTER TABLE table_name COMPACT 'minor' WITH OVERWRITE TBLPROPERTIES
("compactor.mapreduce.map.memory.mb"="3072"); -- Specify the properties of a compression map job.
ALTER TABLE table_name COMPACT 'major' WITH OVERWRITE TBLPROPERTIES
("tblprops.orc.compress.size"="8192"); -- Modify any other Hive table attributes.
```

 **NOTE**

After compression, small files are not deleted immediately. After the cleaner thread performs cleaning, the files are deleted in batches.

----**End**

10.29 Switching the Hive Execution Engine to Tez

Scenario

Hive can use the Tez engine to process data computing tasks. Before executing a task, you can manually switch the execution engine to Tez.

Prerequisites

The TimelineServer role of the Yarn service has been installed in the cluster and is running properly.

Switching the Execution Engine on the Client to Tez

Step 1 Install and log in to the Hive client. For details, see [Using a Hive Client](#).

Step 2 Run the following commands to switch the engine and enable the `yarn.timeline-service.enabled` parameter:

```
set hive.execution.engine=tez;  
set yarn.timeline-service.enabled=true;
```

NOTE

- After `yarn.timeline-service.enabled` is enabled, you can view the details about the tasks executed by the Tez engine on TezUI. After this function is enabled, task information will be reported to TimelineServer. If the TimelineServer instance is faulty, the task will fail.
- Tez uses the ApplicationMaster buffer pool. Therefore, `yarn.timeline-service.enabled` must be enabled before Tez tasks are submitted. Otherwise, this parameter cannot take effect and you need to log in to the client again to configure it.
- When the execution engine needs to be switched to another engine, you need to run the `set yarn.timeline-service.enabled=false` command on the client to disable the `yarn.timeline-service.enabled` parameter.
- To specify a Yarn running queue, run the `set tez.queue.name=default` command on the client.

Step 3 Submit and execute the Tez tasks.

Step 4 Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **Tez** > **TezUI** (*host name*) to view the task execution status on the TezUI page.

For versions earlier than MRS 3.x, log in to MRS Manager, choose **Services**, and click **Tez**. On the displayed page, click the link next to **Tez WebUI** to view the task execution status on the TezUI page.

----End

Switching the Default Execution Engine of Hive to Tez

Step 1 Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* >

Services > Hive > Configurations > All Configurations > HiveServer(Role), and search for **hive.execution.engine**.

For versions earlier than MRS 3.x, log in to MRS Manager, choose **Services**, and click **Hive**. On the displayed page, click the **Service Configuration** tab, select **All** from the **Type** drop-down list. On the navigation pane on the left, choose **HiveServer** and search for **hive.execution.engine**.

Step 2 Set **hive.execution.engine** to **tez**.

Step 3 Choose **Hive(Service) > Customization** and search for **yarn.site.customized.configs**.

Step 4 Add a customized parameter **yarn.timeline-service.enabled** next to **yarn.site.customized.configs** and set its value to **true**.

 **NOTE**

- After **yarn.timeline-service.enabled** is enabled, you can view the details about the tasks executed by the Tez engine on TezUI. After this function is enabled, task information will be reported to TimelineServer. If the TimelineServer instance is faulty, the task will fail.
- Tez uses the ApplicationMaster buffer pool. Therefore, **yarn.timeline-service.enabled** must be enabled before Tez tasks are submitted. Otherwise, this parameter cannot take effect and you need to log in to the client again to configure it.
- When the execution engine needs to be switched to another one, you need to set the value of parameter **yarn.timeline-service.enabled** to **false**.

Step 5 Click **Save**. In the displayed confirmation dialog box, click **OK**.

For versions earlier than MRS 3.x, click **Save Configuration** and click **Yes** in the displayed dialog box.

Step 6 Choose **Dashboard > More > Restart Service** to restart the Hive service. Enter the password to restart the service.

For versions earlier than MRS 3.x, Click the **Service Status** tab and choose **More > Restart Service** to restart the Hive service.

Step 7 Install and log in to the Hive client. For details, see [Using a Hive Client](#).

Step 8 Submit and execute the Tez tasks.

Step 9 Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Tez > TezUI (host name)**. On the displayed TezUI page, view the task execution status.

For versions earlier than MRS 3.x, log in to MRS Manager, choose **Services**, and click **Tez**. On the displayed page, click the link next to **Tez WebUI** to view the task execution status on the TezUI page.

----End

10.30 Hive Materialized View

Introduction

A Hive materialized view is a special table obtained based on the query results of Hive internal tables. A materialized view can be considered as an intermediate

table that stores actual data and occupies physical space. The tables on which a materialized view depends are called the base tables of the materialized view.

Materialized views are used to pre-compute and save the results of time-consuming operations such as table joining or aggregation. When executing a query, you can rewrite the query statement based on the base tables to the query statement based on materialized views. In this way, you do not need to perform time-consuming operations such as join and group by, thereby quickly obtaining the query result.

 **NOTE**

- A materialized view is a special table that stores actual data and occupies physical space.
- Before deleting a base table, you must delete the materialized view created based on the base table.
- The materialized view creation statement is atomic, which means that other users cannot see the materialized view until all query results are populated.
- A materialized view cannot be created based on the query results of another materialized view.
- A materialized view cannot be created based on the results of a tableless query.
- You cannot insert, update, delete, load, or merge materialized views.
- You can perform complex query operations on materialized views, because they are special tables in nature.
- When the data of a base table is updated, you need to manually update the materialized view. Otherwise, the materialized view will retain the old data. That is, the materialized view expires.
- You can use the describe syntax to check whether the materialized view created based on ACID tables has expired.
- The describe statement cannot be used to check whether a materialized view created based on non-ACID tables has expired.

Creating a Materialized View

Syntax

```
CREATE MATERIALIZED VIEW [IF NOT EXISTS] [db_name.]materialized_view_name
[COMMENT materialized_view_comment]
DISABLE REWRITE
[ROW FORMAT row_format]
[STORED AS file_format]
| STORED BY 'storage.handler.class.name' [WITH SERDEPROPERTIES (...)]
]
[LOCATION hdfs_path]
[TBLPROPERTIES (property_name=property_value, ...)]
AS
<query>;
```

 **NOTE**

- Currently, the following materialized view file formats are supported: PARQUET, TextFile, SequenceFile, RCfile, and ORC. If **STORED AS** is not specified in the creation statement, the default file format is ORC.
- Names of materialized views must be unique in the same database. Otherwise, you cannot create a new materialized view, and data files of the original materialized view will be overwritten by the data files queried based on the base table in the new one. As a result, data may be tampered with. (After being tampered with, the materialized view can be restored by re-creating the materialized view.)

Cases

Step 1 Log in to the Hive client and run the following command to enable the following parameters. For details, see [Using a Hive Client](#).

```
set hive.support.concurrency=true;
```

```
set hive.exec.dynamic.partition.mode=nonstrict;
```

```
set hive.txn.manager=org.apache.hadoop.hive.ql.lockmgr.DbTxnManager;
```

Step 2 Create a base table and insert data.

```
create table tb_emp(
empno int,ename string,job string,mgr int,hiredate TIMESTAMP,sal float,comm float,deptno int
)stored as orc
tblproperties('transactional'='true');

insert into tb_emp values(7369, 'SMITH', 'CLERK',7902, '1980-12-17 08:30:09',800.00,NULL,20),
(7499, 'ALLEN', 'SALESMAN',7698, '1981-02-20 17:12:00',1600.00,300.00,30),
(7521, 'WARD', 'SALESMAN',7698, '1981-02-22 09:05:34',1250.00,500.00,30),
(7566, 'JONES', 'MANAGER', 7839, '1981-04-02 10:14:13',2975.00,NULL,20),
(7654, 'MARTIN', 'SALESMAN',7698, '1981-09-28 08:36:17',1250.00,1400.00,30),
(7698, 'BLAKE', 'MANAGER',7839, '1981-05-01 11:12:55',2850.00,NULL,30),
(7782, 'CLARK', 'MANAGER',7839, '1981-06-09 15:45:28',2450.00,NULL,10),
(7788, 'SCOTT', 'ANALYST',7566, '1987-04-19 14:05:34',3000.00,NULL,20),
(7839, 'KING', 'PRESIDENT',NULL, '1981-11-17 10:18:25',5000.00,NULL,10),
(7844, 'TURNER', 'SALESMAN',7698, '1981-09-08 09:05:34',1500.00,0.00,30),
(7876, 'ADAMS', 'CLERK',7788, '1987-05-23 15:07:44',1100.00,NULL,20),
(7900, 'JAMES', 'CLERK',7698, '1981-12-03 16:23:56',950.00,NULL,30),
(7902, 'FORD', 'ANALYST',7566, '1981-12-03 08:48:17',3000.00,NULL,20),
(7934, 'MILLER', 'CLERK',7782, '1982-01-23 11:45:29',1300.00,NULL,10);
```

Step 3 Create a materialized view based on the results of the **tb_emp** query.

```
create materialized view group_mv disable rewrite
row format serde 'org.apache.hadoop.hive.serde2.JsonSerDe'
stored as textfile
tblproperties('mv_content'='Total compensation of each department')
as select deptno,sum(sal) sum_sal from tb_emp group by deptno;
```

----End

Applying a Materialized View

Rewrite the query statement based on base tables to the query statement based on materialized views to improve the query efficiency.

Cases

Execute the following query statement:

```
select deptno,sum(sal) from tb_emp group by deptno having sum(sal)>10000;
```

Based on the created materialized view, rewrite the query statement:

```
select deptno, sum_sal from group_mv where sum_sal>10000;
```

Checking a Materialized View

Syntax

```
SHOW MATERIALIZED VIEWS [IN database_name]
['identifier_with_wildcards'];
```

```
DESCRIBE [EXTENDED | FORMATTED] [db_name.]materialized_view_name;
```

Cases

```
show materialized views;  
describe formatted group_mv;
```

Deleting a Materialized View

Syntax

```
DROP MATERIALIZED VIEW [db_name.]materialized_view_name;
```

Cases

```
drop materialized view group_mv;
```

Rebuilding a Materialized View

When a materialized view is created, the base table data is filled in the materialized view. However, the data that is added, deleted, or modified in the base table is not automatically synchronized to the materialized view. Therefore, you need to manually rebuild the view after updating the data.

Syntax

```
ALTER MATERIALIZED VIEW [db_name.]materialized_view_name REBUILD;
```

Cases

```
alter materialized view group_mv rebuild;
```

NOTE

When the base table data is updated but the materialized view data is not updated, the materialized view is in the expired state by default.

The describe statement can be used to check whether a materialized view created based on transaction tables has expired. If the value of **Outdated for Rewriting** is **Yes**, the license has expired. If the value of **Outdated for Rewriting** is **No**, the license has not expired.

10.31 Hive Log Overview

Log Description

Log path: The default save path of Hive logs is `/var/log/Bigdata/hive/role name`, the default save path of Hive1 logs is `/var/log/Bigdata/hive1/role name`, and the others follow the same rule.

- HiveServer: `/var/log/Bigdata/hive/hiveserver` (run log) and `var/log/Bigdata/audit/hive/hiveserver` (audit log)
- MetaStore: `/var/log/Bigdata/hive/metastore` (run log) and `/var/log/Bigdata/audit/hive/metastore` (audit log)
- WebHCat: `/var/log/Bigdata/hive/webhcat` (run log) and `/var/log/Bigdata/audit/hive/webhcat` (audit log)

Log archive rule: The automatic compression and archiving function of Hive is enabled. By default, when the size of a log file exceeds 20 MB (which is

adjustable), the log file is automatically compressed. The naming rule of a compressed log file is as follows: *<Original log name>-<yyyy-mm-dd_hh-mm-ss>.[ID].log.zip* A maximum of 20 latest compressed files are reserved. The number of compressed files and compression threshold can be configured.

Table 10-12 Hive log list

Log Type	Log File Name	Description
Run log	/hiveserver/hiveserver.out	Log file that records HiveServer running environment information.
	/hiveserver/hive.log	Run log file of the HiveServer process.
	/hiveserver/hive-omm- <i><Date>-<PID>-gc.log.<No.></i>	GC log file of the HiveServer process.
	/hiveserver/prestartDetail.log	Work log file before the HiveServer startup.
	/hiveserver/check-serviceDetail.log	Log file that records whether the Hive service starts successfully
	/hiveserver/cleanupDetail.log	Cleanup log file about the HiveServer uninstallation
	/hiveserver/startDetail.log	Startup log file of the HiveServer process.
	/hiveserver/stopDetail.log	Shutdown log file of the HiveServer process.
	/hiveserver/localtasklog/omm- <i><Date>-<Task ID>.log</i>	Run log file of the local Hive task.
	/hiveserver/localtasklog/omm- <i><Date>-<Task ID>-gc.log.<No.></i>	GC log file of the local Hive task.
	/metastore/metastore.log	Run log file of the MetaStore process.
	/metastore/hive-omm- <i><Date>-<PID>-gc.log.<No.></i>	GC log file of the MetaStore process.
	/metastore/postinstallDetail.log	Work log file after the MetaStore installation.
	/metastore/prestartDetail.log	Work log file before the MetaStore startup

Log Type	Log File Name	Description
	/metastore/cleanupDetail.log	Cleanup log file of the MetaStore uninstallation
	/metastore/startDetail.log	Startup log file of the MetaStore process.
	/metastore/stopDetail.log	Shutdown log file of the MetaStore process.
	/metastore/metastore.out	Log file that records MetaStore running environment information.
	/webhcat/webhcat-console.out	Log file that records the normal start and stop of the WebHCat process.
	/webhcat/webhcat-console-error.out	Log file that records the start and stop exceptions of the WebHCat process.
	/webhcat/prestartDetail.log	Work log file before the WebHCat startup.
	/webhcat/cleanupDetail.log	Cleanup logs generated during WebHCat uninstallation or before WebHCat installation
	/webhcat/hive-omm- <i><Date></i> - <i><PID></i> -gc.log. <i><No.></i>	GC log file of the WebHCat process.
	/webhcat/webhcat.log	Run log file of the WebHCat process
Audit log	hive-audit.log hive-rangeraudit.log	HiveServer audit log file
	metastore-audit.log	MetaStore audit log file.
	webhcat-audit.log	WebHCat audit log file.
	jetty- <i><Date></i> .request.log	Request logs of the jetty service.

Log Levels

Table 10-13 describes the log levels supported by Hive.

Levels of run logs are ERROR, WARN, INFO, and DEBUG from the highest to the lowest priority. Run logs of equal or higher levels are recorded. The higher the specified log level, the fewer the logs recorded.

Table 10-13 Log levels

Level	Description
ERROR	Logs of this level record error information about system running.
WARN	Logs of this level record exception information about the current event processing.
INFO	Logs of this level record normal running status information about the system and events.
DEBUG	Logs of this level record the system information and system debugging information.

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page of the Yarn service by referring to [Modifying Cluster Service Configuration Parameters](#).
- Step 2** On the menu bar on the left, select the log menu of the target role.
- Step 3** Select a desired log level and save the configuration.

 **NOTE**

The Hive log level takes effect immediately after being configured. You do not need to restart the service.

----End

Log Formats

The following table lists the Hive log formats:

Table 10-14 Log formats

Log Type	Format	Example
Run log	<yyyy-MM-dd HH:mm:ss,SSS> <LogLevel> <Thread that generates the log> <Message in the log> <Location of the log event>	2014-11-05 09:45:01,242 INFO main Starting hive metastore on port 21088 org.apache.hadoop.hive.metastore.HiveMetaStore.main(HiveMetaStore.java:5198)

Log Type	Format	Example
Audit log	<yyyy-MM-dd HH:mm:ss,SSS> <LogLevel> <Thread that generates the log> <User Name><User IP><Time><Operation><Re source><Result><Detail > < Location of the log event >	2018-12-24 12:16:25,319 INFO HiveServer2-Handler- Pool: Thread-185 UserName=hive UserIP=10.153.2.204 Time=2018/12/24 12:16:25 Operation=CloseSession Result=SUCCESS Detail= org.apache.hive.service.cli.thrif t.ThriftCLIService.logAuditEven t(ThriftCLIService.java:434)

10.32 Hive Performance Tuning

10.32.1 Creating Table Partitions

Scenario

During the Select query, Hive generally scans the entire table, which is time-consuming. To improve query efficiency, create table partitions based on service requirements and query dimensions.

Procedure

Step 1 For versions earlier than MRS 3.x:

Log in to the MRS console. In the left navigation pane, choose **Clusters > Active Clusters**, and click a cluster name. Choose **Nodes > Node**. The ECS page is displayed. Click **Remote Login** to log in to the Hive node.

For MRS 3.x or later:

Log in to the node where the Hive client has been installed as user **root**.

Step 2 Run the following command to go to the client installation directory, for example, **/opt/client**.

```
cd /opt/client
```

Step 3 Run the **source bigdata_env** command to configure environment variables for the client.

Step 4 Run the following command on the client for login:

```
kinit Username
```

Step 5 Run the following command to log in to the client tool:

```
beeline
```

Step 6 Select the static or dynamic partition.

- **Static partition:**
Manually enter a partition name, and use the keyword **PARTITIONED BY** to specify partition column name and data type when creating a table. During application development, use the **ALTER TABLE ADD PARTITION** statement to add a partition and use the **LOAD DATA INTO PARTITION** statement to load data to the partition, which supports only static partitions.
- **Dynamic partition:** Use a query command to insert results to a partition of a table. The partition can be a dynamic partition.

The dynamic partition can be enabled on the client tool by running the following command:

```
set hive.exec.dynamic.partition=true;
```

The default mode of the dynamic partition is strict. That is, at least a column must be specified as a static partition, under which dynamic sub-partitions can be created. You can run the following command to enable a completely dynamic partition:

```
set hive.exec.dynamic.partition.mode=nonstrict;
```

NOTE

- The dynamic partition may cause a DML statement to create a large number of partitions and new mapping folders, which deteriorates system performance.
- If there are a large number of files, it takes a long time to run a SQL statement. You can run the **set mapreduce.input.fileinputformat.list-status.num-threads = 100;** statement before running a SQL statement to shorten the time. The parameter **mapreduce.input.fileinputformat.list-status.num-threads** can be set only after being added to the Hive whitelist.

----End

10.32.2 Optimizing Join

Scenario

When the Join statement is used, the command execution speed and query speed may be slow in case of large data volume. To resolve this problem, you can optimize Join.

Join optimization can be classified into the following modes:

- Map Join
- Sort Merge Bucket Map Join
- Optimizing Join Sequences

Map Join

Hive Map Join applies to small tables (the table size is less than 25 MB) that can be stored in the memory. The table size can be defined using **hive.mapjoin.smalltable.filesize**, and the default table size is 25 MB.

Map Join has two methods:

- Use `/*+ MAPJOIN(join_table) */`.
- Set the following parameter before running the statement. The default value is true in the current version.

```
set hive.auto.convert.join=true;
```

There is no Reduce task when Map Join is used. Instead, a MapReduce Local Task is created before the Map job. The task uses TableScan to read small table data to the local computer, saves and writes the data in HashTable mode to a hard disk on the local computer, upload the data to DFS, and saves the data in distributed cache. The small table data that the map task reads from the local disk or distributed cache is the output together with the large table join result.

When using Map Join, make sure that the size of small tables cannot be too large. If small tables use up memory, the system performance will deteriorate and even memory leakage occurs.

Sort Merge Bucket Map Join

The following conditions must be met before using Sort Merge Bucket Map Join:

- The two Join tables are large and cannot be stored in the memory.
- The two tables are bucketed (clustered by (column)) and sorted (sorted by(column)) according to the join key, and the buckets counts of the two tables are in integral multiple relationship.

Set the following parameters to enable Sort Merge Bucket Map Join:

```
set hive.optimize.bucketmapjoin=true;
```

```
set hive.optimize.bucketmapjoin.sortedmerge=true;
```

This type of Map Join does not have Reduce tasks too. A MapReduce Local Task is started before the Map job to read small table data by bucket to the local computer. The local computer saves the HashTable backup of multiple buckets and writes the backup into HDFS. The backup is also saved in the distributed cache. The small table data that the map task reads from the local disk or distributed cache by bucket is the output after mapping with the large table.

Optimizing Join Sequences

If the Join operation is to be performed on three or more tables and different Join sequences are used, the execution time will be greatly different. Using an appropriate Join sequence can shorten the time for task execution.

Rules of a Join sequence:

- A table with small data volume or a combination with fewer results generated after a Join operation is executed first.
- A table with large data volume or a combination with more results generated after a Join operation is executed later.

For example, the **customer** table has the largest data volume, and fewer results will be generated if a Join operation is performed on the **orders** and **lineitem** tables first.

The original Join statement is as follows.

```
select
  l_orderkey,
  sum(l_extendedprice * (1 - l_discount)) as revenue,
  o_orderdate,
```

```
o_shippriority
from
  customer,
  orders,
  lineitem
where
  c_mktsegment = 'BUILDING'
  and c_custkey = o_custkey
  and l_orderkey = o_orderkey
  and o_orderdate < '1995-03-22'
  and l_shipdate > '1995-03-22'
limit 10;
```

After the sequence is optimized, the Join statements are as follows:

```
select
  l_orderkey,
  sum(l_extendedprice * (1 - l_discount)) as revenue,
  o_orderdate,
  o_shippriority
from
  orders,
  lineitem,
  customer
where
  c_mktsegment = 'BUILDING'
  and c_custkey = o_custkey
  and l_orderkey = o_orderkey
  and o_orderdate < '1995-03-22'
  and l_shipdate > '1995-03-22'
limit 10;
```

Precautions

Join Data Skew Problem

Data skew refers to the symptom that the task progress is 99% for a long time.

Data skew often exists because the data volume of a few Reduce tasks is much larger than that of others. Most Reduce tasks are complete while a few Reduce tasks are not complete.

To resolve the data skew problem, set **hive.optimize.skewjoin=true** and adjust the value of **hive.skewjoin.key**. **hive.skewjoin.key** specifies the maximum number of keys received by a Reduce task. If the number reaches the maximum, the keys are atomically distributed to other Reduce tasks.

10.32.3 Optimizing Group By

Scenario

Optimize the Group by statement to accelerate the command execution and query speed.

During the Group by operation, Map performs grouping and distributes the groups to Reduce; Reduce then performs grouping again. Group by optimization can be performed by enabling Map aggregation to reduce Map output data volume.

Procedure

On a Hive client, set the following parameter:

```
set hive.map.aggr=true
```

Precautions

Group By Data Skew

Group by have data skew problems. When `hive.groupby.skewindata` is set to true, the created query plan has two MapReduce jobs. The Map output result of the first job is randomly distributed to Reduce tasks, and each Reduce task performs aggregation operations and generates output result. Such processing may distribute the same Group By Key to different Reduce tasks for load balancing purpose. According to the preprocessing result, the second Job distributes Group By Key to Reduce to complete the final aggregation operation.

Count Distinct Aggregation Problem

When the aggregation function `count distinct` is used in deduplication counting, serious Reduce data skew occurs if the processed value is empty. The empty value can be processed independently. If `count distinct` is used, exclude the empty value using the `where` statement and increase the last `count distinct` result by 1. If there are other computing operations, process the empty value independently and then combine the value with other computing results.

10.32.4 Optimizing Data Storage

Scenario

ORC is an efficient column storage format and has higher compression ratio and reading efficiency than other file formats.

You are advised to use **ORC** as the default Hive table storage format.

Prerequisites

You have logged in to the Hive client. For details, see [Using a Hive Client](#).

Procedure

- Recommended: **SNAPPY** compression, which applies to scenarios with even compression ratio and reading efficiency requirements.
Create table *xx* (*col_name data_type*) stored as orc tblproperties ("orc.compress"="SNAPPY");
- Available: **ZLIB** compression, which applies to scenarios with high compression ratio requirements.

Create table *xx* (*col_name data_type*) stored as orc tblproperties ("orc.compress"="ZLIB");

NOTE

xx indicates the specific Hive table name.

10.32.5 Optimizing SQL Statements

Scenario

When SQL statements are executed on Hive, if the **(a&b) or (a&c)** logic exists in the statements, you are advised to change the logic to **a & (b or c)**.

Example

If condition a is **p_partkey = l_partkey**, the statements before optimization are as follows:

```
select
  sum(l_extendedprice* (1 - l_discount)) as revenue
from
  lineitem,
  part
where
  (
    p_partkey = l_partkey
    and p_brand = 'Brand#32'
    and p_container in ('SM CASE', 'SM BOX', 'SM PACK', 'SM PKG')
    and l_quantity >= 7 and l_quantity <= 7 + 10
    and p_size between 1 and 5
    and l_shipmode in ('AIR', 'AIR REG')
    and l_shipinstruct = 'DELIVER IN PERSON'
  )
  or
  (
    p_partkey = l_partkey
    and p_brand = 'Brand#35'
    and p_container in ('MED BAG', 'MED BOX', 'MED PKG', 'MED PACK')
    and l_quantity >= 15 and l_quantity <= 15 + 10
    and p_size between 1 and 10
    and l_shipmode in ('AIR', 'AIR REG')
    and l_shipinstruct = 'DELIVER IN PERSON'
  )
  or
  (
    p_partkey = l_partkey
    and p_brand = 'Brand#24'
    and p_container in ('LG CASE', 'LG BOX', 'LG PACK', 'LG PKG')
    and l_quantity >= 26 and l_quantity <= 26 + 10
    and p_size between 1 and 15
    and l_shipmode in ('AIR', 'AIR REG')
    and l_shipinstruct = 'DELIVER IN PERSON'
  )
)
```

The statements after optimization are as follows:

```
select
  sum(l_extendedprice* (1 - l_discount)) as revenue
from
  lineitem,
  part
where p_partkey = l_partkey and
  ((
    p_brand = 'Brand#32'
    and p_container in ('SM CASE', 'SM BOX', 'SM PACK', 'SM PKG')
    and l_quantity >= 7 and l_quantity <= 7 + 10
    and p_size between 1 and 5
    and l_shipmode in ('AIR', 'AIR REG')
    and l_shipinstruct = 'DELIVER IN PERSON'
  )
  or
  (
    p_brand = 'Brand#35'
    and p_container in ('MED BAG', 'MED BOX', 'MED PKG', 'MED PACK')
```

```
and l_quantity >= 15 and l_quantity <= 15 + 10
and p_size between 1 and 10
and l_shipmode in ('AIR', 'AIR REG')
and l_shipinstruct = 'DELIVER IN PERSON'
)
or
(
  p_brand = 'Brand#24'
  and p_container in ('LG CASE', 'LG BOX', 'LG PACK', 'LG PKG')
  and l_quantity >= 26 and l_quantity <= 26 + 10
  and p_size between 1 and 15
  and l_shipmode in ('AIR', 'AIR REG')
  and l_shipinstruct = 'DELIVER IN PERSON'
))
```

10.32.6 Optimizing the Query Function Using Hive CBO

Scenario

When joining multiple tables in Hive, Hive supports Cost-Based Optimization (CBO). The system automatically selects the optimal plan based on the table statistics, such as the data volume and number of files, to improve the efficiency of joining multiple tables. Hive needs to collect table statistics before CBO optimization.

NOTE

- The CBO optimizes the joining sequence based on statistics and search criteria. However, the joining sequence may fail to be optimized in some special scenarios, such as data skew occurs and query condition values are not in the table.
- When column statistics collection is enabled, Reduce operations must be performed for aggregation. For insert tasks without the Reduce phase, Reduce operations will be performed to collect statistics.
- This section applies to MRS 3.x or later.

Prerequisites

You have logged in to the Hive client. For details, see [Using a Hive Client](#).

Procedure

Step 1 On the Manager UI, search for the **hive.cbo.enable** parameter in the service configuration of the Hive component, and select **true** to enable the function permanently.

Step 2 Collect statistics about the existing data in Hive tables manually.

Run the following command to manually collect statistics: Statistics about only one table can be collected. If statistics about multiple tables need to be collected, the command needs to be executed repeatedly.

```
ANALYZE TABLE [db_name.]tablename [PARTITION(partcol1[=val1],  
partcol2[=val2], ...)]
```

```
COMPUTE STATISTICS
```

```
[FOR COLUMNS]
```

```
[NOSCAN];
```


 NOTE

- When **FOR COLUMNS** is specified, column-level statistics are collected.
- When **NOSCAN** is specified, statistics about the file size and number of files will be collected, but specific files will not be scanned.

For example:

analyze table table_name compute statistics;

analyze table table_name compute statistics for columns;

Step 3 Configure the automatic statistics collection function of Hive. After the function is enabled, new statistics will be collected only when you insert data by running the **insert overwrite/into** command.

- Run the following commands on the Hive client to enable the statistics collection function temporarily:
set hive.stats.autogather = true; enables the automatic collection of table/partition-level statistics.
set hive.stats.column.autogather = true; enables the automatic collection of column-level statistics.

 NOTE

- The column-level statistics collection does not support complex data types, such as Map and Struct.
- The automatic table-level statistics collection does not support Hive on HBase tables.
- On the Manager UI, search for the **hive.stats.autogather** and **hive.stats.column.autogather** parameters in the service configuration of Hive, and select **true** to enable the collection function permanently.

Step 4 Run the following command to view statistics:

DESCRIBE FORMATTED table_name[.column_name] PARTITION partition_spec;

For example:

desc formatted table_name;

desc formatted table_name.id;

desc formatted table_name.id partition(time='2016-05-27');

 NOTE

Partition tables only support partition-level statistics collection, so you must specify partitions to query statistics for partition tables.

----End

10.33 Common Issues About Hive

10.33.1 How Do I Delete UDFs on Multiple HiveServers at the Same Time?

Question

How can I delete permanent user-defined functions (UDFs) on multiple HiveServers at the same time?

Answer

Multiple HiveServers share one MetaStore database. Therefore, there is a delay in the data synchronization between the MetaStore database and the HiveServer memory. If a permanent UDF is deleted from one HiveServer, the operation result cannot be synchronized to the other HiveServers promptly.

In this case, you need to log in to the Hive client to connect to each HiveServer and delete permanent UDFs on the HiveServers one by one. The operations are as follows:

Step 1 Log in to the node where the Hive client is installed as the Hive client installation user.

Step 2 Run the following command to go to the client installation directory:

```
cd Client installation directory
```

For example, if the client installation directory is **/opt/client**, run the following command:

```
cd /opt/client
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 Run the following command to authenticate the user:

```
kinit Hive service user
```

NOTE

The login user must have the Hive admin rights.

Step 5 Run the following command to connect to the specified HiveServer:

```
beeline -u "jdbc:hive2://10.39.151.74:21066/default;sasl.qop=auth-conf;auth=KERBEROS;principal=hive/hadoop.<system domain name>@<system domain name>"
```

 NOTE

- `10.39.151.74` is the IP address of the node where the HiveServer is located.
- `21066` is the port number of the HiveServer. The HiveServer port number ranges from 21066 to 21070 by default. Use the actual port number.
- `hive` is the username. For example, if the Hive1 instance is used, the username is `hive1`.
- You can log in to FusionInsight Manager, choose **System > Permission > Domain and Mutual Trust**, and view the value of **Local Domain**, which is the current system domain name.
- `hive/hadoop.<system domain name>` is the username. All letters in the system domain name contained in the username are lowercase letters.

Step 6 Run the following command to enable the Hive admin rights:

```
set role admin;
```

Step 7 Run the following command to delete the permanent UDF:

```
drop function function_name;
```

 NOTE

- `function_name` indicates the name of the permanent function.
- If the permanent UDF is created in Spark, the permanent UDF needs to be deleted from Spark and then from HiveServer by running the preceding command.

Step 8 Check whether the permanent UDFs are deleted from all HiveServers.

- If yes, no further action is required.
- If no, go to [Step 5](#).

----End

10.33.2 Why Cannot the DROP operation Be Performed on a Backed-up Hive Table?

Question

Why cannot the **DROP** operation be performed for a backed up Hive table?

Answer

Snapshots have been created for an HDFS directory mapping to the backed up Hive table, so the HDFS directory cannot be deleted. As a result, the Hive table cannot be deleted.

When a Hive table is being backed up, snapshots are created for the HDFS directory mapping to the table. The snapshot mechanism of HDFS has the following limitation: If snapshots have been created for an HDFS directory, the directory cannot be deleted or renamed unless the snapshots are deleted. When the **DROP** operation is performed for a Hive table (except the EXTERNAL table), the system attempts to delete the HDFS directory mapping to the table. If the directory fails to be deleted, the system displays a message indicating that the table fails to be deleted.

If you need to delete this table, manually delete all backup tasks related to this table.

10.33.3 How to Perform Operations on Local Files with Hive User-Defined Functions

Question

How to perform operations on local files (such as reading the content of a file) with Hive user-defined functions?

Answer

By default, you can perform operations on local files with their relative paths in UDF. The following are sample codes:

```
public String evaluate(String text) {  
    // some logic  
    File file = new File("foo.txt");  
    // some logic  
    // do return here  
}
```

In Hive, upload the file **foo.txt** used in UDF to HDFS, such as **hdfs://hacluster/tmp/foo.txt**. You can perform operations on the **foo.txt** file by creating UDF with the following sentences:

```
create function testFunc as 'some.class' using jar 'hdfs://hacluster/  
somejar.jar', file 'hdfs://hacluster/tmp/foo.txt';
```

In abnormal cases, if the value of **hive.fetch.task.conversion** is **more**, you can perform operations on local files in UDF by using absolute path instead of relative path. In addition, you must ensure that the file exists on all HiveServer nodes and NodeManager nodes and **omm** user have corresponding operation rights.

10.33.4 How Do I Forcibly Stop MapReduce Jobs Executed by Hive?

Question

How do I stop a MapReduce task manually if the task is suspended for a long time?

Answer

- Step 1** Log in to FusionInsight Manager.
- Step 2** Choose **Cluster > Name of the desired cluster > Services > Yarn**.
- Step 3** On the left pane, click **ResourceManager(Host name, Active)**, and log in to Yarn.
- Step 4** Click the button corresponding to the task ID. On the task page that is displayed, click **Kill Application** in the upper left corner and click **OK** in the displayed dialog box to stop the task.

----End

10.33.5 How Do I Monitor the Hive Table Size?

Question

How do I monitor the Hive table size?

Answer

The HDFS refined monitoring function allows you to monitor the size of a specified table directory.

Prerequisites

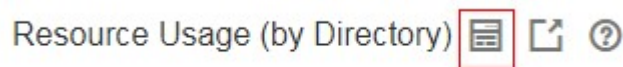
- The Hive and HDFS components are running properly.
- The HDFS refined monitoring function is normal.

Procedure

Step 1 Log in to FusionInsight Manager.

Step 2 Choose **Cluster** > *Name of the desired cluster* > **Services** > **HDFS** > **Resource**.

Step 3 Click the first icon in the upper left corner of **Resource Usage (by Directory)**, as shown in the following figure.



Step 4 In the displayed sub page for configuring space monitoring, click **Add**.

Step 5 In the displayed **Add a Monitoring Directory** dialog box, set **Name** to the name or the user-defined alias of the table to be monitored and **Path** to the path of the monitored table. Click **OK**. In the monitoring result, the horizontal coordinate indicates the time, and the vertical coordinate indicates the size of the monitored directory.

----End

10.33.6 How Do I Prevent Key Directories from Data Loss Caused by Misoperations of the insert overwrite Statement?

Question

How do I prevent key directories from data loss caused by misoperations of the **insert overwrite** statement?

Answer

During monitoring of key Hive databases, tables, or directories, to prevent data loss caused by misoperations of the **insert overwrite** statement, configure **hive.local.dir.confblacklist** in Hive to protect directories.

This configuration item has been configured for directories such as **/opt/** and **/user/hive/warehouse** by default.

Prerequisites

The Hive and HDFS components are running properly.

Procedure

- Step 1** Log in to FusionInsight Manager.
- Step 2** Choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive** > **Configurations** > **All Configurations**, and search for the **hive.local.dir.confblacklist** configuration item.
- Step 3** Add paths of databases, tables, or directories to be protected in the parameter value.
- Step 4** Click **Save** to save the settings.

----End

10.33.7 Why Is Hive on Spark Task Freezing When HBase Is Not Installed?

Scenario

This function applies to Hive.

Perform the following operations to configure parameters. When Hive on Spark tasks are executed in the environment where the HBase is not installed, freezing of tasks can be prevented.

NOTE

The Spark kernel version of Hive on Spark tasks has been upgraded to Spark2x. Hive on Spark tasks can be executed if Spark2x is not installed. If HBase is not installed, when Spark tasks are executed, the system attempts to connect to the ZooKeeper to access HBase until timeout occurs by default. As a result, task freezing occurs.

If HBase is not installed, perform the following operations to execute Hive on Spark tasks. If HBase is upgraded from an earlier version, you do not need to configure parameters after the upgrade.

Procedure

- Step 1** Log in to FusionInsight Manager.
- Step 2** Choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive** > **Configurations** > **All Configurations**.
- Step 3** Choose **HiveServer(Role)** > **Customization**. Add a customized parameter to the **spark-defaults.conf** parameter file. Set **Name** to **spark.security.credentials.hbase.enabled**, and set **Value** to **false**.
- Step 4** Click **Save**. In the dialog box that is displayed, click **OK**.

Step 5 Choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive** > **Instance**, select all Hive instances, choose **More** > **Restart Instance**, enter the password, and click **OK**.

----End

10.33.8 Error Reported When the WHERE Condition Is Used to Query Tables with Excessive Partitions in FusionInsight Hive

Question

When a table with more than 32,000 partitions is created in Hive, an exception occurs during the query with the WHERE partition. In addition, the exception information printed in **metastore.log** contains the following information:

```
Caused by: java.io.IOException: Tried to send an out-of-range integer as a 2-byte value: 32970
    at org.postgresql.core.PGStream.SendInteger2(PGStream.java:199)
    at org.postgresql.core.v3.QueryExecutorImpl.sendParse(QueryExecutorImpl.java:1330)
    at org.postgresql.core.v3.QueryExecutorImpl.sendOneQuery(QueryExecutorImpl.java:1601)
    at org.postgresql.core.v3.QueryExecutorImpl.sendParse(QueryExecutorImpl.java:1191)
    at org.postgresql.core.v3.QueryExecutorImpl.execute(QueryExecutorImpl.java:346)
```

Answer

During a query with partition conditions, HiveServer optimizes the partitions to avoid full table scanning. All partitions whose metadata meets the conditions need to be queried. However, the **sendOneQuery** interface provided by GaussDB limits the parameter value to **32767** in the **sendParse** method. If the number of partition conditions exceeds **32767**, an exception occurs.

10.33.9 Why Cannot I Connect to HiveServer When I Use IBM JDK to Access the Beeline Client?

Scenario

When users check the JDK version used by the client, if the JDK version is IBM JDK, the Beeline client needs to be reconstructed. Otherwise, the client will fail to connect to HiveServer.

Procedure

- Step 1** Log in to FusionInsight Manager and choose **System** > **Permission** > **User**. In the **Operation** column of the target user, choose **More** > **Download Authentication Credential**, select the cluster information, and click **OK** to download the keytab file.
- Step 2** Decompress the keytab file and use WinSCP to upload the decompressed **user.keytab** file to the Hive client installation directory on the node to be operated, for example, **/opt/client**.
- Step 3** Run the following command to open the **Hive/component_env** configuration file in the Hive client directory:

```
vi Hive client installation directory/Hive/component_env
```

Add the following content to the end of the line where **export CLIENT_HIVE_URI** is located:
`\; user.principal=Username @HADOOP.COM\;user.keytab=user.keytab file path/user.keytab`

----End

10.33.10 Description of Hive Table Location (Either Be an OBS or HDFS Path)

Question

Does a Hive Table Can Be Stored Either in OBS or HDFS?

Answer

1. The location of a common Hive table stored on OBS can be set to an HDFS path.
2. In the same Hive service, you can create tables stored in OBS and HDFS, respectively.
3. For a Hive partitioned table stored on OBS, the location of the partition cannot be set to an HDFS path. (For a partitioned table stored on HDFS, the location of the partition cannot be changed to OBS.)

10.33.11 Why Cannot Data Be Queried After the MapReduce Engine Is Switched After the Tez Engine Is Used to Execute Union-related Statements?

Question

Hive uses the Tez engine to execute union-related statements to write data. After Hive is switched to the MapReduce engine for query, no data is found.

Answer

When Hive uses the Tez engine to execute the union-related statement, the generated output file is stored in the **HIVE_UNION_SUBDIR** directory. After Hive is switched back to the MapReduce engine, files in the directory are not read by default. Therefore, data in the **HIVE_UNION_SUBDIR** directory is not read.

In this case, you can set **mapreduce.input.fileinputformat.input.dir.recursive** to **true** to enable union optimization and determine whether to read data in the directory.

10.33.12 Why Does Hive Not Support Concurrent Data Writing to the Same Table or Partition?

Question

Why Does Data Inconsistency Occur When Data Is Concurrently Written to a Hive Table Through an API?

Answer

Hive does not support concurrent data insertion for the same table or partition. As a result, multiple tasks perform operations on the same temporary data directory, and one task moves the data of another task, causing task data exception. The service logic is modified so that data is inserted to the same table or partition in single thread mode.

10.33.13 Why Does Hive Not Support Vectorized Query?

Question

When the vectorized parameter **hive.vectorized.execution.enabled** is set to **true**, why do some null pointers or type conversion exceptions occur occasionally when Hive on Tez/MapReduce/Spark is executed?

Answer

Currently, Hive does not support vectorized execution. Many community issues are introduced during vectorized execution and are not resolved stably. The default value of **hive.vectorized.execution.enabled** is **false**. You are advised not to set this parameter to **true**.

10.33.14 Why Does Metadata Still Exist When the HDFS Data Directory of the Hive Table Is Deleted by Mistake?

Question

The HDFS data directory of the Hive table is deleted by mistake, but the metadata still exists. As a result, an error is reported during task execution.

Answer

This is a exception caused by misoperation. You need to manually delete the metadata of the corresponding table and try again.

Example:

Run the following command to go to the console:

```
source ${BIGDATA_HOME}/FusionInsight_BASE_8.0.2.1/install/FusionInsight-  
dbservice-2.7.0/.dbservice_profile
```

```
gsql -p 20051 -U hive -d hivemeta -W HiveUser@
```

Run the **delete from tbls where tbl_id='xxx'**; command.

10.33.15 Hive Configuration Problems

- The error message "java.lang.OutOfMemoryError: Java heap space." is displayed during Hive SQL execution.

Solution:

- For MapReduce tasks, increase the values of the following parameters:

- set mapreduce.map.memory.mb=8192;**
 - set mapreduce.map.java.opts=-Xmx6554M;**
 - set mapreduce.reduce.memory.mb=8192;**
 - set mapreduce.reduce.java.opts=-Xmx6554M;**
- For Tez tasks, increase the value of the following parameter:
 - set hive.tez.container.size=8192;**
- After a column name is changed to a new one using the Hive SQL **as** statement, the error message "Invalid table alias or column reference 'xxx!'" is displayed when the original column name is used for compilation.

Solution: Run the **set hive.cbo.enable=true;** statement.
- The error message "Unsupported SubQuery Expression 'xxx': Only SubQuery expressions that are top level conjuncts are allowed." is displayed during Hive SQL subquery compilation.

Solution: Run the **set hive.cbo.enable=true;** statement.
- The error message "CalciteSubquerySemanticException [Error 10249]: Unsupported SubQuery Expression Currently SubQuery expressions are only allowed as Where and Having Clause predicates." is displayed during Hive SQL subquery compilation.

Solution: Run the **set hive.cbo.enable=true;** statement.
- The error message "Error running query: java.lang.AssertionError: Cannot add expression of different type to set." is displayed during Hive SQL compilation.

Solution: Run the **set hive.cbo.enable=false;** statement.
- The error message "java.lang.NullPointerException at org.apache.hadoop.hive.ql.udf.generic.GenericUDAFComputeStats \$GenericUDAFNumericStatsEvaluator.init." is displayed during Hive SQL execution.

Solution: Run the **set hive.map.aggr=false;** statement.
- When **hive.auto.convert.join** is set to **true** (enabled by default) and **hive.optimize.skewjoin** is set to **true**, the error message "ClassCastException org.apache.hadoop.hive.ql.plan.ConditionalWork cannot be cast to org.apache.hadoop.hive.ql.plan.MapredWork" is displayed.

Solution: Run the **set hive.optimize.skewjoin=false;** statement.
- When **hive.auto.convert.join** is set to **true** (enabled by default), **hive.optimize.skewjoin** is set to **true**, and **hive.exec.parallel** is set to **true**, the error message "java.io.FileNotFoundException: File does not exist:xxx/reduce.xml" is displayed.

Solution:

 - Method 1: Switch the execution engine to Tez. For details, see [Switching the Hive Execution Engine to Tez](#).
 - Method 2: Run the **set hive.exec.parallel=false;** statement.
 - Method 3: Run the **set hive.auto.convert.join=false;** statement.

11 Using Hue (Versions Earlier Than MRS 3.x)

11.1 Using Hue from Scratch

Hue provides the file browser function using a graphical user interface (GUI) so that you can view files and directories on Hive.

Prerequisites

You have installed Hive and Hue, and the Kerberos authentication cluster in the running state.

Procedure

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 Open the Hue web UI and choose **Query Editors > Hive**.


Step 3 In **Databases**, select a Hive database, the default database is **default**.


The system displays all available tables. You can enter a keyword of the table name to search for the desired table.

Step 4 Click the desired table name. All columns in the table are displayed.


Step 5 Enter the HiveQL statements in the area for editing.

```
create table hue_table(id int,name string,company string) row format delimited fields terminated by ',' stored as textfile;
```

Click  and select **Explain**. The editor checks the syntax and execution plan of the entered HiveQL statements. If the statements have syntax errors, the editor reports **Error while compiling statement**.

Step 6 Click , and select the engine for executing the HiveQL statements.

Step 7 Click  to execute the HiveQL statements.

Step 8 In the command text box, enter **show tables;** and click . Check whether the **hue-table** table created in [Step 5](#) exists in the result.

----End

11.2 Accessing the Hue Web UI

Scenario

After Hue is installed in an MRS cluster, users can use Hadoop and Hive on the Hue web UI.

This section describes how to open the Hue web UI on the MRS cluster.

NOTE

To access the Hue web UI, you are advised to use a browser that is compatible with the Hue WebUI, for example, Google Chrome 50. The Internet Explorer may be incompatible with the Hue web UI.

Impact on the System

Site trust must be added to the browser when you access Manager and Hue web UI for the first time. Otherwise, the Hue web UI cannot be accessed.

Prerequisites

When Kerberos authentication is enabled, the MRS cluster administrator has assigned the permission for using Hive to the user. For details, see [Creating a User](#). For example, create a human-machine user named **hueuser**, add the user to user groups **hive** (the primary group), **hadoop**, and **supergroup**, and role **System_administrator**.

This user is used to log in to the Hue WebUI.

Procedure

Step 1 Log in to the service page, click the cluster name to go to the cluster details page, and choose **Components**.



NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

Step 2 Select **Hue**. On the right side of **Hue WebUI**, click the link to log in to the Hue web UI as user **hueuser**.

Hue WebUI provides the following functions:

- If Hive is installed in the MRS cluster, you can use **Query Editors** to execute query statements of Hive. Hive has been installed in the MRS cluster.
- If Hive is installed in the MRS cluster, you can use **Data Browsers** to manage Hive tables.

- If HDFS is installed in the MRS cluster, you can use  to view directories and files in HDFS.
- If Yarn is installed in the MRS cluster, you can use  to view all jobs in the MRS cluster.

 **NOTE**

- When you log in to the Hue web UI as user **hueuser** for the first time, you need to change the password.
- After obtaining the URL for accessing the Hue web UI, you can give the URL to other users who cannot access MRS Manager for accessing the Hue web UI.
- If you perform operations on the Hue WebUI only but not on Manager, you must enter the password of the current login user when accessing Manager again.

----End

11.3 Hue Common Parameters

Navigation Path

For details about how to set parameters, see [Modifying Cluster Service Configuration Parameters](#).

Parameters

Table 11-1 Hue common parameters

Parameter	Description	Default Value	Value Range
HANDLER_ACCESSLOG_LEVEL	Hue access log level	DEBUG	<ul style="list-style-type: none"> • ERROR • WARN • INFO • DEBUG
HANDLER_AUDITLOG_LEVEL	Hue audit log level	DEBUG	<ul style="list-style-type: none"> • ERROR • WARN • INFO • DEBUG
HANDLER_ERRORLOG_LEVEL	Hue error log level	ERROR	<ul style="list-style-type: none"> • ERROR • WARN • INFO • DEBUG

Parameter	Description	Default Value	Value Range
HANDLER_LOGFILE_LEVEL	Hue run log level	INFO	<ul style="list-style-type: none"> • ERROR • WARN • INFO • DEBUG
HANDLER_LOGFILE_MAXBACKUPINDEX	Maximum number of Hue log files.	20	1 to 999
HANDLER_LOGFILE_SIZE	Maximum size of a Hue log file.	5 MB	-

11.4 Using HiveQL Editor on the Hue Web UI

Scenario


Users can use the Hue web UI to execute HiveQL statements in a cluster.

Accessing Query Editors

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 Choose **Query Editors > Hive**. The **Hive** page is displayed.

Hive supports the following functions:

- Executes and manages HiveQL statements.
- View the HiveQL statements saved by the current user in **Saved Queries**.
- Query HiveQL statements executed by the current user in **Query History**.
- Click  to display all databases included in **Databases** of Hive.

----End


Executing HiveQL Statements

Step 1 Choose **Query Editors > Hive**. The **Hive** page is displayed.


Step 2 Click  and select a database from **Databases**. The default database is **default**.


The system displays all available tables in the database. You can enter a keyword of the table name to search for the desired table.

Step 3 Click the desired table name. All columns in the table are displayed.

Move the cursor to the row of the table and click . Column details are displayed.

Step 4 Enter the query statements in the area for editing HiveQL statements.

Click  and select **Explain**. The editor checks the syntax and execution plan of the entered statements. If the statements have syntax errors, the editor reports **Error while compiling statement**.

Step 5 Click  and select the engine for executing the HiveQL statements.







- **mr**: MapReduce computing framework
- **spark**: Spark computing framework
- **tez**: Tez computing framework

 **NOTE**

Tez is applicable to MRS 1.9.x and later versions.

Step 6 Click  to execute the HiveQL statements.

 **NOTE**

- If you want to use the entered HiveQL statements again, click  to save them.
- To format HiveQL statements, click  and select **Format**.
- To delete an entered HiveQL statement, click  and select **Clear**.
- Clear the entered statement and execute a new statement. Click  and select **New query**.
- Viewing history:
Click **Query History** to view the HiveQL running status. You can view the history of all the statements or only the saved statements. If many historical records exist, you can enter keywords in the text box to search for desired records.
- Advanced query configuration:
Click  in the upper right corner to configure information such as files, functions, and settings.
- Viewing the information of shortcut keys:
Click  in the upper right corner to view all shortcut keys.

----End

Viewing Execution Results

Step 1 In the **Hive** execution area, **Query History** is displayed by default.

Step 2 Click **Results** to view the execution result of the executed statement.

----End

Managing Query Statements

Step 1 Choose **Query Editors > Hive**. The **Hive** page is displayed.

Step 2 Click **Saved Queries**.


Click a saved statement. The system automatically adds the statement to the editing area.


----End


Modifying Query Editors Settings


Step 1 On the **Hive** tab page, click .


Step 2 Click  on the right of **Files** and click  to specify the directory for storing the file.

You can click  to add a file resource.

Step 3 Click  on the right of **Functions** and enter the names of user-defined function and function class.

You can click  to add a customized function.

Step 4 Click  on the right of **Settings**, enter the Hive parameter name in the **Key**, and value in **Value**. The current Hive session connects to Hive based on the customized configuration.

You can click  to add a parameter.

----End

11.5 Using the Metadata Browser on the Hue Web UI


Scenario




Users can use the Hue web UI to manage Hive metadata in an MRS cluster.

Using Metastore Manager

Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Choose **Data Browsers > Metastore Tables**, and access **Metastore Manager**.



- Viewing metadata of Hive tables
In the left navigation pane, move the cursor to a table and click  on the right. The metadata of the Hive table is displayed.
- Managing metadata of Hive tables

On the metadata page of a Hive table, you can click  in the upper right corner to import data, click  to browse data, and click  to view the location of the table file.

 **CAUTION**

The Hue page is used to view and analyze data such as files and tables. Do not perform high-risk management operations such as deleting objects on the page. If an operation is required, you are advised to perform the operation on each component after confirming that the operation has no impact on services. For example, you can use the HDFS client to perform operations on HDFS files and use the Hive client to perform operations on Hive tables.

- Managing Hive metadata tables

Click  in the upper right corner to create a table in the database based on the uploaded files. Or click  in the upper right corner to manually create a table.

Accessing Metastore Manager

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 Choose **Data Browsers > Metastore Tables**, and access **Metastore Manager**.

Metastore Manager supports the following functions:

- Creating a Hive table from a file
- Manually creating a Hive table
- Viewing Hive table metadata

----End


Creating a Hive table from a File

Step 1 Access **Metastore Manager** and select a database in **Databases**.

The default database is **default**.

Step 2 Click . The **Create a new table from a file** page is displayed.

Step 3 Select a file.

1. In **Table Name**, enter a Hive table name.
A Hive table name contains no more than 128 characters, including letters, numbers, or underscores (_), and must start with a letter or number.
2. In **Description**, enter description about the Hive table as required.
3. In **Input File or Location**, click  and select a Hive table file from HDFS. The file is used to store new data of the Hive table.
If the file is not stored in HDFS, click **Upload a file** to upload the file from the local directory to HDFS. Multiple files can be simultaneously uploaded. The files cannot be empty.
4. If you need to import the data in the file to the Hive table, select **Import data as Load method**. By default, **Import data** is selected.

If you select **Create External Table**, a Hive external table is created.

 **NOTE**

If you select **Create External Table**, set **Input File or Location** to a path.

If you select **Leave Empty**, an empty Hive table is created.

5. Click **Next**.

Step 4 Set a delimiter.


1. In **Delimiter**, select one.

If your desired delimiter is not in the list, select **Other..** and enter a delimiter.

2. Click **Preview** to preview data processing.

3. Click **Next**.


Step 5 Define a column.

1. If you click  on the right side of **Use first row as column names**, the first row of data in the file is used as a column name. If you do not click it, the first row of data is not used as the column name.

2. In **Column name**, set a name for each column.

A Hive table name contains no more than 128 characters, including letters, numbers, or underscores (_), and must start with a letter or number.

 **NOTE**

You can rename columns in batches by clicking  on the right side of **Bulk edit column names**. Enter all column names and separate them by commas (,).

3. In **Column Type**, select a type for each column.

Step 6 Click **Create Table** to create the table. Wait for Hue to display information about the Hive table.

----End

Manually Creating a Hive Table

Step 1 Access **Metastore Manager** and select a database in **Databases**.

The default database is **default**.

Step 2 Click . The **Create a new table manually** page is displayed.

Step 3 Set a table name.

1. In **Table Name**, enter a Hive table name.

A Hive table name contains no more than 128 characters, including letters, numbers, or underscores (_), and must start with a letter or number.

2. In **Description**, enter description about the Hive table as required.

3. Click **Next**.

Step 4 Select a data storage format.

- If data needs to be separated by delimiters, select **Delimited** and perform [Step 5](#).

- If data needs to be stored in serialization format, select **SerDe** and perform [Step 6](#).

Step 5 Set a delimiter.

1. In **Field terminator**, set a column delimiter.
If your desired delimiter is not in the list, select **Other..** and enter a delimiter.
2. In **Collection terminator**, set a delimiter to separate the data set of columns of the **array** type in Hive. For example, the type of a column is array. A value needs to store **employee** and **manager**. The user specifies a colon (:) as the delimiter. Therefore, the final value is **employee:manager**.
3. In **Map key terminator**, set a delimiter to separate the data set of columns of the **map** type in Hive. For example, the type of a column is map. A value needs to store **home** of **aaa** and **company** of **bbb**. The user defines | as the delimiter. Therefore, the final value is **home|aaa:company|bbb**.
4. Click **Next** and perform [Step 7](#).

Step 6 Set serialization properties.

1. In **SerDe Name**, enter the class name of the serialization format:
org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
Users can expand Hive to support more customized serialization classes.
2. In **Serde properties**, enter the value of the serialization format:
"field.delim"="," "collection.delim"=":" "mapkey.delim"="|"
3. Click **Next** and perform [Step 7](#).

Step 7 Select a data table format and click **Next**.

- **TextFile**: indicates that data is stored in text files.
- **SequenceFile**: indicates that data is stored in binary files.
- **InputFormat**: indicates that data in files is used in the customized input and output formats.
Users can expand Hive to support more customized formatting classes.
 - a. In **InputFormat Class**, enter the class used by input data:
org.apache.hadoop.hive.ql.io.RCFileInputFormat
 - b. In **OutputFormat Class**, enter the class used by output data:
org.apache.hadoop.hive.ql.io.RCFileOutputFormat

Step 8 Select a file storage location and click **Next**.

Use default location is selected by default. If you want to customize a storage location, deselect the default value and specify a file storage location in **External location** by clicking ******.

Step 9 Set columns of the Hive table.

1. In **Column name**, set a column name.
A Hive table name contains no more than 128 characters, including letters, numbers, or underscores (_), and must start with a letter or number.
2. In **Column type**, select a type for each column.
Click **Add a column** to add a new column.

3. Click **Add a partition** to add a new partition for the Hive table to improve the query efficiency.

Step 10 Click **Create Table** to create a new table. Wait for Hue to display information about the Hive table.

----End

Managing the Hive Table

Step 1 Access **Metastore Manager** and select a database in **Databases**. All tables in the database are displayed on the page.

The default database is **default**.

Step 2 Click a table name in the database to view table details.

The following operations are supported: importing data, browsing data,, or viewing file storage location. When viewing all tables in the database, you can select tables and perform the following operations such as viewing tables and browsing data.

 **CAUTION**

The Hue page is used to view and analyze data such as files and tables. Do not perform high-risk management operations such as deleting objects on the page. If an operation is required, you are advised to perform the operation on each component after confirming that the operation has no impact on services. For example, you can use the HDFS client to perform operations on HDFS files and use the Hive client to perform operations on Hive tables.

----End

11.6 Using File Browser on the Hue Web UI

Scenario

Users can use the Hue web UI to manage files in HDFS in a cluster.

 **CAUTION**

The Hue page is used to view and analyze data such as files and tables. Do not perform high-risk management operations such as deleting objects on the page. If an operation is required, you are advised to perform the operation on each component after confirming that the operation has no impact on services. For example, you can use the HDFS client to perform operations on HDFS files and use the Hive client to perform operations on Hive tables.

File Browser (File Browser)

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 Click . The **File Browser** page is displayed.

You can view the home directory of the current login user.

On the **File Browser** page, the following information about subdirectories for files in the directory is displayed.

Table 11-2 HDFS file attributes


Attribute	Description
Name	Name of a directory or file
Size	File size
User	Owner of a directory or file
Group	Group of a directory or file
Permissions	Permission of a directory or file
Date	Time when a directory or file is created

Step 3 In the search box, enter a keyword. The system automatically searches directories or files in the current directory.

Step 4 Clear the search criteria. The system displays all directories or files.

----End

Performing Actions

Step 1 Click  and select one or more directories or files.

Step 2 Click **Actions**. On the menu that is displayed, select an operation.

- **Rename**: renames a directory or file.
- **Move**: moves a file. In **Move to**, select a new directory and click **Move**.
- **Copy**: copies the selected files or directories.
- **Change permissions**: changes permission to access the selected directory or file.
 - You can grant the owner, the group, or other users with the **Read**, **Write**, and **Execute** permissions.
 - **Sticky**: indicates that only HDFS administrators, directory owners, and file owners can move files in the directory.
 - **Recursive**: indicates that permission is granted to subdirectories recursively.
- **Storage policies**: indicates the policies for storing files or directories in HDFS.

- **Summary:** indicates that you can view HDFS storage information about the selected file or directory.

----End

Accessing Other Directories

Step 1 Click the directory name, type a full path you want to access, for example, **/mr-history/tmp**, and press **Enter**.

The current user must have permission to access other directories.

Step 2 Click **Home** to go to the home directory.

Step 3 Click **History**. The history records of directory access are displayed and the directories can be accessed again.

Step 4 Click **Trash** to access the recycle bin of the current directory.

Click **Empty Trash** to clean up the recycle bin.

----End

Uploading User Files

Step 1 Click  and click **Upload**.

Step 2 Select an operation.

- **Files:** uploads user files to the current user.
- **Zip/Tgz/Bz2 file:** uploads a compressed file. In the dialog box that is displayed, click **Select ZIP, TGZ or BZ2 files** to select the compressed file to be uploaded. The system automatically decompresses the file in HDFS. Compressed files in **ZIP, TGZ, and BZ2** formats are supported.

----End

Creating a New File or Directory

Step 1 Click  and click **New**.

Step 2 Select an operation.

- **File:** creates a file. Enter a file name and click **Create**.
- **Directory:** creates a directory. Enter a directory name and click **Create**.

----End


Storage Policy Definition and Usage

NOTE

If the value of Hue parameter **fs_defaultFS** is set to **viewfs://ClusterX**, the big data storage policy cannot be enabled.

Step 1 Log in to MRS Manager.

Step 2 On MRS Manager, choose **System > Permission > Manage Role > Create Role**.

1. Set **Role Name**.
 2. Choose **Configure Resource Permission > Hue**, select **Storage Policy Admin**, and click **OK** to grant the storage policy administrator permission to the role.
- Step 3** Choose **System > Permission > Manage User Group > Create User Group**, set **Group Name**, and click **Select and Add Role** next to **Role**. On the displayed page, select the created role and click **OK** to add the role to the group.
- Step 4** Choose **System > Permission > Manage User > Create User**.
1. Specify the **Username** of a user who can log in to the Hue web UI and has the **Storage Policy Admin** permission.
 2. Set **User Type** to **Human-machine**.
 3. Set **Password** and **Confirm Password** for logging in to the Hue web UI.
 4. Click **Select and Join User Group** next to **User Group**. On the page that is displayed, select the created user group, **supergroup**, **hadoop**, and **hive**, and click **OK**.
 5. Set **Primary Group** to **hive**.
 6. Click **Select and Add Role** on the right of **Assign Rights by Role**. On the Select snf page that is displayed, select the newly created role and the **System_administrator** role, and click **OK**.
 7. Click **OK**. The user is added successfully.
- Step 5** Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).
- Step 6** Click  in the upper right corner.
- Step 7** Select the check box of the directory and click **Action** on the upper part of the page. Then select **Storage policies**.
- Step 8** In the dialog box that is displayed, set a new storage policy and click **OK**.
- End

11.7 Using Job Browser on the Hue Web UI

Scenario

You can use the Hue web UI to query all jobs in the cluster.

Accessing Job Browser

- Step 1** Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).
- Step 2** Click **Job Browser**.


View the jobs in the cluster.

NOTE

The number on **Job Browser** indicates the total number of jobs in the cluster.

Job Browser displays the following job information.

Table 11-3 MRS job attributes

Attribute	Description
Logs	Log information. If a job has logs,  is displayed.
ID	Job ID, which is generated by the system automatically.
Name	Job name
Application Type	Job type
Status	Job status. Possible values are RUNNING , SUCCEEDED , FAILED , and KILLED .
User	User who starts the job
Maps	Map progress
Reduces	Reduce progress
Queue	Yarn queue used for job running
Priority	Job running priority
Duration	Job running duration
Submitted	Time when the job is submitted to the MRS cluster

 **NOTE**

If the MRS cluster has Spark, the **Spark-JDBCServer** job is started by default to execute tasks.

----End

Searching for Jobs

Step 1 Enter keywords in **Username** or **Text** on the **Job Browser** page to search for the desired jobs.

Step 2 Clear the search criteria. The system displays all jobs.


----End

Querying Job Details

Step 1 In the job list on the **Job Browser** page, click the row that contains the desired job to view details.

Step 2 On the **Metadata** tab page, you can view the metadata of the job.

 **NOTE**

You can click  to open job running logs.

----End

12 Using Hue (MRS 3.x or Later)

12.1 Using Hue from Scratch


Hue aggregates interfaces which interact with most Apache Hadoop components and enables you to use Hadoop components with ease on a web UI. You can operate components such as HDFS, Hive, HBase, Yarn, MapReduce, Oozie, and Spark SQL on the Hue web UI.

Prerequisites

You have installed Hue, and the Kerberos authentication cluster is in the running state.

Procedure

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 In the navigation tree on the left, click the editor icon  and choose **Hive**.

Step 3 Select a Hive database from the **Database** drop-down list box. The default database is **default**.


The system displays all available tables. You can enter a keyword of the table name to search for the desired table.

Step 4 Click the desired table name. All columns in the table are displayed.

Step 5 Enter the HiveQL statements in the area for editing.

```
create table hue_table(id int,name string,company string) row format  
delimited fields terminated by ',' stored as textfile;
```

Step 6 Click  to execute the HiveQL statements.

Step 7 In the command text box, enter **show tables;** and click . Check whether the **hue_table** table created in [Step 5](#) exists in the **Result**.

----End

12.2 Accessing the Hue Web UI

Scenario

After Hue is installed in an MRS cluster, users can use Hadoop-related components on the Hue web UI.

This section describes how to open the Hue web UI on the MRS cluster.

NOTE

To access the Hue web UI, you are advised to use a browser that is compatible with the Hue WebUI, for example, Google Chrome 50. The Internet Explorer may be incompatible with the Hue web UI.

Impact on the System

Site trust must be added to the browser when you access Manager and Hue web UI for the first time. Otherwise, the Hue web UI cannot be accessed.

Prerequisites

When Kerberos authentication is enabled, the MRS cluster administrator has assigned the permission for using Hive to the user. For details, see [Creating a User](#). For example, create a human-machine user named **hueuser**, add the user to user groups **hive** (the primary group), **hadoop**, **supergroup**, and **System_administrator**, and assign the **System_administrator** role.

This user is used to log in to Manager.

Procedure




Step 1 Log in to the service page.






For versions earlier than MRS 3.x, click the cluster name on the MRS console and choose **Components > Hue**.

For MRS 3.x or later, log in to FusionInsight Manager (for details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#)) and choose **Cluster > Services > Hue**.

Step 2 On the right of **Hue WebUI**, click the link to open the Hue web UI.

Hue WebUI provides the following functions:

- Click  to execute query statements of Hive and SparkSQL as well as Notebook code. Make sure that Hive and Spark2x have been installed in the MRS cluster before this operation.
- Click  to submit workflow tasks, scheduled tasks, and bundle tasks.
- Click  to view, import, and export tasks on the Hue web UI, such as workflow tasks, scheduled tasks, and bundle tasks.

- Click  to manage metadata in Hive and SparkSQL. Make sure that Hive and Spark2x have been installed in the MRS cluster before this operation.
- Click  to view the directories and files in HDFS. Make sure that HDFS has been installed in the MRS cluster before this operation.
- Click  to view all jobs in the MRS cluster. Make sure that Yarn has been installed in the MRS cluster before this operation.
- Use  to create or query HBase tables. Make sure that the HBase component has been installed in the MRS cluster and the Thrift1Server instance has been added before this operation.
- Use  to import data that is in the CSV or TXT format.

 **NOTE**

- When you log in to the Hue web UI as user **hueuser** for the first time, you need to change the password.
- After obtaining the URL for accessing the Hue web UI, you can give the URL to other users who cannot access MRS Manager for accessing the Hue web UI.
- If you perform operations on the Hue WebUI only but not on Manager, you must enter the password of the current login user when accessing Manager again.

----End

12.3 Hue Common Parameters

Page Access

Go to the **All Configurations** page of the Hue service by referring to [Modifying Cluster Service Configuration Parameters](#).

Parameter Description

For details about Hue common parameters, see [Table 12-1](#).

Table 12-1 Hue common parameters

Configuration	Description	Default Value	Value Range
HANDLER_ACCESSLOG_LEVEL	Hue access log level.	DEBUG	<ul style="list-style-type: none"> • ERROR • WARN • INFO • DEBUG

Configuration	Description	Default Value	Value Range
HANDLER_AUDITSL OG_LEVEL	Hue audit log level.	DEBUG	<ul style="list-style-type: none"> • ERROR • WARN • INFO • DEBUG
HANDLER_ERRORLO G_LEVEL	Hue error log level.	ERROR	<ul style="list-style-type: none"> • ERROR • WARN • INFO • DEBUG
HANDLER_LOGFILE_L EVEL	Hue run log level.	INFO	<ul style="list-style-type: none"> • ERROR • WARN • INFO • DEBUG
HANDLER_LOGFILE_ MAXBACKUPINDEX	Maximum number of Hue log files.	20	1 to 999
HANDLER_LOGFILE_S IZE	Maximum size of a Hue log file.	5 MB	-


12.4 Using HiveQL Editor on the Hue Web UI

Scenario

Users can use the Hue web UI to execute HiveQL statements in an MRS cluster.

Access Editor

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 In the navigation tree on the left, click  and choose **Hive**. The **Hive** page is displayed.

Hive supports the following functions:

- Executes and manages HiveQL statements.
- Views the HiveQL statements saved by the current user in **Saved Queries**.
- Queries HiveQL statements executed by the current user in **Query History**.


----End

Executing HiveQL Statements


Step 1 Select a Hive database from the **Database** drop-down list box. The default database is **default**.

The system displays all available tables. You can enter a keyword of the table name to search for the desired table.





Step 2 Click the desired table name. All columns in the table are displayed.

Move the cursor to the row where the table or column is located and click . Column details are displayed.

Step 3 Enter the query statements in the area for editing HiveQL statements.

Step 4 Click  to execute the HiveQL statements.

 **NOTE**

- If you want to use the entered HiveQL statements again, click  to save them.
- Advanced query configuration:
Click  in the upper right corner to configure information such as files, functions, and settings.
- Viewing the information of shortcut keys:
Click  in the upper right corner to view the syntax and keyboard shortcut information.
- To delete an entered HiveQL statement, click the triangle next to  and select **Clear**.
- Viewing history:
Click **Query History** to view the HiveQL running status. You can view the history of all the statements or only the saved statements. If many historical records exist, you can enter keywords in the text box to search for desired records.

----End

Viewing Execution Results

Step 1 View the execution results below the execution area on **Hive**. The **Query History** tab page is displayed by default.

Step 2 Click a result to view the execution result of the executed statement.

----End

Managing Query Statements

Step 1 Click **Saved Queries**.

Step 2 Click a saved statement. The system automatically adds the statement to the editing area.


----End

Modifying the Session Configuration of the Hue Editor

Step 1 On the editor page, click .


Step 2 Click  on the right of **Files**, and then click  to select files.

You can click  next to **Files** to add a file resource.

Step 3 In the **Functions**  area, enter a user-defined name and the class name of the function.

You can click  next to **Functions** to add a customized function.

Step 4 In the **Settings**  area, enter the Hive parameter name in the **Key**, and value in **Value**. The current Hive session connects to Hive based on the customized configuration.

You can click  to add a parameter.

----End

12.5 Using the SparkSql Editor on the Hue Web UI

Scenario

You can use Hue to execute SparkSql statements in a cluster on a graphical user interface (GUI).

Configuring Spark2x

Before using the SparkSql editor, you need to modify the Spark2x configuration.

Step 1 Go to the Spark2x configuration page. For details, see [Modifying Cluster Service Configuration Parameters](#).

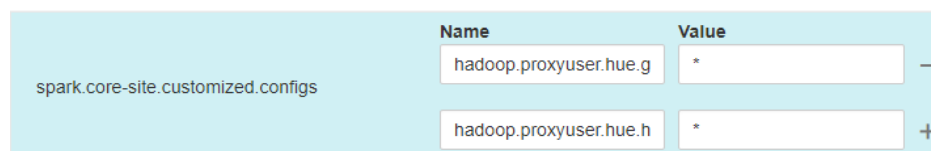
Step 2 Set the Spark2x multi-instance mode. Search for and modify the following parameters of the Spark2x service:

Parameter	Value
spark.thriftserver.proxy.enabled	false
spark.scheduler.allocation.file	#{conf_dir}/fairscheduler.xml

Step 3 Go to the JDBCServer2x customization page and add the following customized items to the **spark.core-site.customized.configs** parameter:

Set **hadoop.proxyuser.hue.groups** to *****.

Set **hadoop.proxyuser.hue.hosts** to *****.



Name	Value
hadoop.proxyuser.hue.g	*
hadoop.proxyuser.hue.h	*

Step 4 Save the configuration and restart the meta and Spark2x services.

----End

Accessing the Editor

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 In the navigation tree on the left, click  and choose **SparkSql**. The **SparkSql** page is displayed.

SparkSql supports the following functions:

- Executes and manages SparkSql statements.
- Views the SparkSql statements saved by the current user in **Saved Queries**.
- Queries SparkSql statements executed by the current user in **Query History**.


----End

Executing SparkSql Statements


Step 1 Select a SparkSql database from the **Database** drop-down list box. The default database is **default**.

The system displays all available tables. You can enter a keyword of the table name to search for the desired table.

Step 2 Click the desired table name. All columns in the table are displayed.






Move the cursor to the row of the table and click . Column details are displayed.

Step 3 In the SparkSql statement editing area, enter the query statement.

Click the triangle next to  and select **Explain**. The editor checks the syntax and execution plan of the entered statements. If the statements have syntax errors, the editor reports **Error while compiling statement**.

Step 4 Click  to execute the SparkSql statement.

 **NOTE**

- If you want to use the entered SparkSql statements again, click  to save them.
- Advanced query configuration:
Click  in the upper right corner to configure information such as files, functions, and settings.
- Viewing the information of shortcut keys:
Click  in the upper right corner to view the syntax and keyboard shortcut information.
- To format the SparkSql statement, click the triangle next to  and select **Format**.
- To delete an entered SparkSql statement, click the triangle next to  and select **Clear**.
- Viewing historical records:
Click **Query History** to view the SparkSql running status. You can view the history of all the statements or only the saved statements. If many historical records exist, you can enter keywords in the text box to search for desired records.

----End

Viewing Execution Results

Step 1 View the execution results below the execution area on **SparkSql**. The **Query History** tab page is displayed by default.

Step 2 Click a result to view the execution result of the executed statement.

----End

Managing Query Statements

Step 1 Click **Saved Queries**.

Step 2 Click a saved statement. The system automatically adds the statement to the editing area.

----End

12.6 Using the Metadata Browser on the Hue Web UI


Scenario

Users can use the Hue web UI to manage Hive metadata in an MRS cluster.

Using Metadata Manager

Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

- Viewing metadata of Hive tables

Click  in the navigation tree on the left and click a table name. The metadata of the Hive table is displayed.

- Managing metadata of Hive tables
On the metadata information page of a Hive table:
 - Click **Import** in the upper right corner to import data.
 - Click **Overview** to view the location of the table file in the **PROPERTIES** field.
 - Click **Sample** to browse data.
- Managing Hive metadata tables
Click **+** in the left list to create a table based on the uploaded file in the database. You can also manually create a table.

 **CAUTION**

The Hue page is used to view and analyze data such as files and tables. Do not perform high-risk management operations such as deleting objects on the page. If an operation is required, you are advised to perform the operation on each component after confirming that the operation has no impact on services. For example, you can use the HDFS client to perform operations on HDFS files and use the Hive client to perform operations on Hive tables.

12.7 Using File Browser on the Hue Web UI

Scenario

Users can use the Hue web UI to manage files in HDFS.

 **CAUTION**

The Hue page is used to view and analyze data such as files and tables. Do not perform high-risk management operations such as deleting objects on the page. If an operation is required, you are advised to perform the operation on each component after confirming that the operation has no impact on services. For example, you can use the HDFS client to perform operations on HDFS files and use the Hive client to perform operations on Hive tables.

Accessing File Browser

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 In the left navigation pane, click . The **File Browser** page is displayed.

By default, the homepage of **File Browser** is the home directory of the current login user. On the displayed page, the following information about subdirectories for files in the directory is displayed:

Table 12-2 HDFS file attributes

Attribute	Description
Name	Name of a directory or file
Size	File size
User	Owner of a directory or file
Group	Group of a directory or file
Permission	Permission of a directory or file
Date	Time when a directory or file is created

Step 3 In the search box, enter a keyword. The system automatically searches directories or files in the current directory.

Step 4 Clear the search criteria. The system displays all directories or files.

----End

Performing Actions

Step 1 On the **File Browser** page, select one or more directories or files.

Step 2 Click **Actions**. On the menu that is displayed, select an operation.

- **Rename:** renames a directory or file.
- **Move:** moves a file. In **Move to**, select a new directory and click **Move**.
- **Copy:** copies the selected files or directories.
- **Change permissions:** changes permission to access the selected directory or file.
 - You can grant the owner, the group, or other users with the **Read, Write,** and **Execute** permissions.
 - **Sticky:** indicates that only HDFS administrators, directory owners, and file owners can move files in the directory.
 - **Recursive:** indicates that permission is granted to subdirectories recursively.
- **Storage policies:** indicates the policies for storing files or directories in HDFS.
- **Summary:** indicates that the HDFS storage information about the selected file or directory can be viewed.

----End

Uploading User Files

Step 1 On the **File Browser** page, click **Upload**.

Step 2 In the displayed dialog box for uploading files, click **Select files** or drag the file to the dialog box.

----End

Creating a New File or Directory

Step 1 On the **File Browser** page, click **New**.

Step 2 Select an operation.

- **File**: creates a file. Enter a file name and click **Create**.
- **Directory**: creates a directory. Enter a directory name and click **Create**.

----End

Storage Policy Definition and Usage

NOTE

If the value of Hue parameter **fs_defaultFS** is set to **viewfs://ClusterX**, the big data storage policy cannot be enabled.

Step 1 Log in to FusionInsight Manager.

Step 2 On FusionInsight Manager, choose **System > Permission > Manage Role > Create Role**.

1. Set **Role Name**.
2. In the **Configure Resource Permission** area, choose *Name of the desired cluster* > **Hue**, select **Storage Policy Admin**, and click **OK**. Then, grant the permission to the role.

Step 3 Choose **System > Permission > User Group > Create User Group**. Set **Group Name** and click **Select and Add Role** next to **Role**. On the displayed page, select the role created in [Step 2](#) and click **OK** to add the role to the group.

Step 4 Choose **System > Permission > User > Create**.

1. **Username**: Enter the name of the user to be added.
2. Set **User Type** to **Human-machine**.
3. Set **Password** and **Confirm Password** for logging in to the Hue web UI.
4. Click **Add** next to **User Group**. On the page that is displayed, select the user group created in [Step 3](#), **supergroup**, **hadoop**, and **hive**, and click **OK**.
5. Set **Primary Group** to **hive**.
6. Click **Add** on the right of **Role**. On the page that is displayed, select the role created in [Step 2](#) and **System_administrator** role, and click **OK**.
7. Click **OK**. The user is added successfully.

Step 5 Access the Hue web UI as the created user. For details, see [Accessing the Hue Web UI](#).

Step 6 In the left navigation tree, click . The **File Browser** page is displayed.

Step 7 Select the check box of the directory and click **Actions** on the top of the page. Choose **Storage policies**.

Step 8 In the dialog box that is displayed, set a new storage policy and click **OK**.

----End

12.8 Using Job Browser on the Hue Web UI

Scenario

Users can use the Hue web UI to query all jobs in an MRS cluster.

Accessing Job Browser

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 Click .

View the jobs in the current cluster.

NOTE

The number on **Job Browser** indicates the total number of jobs in the cluster.

Job Browser displays the following job information:

Table 12-3 MRS job attributes

Attribute	Description
Name	Job name
User	User who starts a job
Type	Job type
Status	Job status, including Succeeded , Running , and Failed .
Progress	Job running progress
Group	Group to which a job belongs
Start	Start time of a job
Duration	Job running duration
Id	Job ID, which is generated by the system automatically.

NOTE

If the MRS cluster has Spark, the **Spark-JDBCServer** job is started by default to execute tasks.

----End

Searching for Jobs

Step 1 In the search box of **Job Browser**, enter the specified character. The system automatically searches for all jobs that contain the keyword by ID, name, or user.

Step 2 Clear the search criteria. The system displays all jobs.

----End

Querying Job Details

Step 1 In the job list on the **Job Browser** page, click the row that contains the desired job to view details.

Step 2 On the **Metadata** tab page, you can view the metadata of the job.

 **NOTE**

You can click **Log** to open the job running log.

----End

12.9 Using HBase on the Hue Web UI

Scenario

You can use Hue to create or query HBase tables in a cluster and run tasks on the Hue web UI.

Make sure that the HBase component has been installed in the MRS cluster and the Thrift1Server instance has been added before this operation.

Accessing Job Browser

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 Click HBase . The **HBase Browser** page is displayed.

----End

Creating an HBase Table

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 Click HBase . The **HBase Browser** page is displayed.

Step 3 Click **New Table** on the right, enter the table name and column family parameters, and click **Submit**.

----End

Querying Data in an HBase Table

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 Click HBase . The **HBase Browser** page is displayed.

Step 3 Click the HBase table to be queried. Then, click the key value next to search box in the upper part, and query the HBase table.

----End

12.10 Typical Scenarios

12.10.1 HDFS on Hue


Hue provides the file browser function for users to use HDFS in GUI mode.

 **CAUTION**

The Hue page is used to view and analyze data such as files and tables. Do not perform high-risk management operations such as deleting objects on the page. If an operation is required, you are advised to perform the operation on each component after confirming that the operation has no impact on services. For example, you can use the HDFS client to perform operations on HDFS files and use the Hive client to perform operations on Hive tables.

How to Use File Browser

Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Click . The **File Browser** page is displayed. You can perform the following operations:

- Viewing files or directories
By default, the directory and files in the directory of the login user are displayed. You can view **Name**, **Size**, **User**, **Group**, **Permission**, and **Date**.
Click a file name to view the text information or binary data in the text file. The file content can be edited.
If there are a large number of files and directories, you can enter keywords in the search text box to search for specific files or directories.
- Creating files or directories
Click **New** in the upper right corner. Choose **File** to create the file. Choose **Directory** to create a directory.
- Managing files or directories
Select the check box of a file or director, and click **Actions**. In the displayed menu, choose **Rename**, **Move**, **Copy**, and **Change permissions** to rename, move, copy, or change the file or directory permissions.
- Uploading files
Click **Upload** in the upper right corner and click **Select files** or drag the file to the window.

How to Use Storage Policies

NOTE

If the value of Hue parameter **fs_defaultFS** is set to **viewfs://ClusterX**, the big data storage policy cannot be enabled.

Storage policies on the Hue web UI are classified into the following two types:

- **Static Storage Policies**
 - a. **Current storage policy**
According to the access frequency and importance of documents in HDFS, specify a storage policy for an HDFS directory, such as ONE_SSD or ALL_SSD. The files in this directory can be migrated to the storage media.
 - b. **Current EC policy**
Specify EC policies for HDFS directories, such as RS-10-4-1024k and RS-3-2-1024k. For details, see [Configuring HDFS EC Storage](#).
- **Dynamic Storage Policies**
Set rules for an HDFS directory. The system can automatically change the storage policy, the number of file copies, migrate the file directory.
Before configuring a dynamic storage policy on the Hue WebUI, you must set the CRON expressions for cold and hot data migration and start automatic cold and hot data migration on Manager.

Operations:

Change the value of **dfs.auto.data.mover.cron.expression** for NameNode of the HDFS service. For details, see [Modifying Cluster Service Configuration Parameters](#).

NOTE

- **dfs.auto.data.mover.cron.expression** indicates the CRON expression for checking whether HDFS data meets the dynamic storage policy rule. It is used to control the start time of data migration. The default value is **0******, indicating that the detection is performed on the hour. When the dynamic storage policy rule is met, cold and hot data migration tasks are executed on the hour.
- The default value of **dfs.auto.data.mover.enable** is **false**. This parameter value is valid only when **dfs.auto.data.mover.enable** is set to **true**.

Table 12-4 describes the expression for modifying this parameter. * indicates consecutive time segments.

Table 12-4 Parameters in the execution expression

Column	Description
1	Minute. The value ranges from 0 to 59.
2	Hour. The value ranges from 0 to 23.
3	Date. The value ranges from 1 to 31.
4	Month. The value ranges from 1 to 12.

Column	Description
5	Week. The value ranges from 0 to 6. 0 indicates Sunday.

To set storage policies on the web UI, perform the following operations:


- Step 1** Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
- Step 2** On FusionInsight Manager, choose **System > Permission > Role > Create Role**.
1. Set **Role Name**.
 2. In the **Configure Resource Permission** area, choose *Name of the desired cluster* > **Hue**, select **Storage Policy Admin**, and click **OK**. Then, grant the permission to the role.
- Step 3** Choose **System > Permission > User Group > Create User Group**. Set **Group Name**, and click **Add** next to **Role**. On the displayed page, select the created role, click **OK** to add the role to the group, and click **OK**.
- Step 4** Choose **System > Permission > User > Create**.
1. **Username**: Enter the name of the user to be added.
 2. Set **User Type** to **Human-machine**.
 3. Set **Password** and **Confirm Password** for logging in to the Hue web UI.
 4. Click **Add** next to **User Group**. On the page that is displayed, select the created user group in [Step 3](#), **supergroup**, **hadoop**, and **hive**, and click **OK**.
 5. Set **Primary Group** to **hive**.
 6. Click **Add** next to **Role**. On the page that is displayed, select the created role in [Step 2](#) and the **System_administrator** role, and click **OK**.
 7. Click **OK**. The user is added successfully.
- Step 5** Access the Hue web UI as the created user. For details, see [Accessing the Hue Web UI](#).
- Step 6** In the left navigation pane, click . The **File Browser** page is displayed.
- Step 7** Select the check box of a directory and choose **Action** on the top of the page. Choose **Storage policies**.
- Step 8** In the dialog box that is displayed, set a new storage policy and click **OK**.
- On the **Static Storage Policy** tab page, select a policy from the **New EC Policy** drop-down list box and click **Save**. For details about the parameters, see [Configuring HDFS EC Storage](#).
 - On the **Dynamic Storage Policy** tab page, you can create, delete, or modify a dynamic storage policy. [Table 12-5](#) describes the parameters.

Table 12-5 Parameters of the dynamic storage policy

Category	Parameter	Description
Rule	Last Access to File	Indicates the time when the file is last accessed.
	Last File Modification	Indicates the time when the file is last modified.
Operation	Change Number of Copies	Indicates the number of file copies.
	Modify Storage Policy	Indicates that you can modify storage policies to the following: HOT, WARM, COLD, ONE_SSD, and ALL_SSD.
	Move to Directory	Indicates that you can move the file to another directory.

 **NOTE**

- You need to consider whether the rules conflict with each other and whether the rules damage the system when setting rules.
- When a directory is configured with multiple rules and operations, the rule that is triggered first is located at the bottom of the rule/operation list, and the rules that are triggered later are placed from bottom to top to prevent repeated operations.
- The system checks whether the files under the directory specified by the dynamic storage policy meet the rules on an hourly basis. If the files meet the rules, the execution is triggered. Execution logs are recorded in the `/var/log/Bigdata/hdfs/nn/hadoop.log` directory of the active NameNode.

----End

Typical Scenarios

On the Hue page, view and edit HDFS files in text or binary mode as follows:

Viewing a File

Step 1 Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

Step 2 In the left navigation pane, click . The **File Browser** page is displayed.

Step 3 Click the name of the file to be viewed.

Step 4 Click **View as binary** to switch from the text mode to the binary mode. Click **View as file** to switch from the binary mode to the text mode.

Editing a file

Step 5 Click **Edit File**. The file content can be edited.

Step 6 Click **Save** or **Save As** to save the file.

----End

12.10.2 Hive on Hue


Hue provides the Hive GUI management function so that users can query Hive data in GUI mode.


How to Use Query Editor

Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).


In the navigation tree on the left, click  and choose **Hive**. The **Hive** page is displayed.

- Running Hive HQL statements


Select the target database on the left. You can also click **default**  in the upper right corner and enter the target database name to search for the target database.

Enter a Hive HQL statement in the text box and click  or press **Ctrl+Enter** to run the HQL statement. The execution result is displayed on the **Result** tab page.

- Analyzing Hive HQL statements

Select the target database on the left, enter the Hive HQL statement in the text box, and click  to compile the HQL statement and check whether the statement is correct. The execution result is displayed under the text editing box.

- Saving HQL statements

Enter the Hive HQL statement in the text box, click  in the upper right corner, and enter the name and description. You can view the saved statements on the **Saved Queries** tab page.

- Viewing historical records

Click **Query History** to view the HQL running status. You can view the history of all the statements or only the saved statements. If many historical records exist, you can enter keywords in the text box to search for desired records.

- Configuring advanced query

Click  in the upper right corner to configure the file, function, and settings.

- Viewing the information of shortcut keys


Click  in the upper right corner to view information about all shortcut keys.

How to Use Metadata Browser

Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

- Viewing metadata of Hive tables




Click  in the navigation tree on the left and click a table name. The metadata of the Hive table is displayed.

- Managing metadata of Hive tables

On the metadata information page of a Hive table:

- Click **Import** in the upper right corner to import data.
- Click **Overview** to view the location of the table file in the **PROPERTIES** field.
- Click **Sample** to browse data.

- Managing Hive metadata tables


Click  in the left list to create a table based on the uploaded file in the database. You can also manually create a table.

 **CAUTION**

The Hue page is used to view and analyze data such as files and tables. Do not perform high-risk management operations such as deleting objects on the page. If an operation is required, you are advised to perform the operation on each component after confirming that the operation has no impact on services. For example, you can use the HDFS client to perform operations on HDFS files and use the Hive client to perform operations on Hive tables.


Typical Scenarios

On the Hue page, create a Hive table as follows:

Step 1 Click  at the upper left corner of Hue web UI and select the Hive instance to be operated to enter the Hive command execution page.

Step 2 Enter an HQL statement in the command input box, for example:

```
create table hue_table(id int,name string,company string) row format  
delimited fields terminated by ',' stored as textfile;
```

Click  to execute the HQL statements.

Step 3 Enter the following command in the command input box:

```
show tables;
```

Click  to view the created table **hue_table** in **Result**.

----End

12.10.3 Oozie on Hue


Hue provides the Oozie job manager function, in this case, you can use Oozie in GUI mode.

 **CAUTION**

The Hue page is used to view and analyze data such as files and tables. Do not perform high-risk management operations such as deleting objects on the page. If an operation is required, you are advised to perform the operation on each component after confirming that the operation has no impact on services. For example, you can use the HDFS client to perform operations on HDFS files and use the Hive client to perform operations on Hive tables.

How to Use Oozie Job Designer

Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

In the navigation tree on the left, click  and choose **Workflow**.

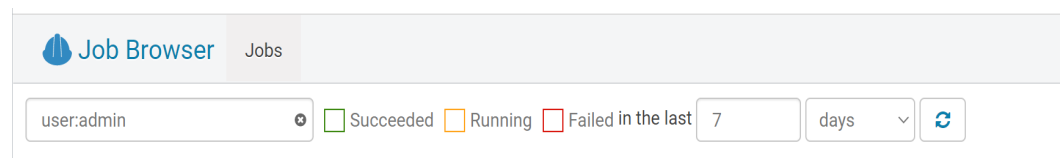
The job designer allows users to create MapReduce, Java, Streaming, Fs, SSH, Shell and DistCp jobs.

How to Use Dashboard

Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).


Click **Jobs** in the upper right corner. The **Job Browser** page is displayed.

View the running status of the Workflow, Coordinator, and Bundles jobs.



How to Use Editor

Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

In the navigation tree on the left, click  and choose **Workflow**.

Workflows, Schedule, and Bundle tasks can be created. Existing applications can be submitted for running, shared, copied, and exported.

- Each Workflow can contain one or more jobs to form a complete workflow for a specified service.
When creating a Workflow, you can design jobs in the Hue editor and add the jobs to the Workflow.
- Each Schedule can define a time trigger to periodically execute a specified Workflow. One time trigger cannot execute multiple Workflows.
- Each Bundles can define a set to execute multiple Schedules so that different Workflows can be executed in batches.

12.11 Hue Log Overview

Log Description

Log paths: The default paths of Hue logs are `/var/log/Bigdata/hue` (for storing run logs) and `/var/log/Bigdata/audit/hue` (for storing audit logs).

Log archive rules: The automatic compression and archiving function of the Hue logs is enabled. By default, when the size of a log file (`access.log`, `error.log`, `runcpserver.log`, or `hue-audits.log`) exceeds 5 MB, logs are automatically compressed. A maximum of 20 latest compressed files are reserved. The number of compressed files and compression threshold can be configured.

Table 12-6 Hue log list

Type	Log File Name	Description
Run log	access.log	Access log file
	error.log	Error log file
	gsdb_check.log	Log file of the GaussDB check information
	kt_renewer.log	Log file of Kerberos authentication
	kt_renewer.out.log	Log file of the abnormal Kerberos authentication logs
	runcpserver.log	Log file of operation records
	runcpserver.out.log	Log file of process running exceptions
	supervisor.log	Log file of process startup
	supervisor.out.log	Log file of process startup exceptions
	dbDetail.log	Log file of database initialization
	initSecurityDetail.log	Download initialization log file of the Keytab file
	postinstallDetail.log	Work log file generated after the Hue service is installed
	prestartDetail.log	Prestart log file
	statusDetail.log	Log file of the Hue health status
	startDetail.log	Startup log
get-hue-ha.log	Log file of the Hue HA status	

Type	Log File Name	Description
	hue-ha-status.log	Log file of the Hue HA status monitoring
	get-hue-health.log	Log file of the Hue health status
	hue-health-check.log	Log file of the Hue health check
	hue-refresh-config.log	Log file of the Hue configuration update
	hue-script-log.log	Log file of the Hue operations on the Manager console
	hue-service-check.log	Log file of the Hue service status monitoring
	db_pwd.log	Log that records the changes of the password for Hue to connect to the DBService database
	modifyDBPwd_Date.log	-
	watch_config_update.log	Parameter update log file
Audit log	hue-audits.log	Audit log file

Log Level

Table 12-7 describes the log levels supported by Hue.

Levels of logs are ERROR, WARN, INFO, and DEBUG from the highest to the lowest priority. Run logs of equal or higher levels are recorded. The higher the specified log level, the fewer the logs recorded.

Table 12-7 Log levels

Level	Description
ERROR	Logs of this level record error information about system running.
WARN	Logs of this level record exception information about the current event processing.
INFO	Logs of this level record normal running status information about the system and events.
DEBUG	Logs of this level record the system information and system debugging information.

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page of the Hue service by referring to **Modifying Cluster Service Configuration Parameters**.
- Step 2** In the navigation tree on the left, select **Log** corresponding to the role to be modified.
- Step 3** Select the log level to be changed on the right.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.
- Step 5** Restart the service or instance whose configuration has expired for the configuration to take effect.

----End

Log Format

The following table lists the Hue log formats:

Table 12-8 Log formats

Type	Format	Example
Run log	<i><dd-MM-yy HH:mm:ss,SSS><Location where the log event occurs><Log level><Message in the log></i>	[03/Nov/2014 11:57:19] middleware INFO Unloading MimeTypeJSFileFixStreamingMiddleware.
	<i><Log level><Time format><yyyy-MM-dd HH:mm:ss,SSS><Location where the log event occurs><Message in the log></i>	INFO : CST 2014-11-06 11:22:52 hue-ha-status.sh : update 4 <= 15:myHostName=10.0.0.250 ACTIVE=10.0.0.250
Audit log	<i><UserName><yyyy-MM-dd HH:mm:ss,SSS><Audit operation description> <Resource parameter> <URL> <Whether to allow> <Audit operation> <IP address></i>	{"username": "admin", "eventTime": "2014-11-06 10:28:34", "operationText": "Successful login for user: admin", "service": "accounts", "url": "/accounts/login/", "allowed": true, "operation": "USER_LOGIN", "ipAddress": "10.0.0.250"}

12.12 Common Issues About Hue

12.12.1 How Do I Solve the Problem that HQL Fails to Be Executed in Hue Using Internet Explorer?

Question

What do I do if all HQL statements fail to be executed when I use Internet Explorer to access Hive Editor in Hue and the message "There was an error with your query" is displayed?

Answer

Internet Explorer does not support processing of AJAX POST requests containing form data in 307 redirection. You are advised to use a compatible browser, for example, Google Chrome.

12.12.2 Why Does the use database Statement Become Invalid When Hive Is Used?

Question

When Hive is used, the **use database** statement is entered in the text box to switch the database, and other statements are also entered, why does the database fail to be switched?

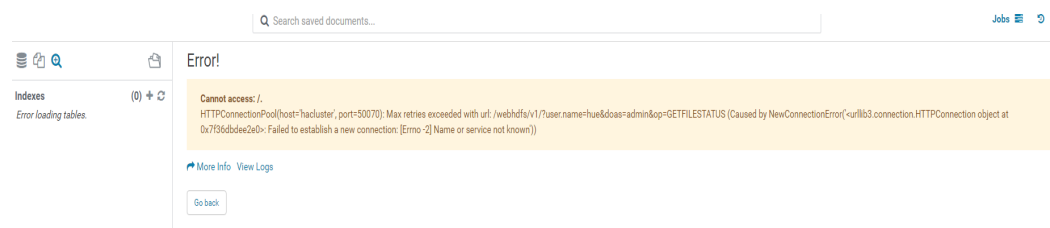
Answer

Using Hive on Hue is different from using Hive on the Hive client. There is an option to select a database on the Hue interface, and the database where the current SQL is executed is the one that is displayed on the interface. You are advised to use functions on the Hue interface instead of using statements to perform session-level and one-off operations, for example, setting parameters. If you must enter specific statements to perform an operation, ensure that all statements you enter are in one text box.

12.12.3 What Can I Do If HDFS Files Fail to Be Accessed Using Hue WebUI?

Question

What can I do if an error message shown in the following figure is displayed, indicating that the HDFS file cannot be accessed when I use Hue web UI to access the HDFS file?



Answer

1. Check whether the user who logs in to the Hue web UI has the permissions of the **hadoop** user group.
2. Check whether the HttpFS instance has been installed for the HDFS service and is running properly. If the HttpFS instance is not installed, manually install and restart the Hue service.

12.12.4 What Can I Do If a Large File Fails to Be Uploaded on the Hue Page?

Question

What can I do when a large file fails to be uploaded on the Hue page?

Answer

1. You are advised to run commands on the client to upload large files instead of using the Hue file browser.
2. If you must use Hue to upload the file, perform the following steps to modify Httpd parameters:
 - a. Log in to the active management node as user **omm**.
 - b. Run the following command to edit the **httpd.conf** file:
vi \$BIGDATA_HOME/om-server/Apache-httpd-*/conf/httpd.conf
 - c. Search for **21201** and add **RequestReadTimeout handshake=0 header=0 body=0** to the **</VirtualHost>** configuration, as shown in the following:

```
...
<VirtualHost *:21201>
  ServerName https://10.112.16.93:21201
  AllowEncodedSlashes On
  SSLProxyEngine On
  ProxyRequests Off
  TraceEnable off
  ProxyTimeout 1200
  RewriteEngine on
  RewriteMap proxylist dbm:${BIGDATA_ROOT_HOME}/om-server_*/Apache-httpd-*/conf/
proxylist.dbm

  RewriteRule ^(\./.*)$ ${proxylist:/Hue/Hue/21201}$1 [E=TARGET_PATH:$1,L,P]

  Header edit Location ^(!https://10.112.16.93:20009|https://10.112.16.93:21201)http[s]?://
[^/]*(.*)$ https://10.112.16.93:21201$1

  ProxyPassReverseCookiePath / / interpolate

  SSLEngine On
  SSLProxyProtocol All +TLSv1.2 -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
  SSLProtocol ALL +TLSv1.2 -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
  SSLCipherSuite ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:DHE-DSS-
AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:DHE-DSS-AES128-GCM-SHA256:DHE-
RSA-AES128-GCM-SHA256
  SSLProxyCheckPeerName off
  SSLProxyCheckPeerCN off
  SSLCertificateFile "${BIGDATA_ROOT_HOME}/om-server_*/Apache-httpd-*/conf/security/
proxy_ssl.cert"
  SSLCertificateKeyFile "${BIGDATA_ROOT_HOME}/om-server_*/Apache-httpd-*/conf/security/
```

```
server.key"  
    SSLProxyCACertificateFile "${BIGDATA_ROOT_HOME}/om-server_*/apache-tomcat-*/conf/  
security/tomcat.crt  
    SSLCertificateChainFile "${BIGDATA_ROOT_HOME}/om-server_*/Apache-httpd-2.4.39/conf/  
security/proxy_chain.crt"  
    RequestReadTimeout handshake=0 header=0 body=0  
</VirtualHost>  
...
```

- d. Run the `ps -ef|grep httpd|grep -v grep|xargs kill -9` command to restart `httpd`.

12.12.5 Why Is the Hue Native Page Cannot Be Properly Displayed If the Hive Service Is Not Installed in a Cluster?

Question

Why is the native Hue page blank if the Hive service is not installed in a cluster?

Answer

In MRS 3.x, Hue depends on Hive. If this problem occurs, check whether the Hive component is installed in the current cluster. If not, install it.

13 Using Impala

13.1 Using Impala from Scratch

Impala is a massively parallel processing (MPP) SQL query engine for processing vast amounts of data stored in Hadoop clusters. It is an open source software written in C++ and Java. It provides high performance and low latency compared with other SQL engines for Hadoop.

Background

Suppose a user develops an application to manage users who use service A in an enterprise. The procedure of operating service A on the Impala client is as follows:

Operations on common tables:

- Create the **user_info** table.
- Add users' educational backgrounds and titles to the table.
- Query user names and addresses by user ID.
- Delete the user information table after service A ends.

Table 13-1 User information

No.	Name	Gender	Age	Address
12005000201	A	Male	19	City A
12005000202	B	Female	23	City B
12005000203	C	Male	26	City C
12005000204	D	Male	18	City D
12005000205	E	Female	21	City E
12005000206	F	Male	32	City F
12005000207	G	Female	29	City G

No.	Name	Gender	Age	Address
12005000208	H	Female	30	City H
12005000209	I	Male	26	City I
12005000210	J	Female	25	City J

Prerequisites

The client has been installed. For example, the client is installed in the `/opt/hadoopclient` directory. The client directory in the following operations is only an example. Change it to the actual installation directory.

Procedure

Step 1 Log in to the node where the client is installed as the client installation user.

Step 2 Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

Step 3 Run the following command to configure environment variables:

```
source bigdata_env
```

Step 4 Run the Impala client command to implement service A.

Run the client command of the Impala component directly.

```
impala-shell
```

NOTE

By default, **impala-shell** attempts to connect to the Impala daemon on port 21000 of **localhost**. To connect to another host, use the **-i <host:port>** option, for example, **impala-shell -i xxx.xxx.xxx.xxx:21000**. To automatically connect to a specific Impala database, use the **-d <database>** option. For example, if all your Kudu tables are in the **impala_kudu** database, **-d impala_kudu** can use this database. To exit the Impala Shell, run the **quit** command.

Operations on internal tables:

1. Create the **user_info** user information table according to [Table 13-1](#) and add data to it.

```
create table user_info(id string,name string,gender string,age int,addr string);
insert into table user_info(id,name,gender,age,addr) values("12005000201", "A", "Male", 19, "City A");
```

... (Other statements are the same.)

2. Add users' educational backgrounds and titles to the **user_info** table.

For example, to add educational background and title information about user 12005000201, run the following commands.

```
alter table user_info add columns(education string,technical string);
```

3. Query user names and addresses by user ID.

For example, to query the name and address of user 12005000201, run the following command:

```
select name,addr from user_info where id='12005000201';
```

4. Delete the user information table:

```
drop table user_info;
```

Operations on external partition tables:

Create an external partition table and import data.

1. Create a path for storing external table data.

```
hdfs dfs -mkdir /hive
```

```
hdfs dfs -mkdir /hive/user_info
```

2. Create a table.

```
create external table user_info(id string,name string,gender string,age int,addr string) partitioned  
by(year string) row format delimited fields terminated by ' ' lines terminated by '\n' stored as textfile  
location '/hive/user_info';
```

NOTE

fields terminated indicates delimiters, for example, spaces.

lines terminated indicates line breaks, for example, `\n`.

`/hive/user_info` indicates the path of the data file.

3. Import data.

- a. Execute the **insert** statement to insert data.

```
insert into user_info partition(year="2018") values ("12005000201", "A", "Male", 19, "City A");
```

- b. Run the **load data** command to import file data.

- i. Create a file based on the data in [Table 13-1](#). For example, the file name is **txt.log**. Fields are separated by space, and the line feed characters are used as the line breaks.

- ii. Upload the file to HDFS.

```
hdfs dfs -put txt.log /tmp
```

- iii. Load data to the table.

```
load data inpath '/tmp/txt.log' into table user_info partition  
(year='2018');
```

4. Query the imported data:

```
select * from user_info;
```

5. Delete the user information table:

```
drop table user_info;
```

----End

13.2 Accessing the Impala Web UI

You can view Impala job information on the Impala web UI. Impala web UIs are classified into the following types based on instances:

- **StateStore WebUI:** used to manage nodes.
- **Catalog WebUI:** used to view metadata.
- **Impalad WebUI:** used to view details about each SQL statement.

Prerequisites

Impala has been installed in a cluster.

Accessing the StateStore Web UI

- Step 1** Access Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
 - Step 2** Choose **Services > Impala**.
 - Step 3** In **StateStore WebUI** of **Impala Summary**, click **StateStore(Statestore)**. The StateStore web UI is displayed.
- End

Accessing the Catalog Web UI

- Step 1** Access Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
 - Step 2** Choose **Services > Impala**.
 - Step 3** In **Catalog WebUI** of **Impala Summary**, click **Catalog(Catalog)**. The Catalog web UI is displayed.
- End

Accessing the Impalad Web UI

- Step 1** Access Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
 - Step 2** Choose **Services > Impala > Instance**.
 - Step 3** Move the cursor to the Impalad instance in the **Role** column. The following link is displayed in the lower left corner of the page. Obtain the value after **null**, for example, **82** in this example.

`https://EIP:9022/mrsmanager/index.jsp?locale=zh-cn#/app/services/Impala/Impalad/null/82/EIP/STARTED/status/detail`

In the preceding command, **82** is an example. Change it based on the site requirements.
 - Step 4** For details, see [Accessing the StateStore Web UI](#).
 - Step 5** Change **StateStore/xx** in the URL of the StateStore web UI to **Impalad/xx** and access the new URL, where **xx** is the value obtained in **Step 3**.
- End

13.3 Using Impala to Operate Kudu

You can use the SQL statements of Impala to insert, query, update, and delete data in Kudu as an alternative to using Kudu APIs to build custom Kudu applications.

Prerequisite

A complete cluster client has been installed. For example, the installation directory is **/opt/Bigdata/client**. The client directory in the following operations is only an example. Replace it with the actual installation directory.

Impala on Kudu

Step 1 Log in to the node where the client is installed.

Step 2 Run the following command to initialize environment variables:

```
source /opt/Bigdata/client/bigdata_env
```

Step 3 If Kerberos authentication is enabled for the cluster, perform the following operation to authenticate the user. If Kerberos authentication is not enabled for the cluster, skip this step.

```
kinit Service user
```

Step 4 Run the following command to log in to the Impala client:

```
impala-shell
```

NOTE

By default, **impala-shell** attempts to connect to the Impala daemon on port 21000 of **localhost**. To connect to another host, use the **-i <host:port>** option. To automatically connect to a specific Impala database, use the **-d <database>** option. For example, if all your Kudu tables are in the **impala_kudu** database, **-d impala_kudu** can use this database. To exit the Impala shell, run the **quit** command.

Step 5 Run the following commands to create an Impala table and import the prepared data, for example, data in the **/tmp/data10** directory:

```
create table dataorigin (name string,age string,pt string, date_p date) row  
format delimited fields terminated by ',' stored as textfile;
```

```
load data inpath '/tmp/data10' overwrite into table dataorigin;
```

Step 6 Run the following command to create a Kudu table. In the command, **kudu.master_addresses** indicates the IP address of the KuduMaster instance. Set it to the actual IP address.

```
create table dataorigin2 (name string,age string,pt string, date_p date,  
primary key(name)) stored as kudu  
TBLPROPERTIES('kudu.master_addresses'='192.168.190.164:7051,192.168.204.1  
78:7051,192.168.244.63:7051');
```

Step 7 Perform the following operations on the Kudu table.

1. Insert data.

```
insert into dataorigin2 select * from dataorigin;
```

2. Update data.

```
UPDATE dataorigin2 SET date_p="2021-03-31" where age="73";
```

3. Upsert rows.

```
UPSERT INTO dataorigin2 VALUES ("spjted","75","28","2021-03-32");
```

```
UPSERT INTO dataorigin2 VALUES ("kwhakb","92","29","2021-03-33");
```

```

UPSERT INTO dataorigin2 VALUES ("oftrkf","13","30","2021-03-34");
UPSERT INTO dataorigin2 VALUES ("kiewti","36","31","2021-03-35");
UPSERT INTO dataorigin2 VALUES ("rknmql","98","32","2021-03-36");
UPSERT INTO dataorigin2 VALUES ("fwcoij","52","33","2021-03-37");
UPSERT INTO dataorigin2 VALUES ("pgvpdo","37","34","2021-03-35");
4. Delete a row.
   DELETE FROM dataorigin2 WHERE date_p="2021-03-31";
----End

```

13.4 Interconnecting Impala with External LDAP

This section applies to MRS 3.1.0 or later.

Step 1 Log in to Manager.

Step 2 On Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Impala** > **Configurations** > **All Configurations** > **Impalad(Role)** > **LDAP**.

Step 3 Set the following parameters.

Table 13-2 Parameter configuration

Parameter	Description	Remarks
--enable_ldap_auth	Whether to enable LDAP authentication	Value: true or false
--ldap_bind_pattern	LDAP user DN pattern	Example: cn=%s,ou=People,dc=xxx,dc=com

Parameter	Description	Remarks
--ldap_passwords_in_clear_ok	Whether the LDAP password is sent in plaintext	<p>If this parameter is set to true, the LDAP password can be sent in plaintext.</p> <p>Value: true or false</p> <p>NOTE If --enable_ldap_auth is set to true, the LDAP TLS protocol is disabled by default during authentication. Therefore, you need to set --ldap_passwords_in_clear_ok to true. Otherwise, the Impala role will fail to be started.</p> <p>To enable the Ldap TLS protocol, set --ldap_tls to true in the customized configuration of the Impala role. After the configuration, the password can be sent in ciphertext.</p>
--ldap_uri-ip	LDAP IP address	-
--ldap_uri-port	LDAP port number	Default value: 389

Step 4 After the modification, click **Save** in the upper left corner. In the displayed dialog box, click **OK**.

Step 5 Choose **Cluster > Name of the desired cluster > Services > Impala > Instance**. On the displayed page, select the instances whose **Configuration Status** is **Expired**, choose **More > Restart Instance**, and restart the instance.

----End

14 Using Kafka

14.1 Using Kafka from Scratch

Scenario

You can create, query, and delete topics on a cluster client.

Prerequisites

The client has been installed. For example, the client is installed in the `/opt/hadoopclient` directory. The client directory in the following operations is only an example. Change it to the actual installation directory.

Using the Kafka Client (Versions Earlier Than MRS 3.x)

Step 1 Access the ZooKeeper instance page.

Click the cluster name to go to the cluster details page and choose **Components > ZooKeeper > Instances**.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

Step 2 View the IP addresses of the ZooKeeper role instance.

Record any IP address of the ZooKeeper instance.

Step 3 Log in to the node where the client is installed.

Step 4 Run the following command to switch to the client directory, for example, `/opt/hadoopclient/Kafka/kafka/bin`.

```
cd /opt/hadoopclient/Kafka/kafka/bin
```

Step 5 Run the following command to configure environment variables:

```
source /opt/hadoopclient/bigdata_env
```

Step 6 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the current user. If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit Kafka user
```

Step 7 Create a topic.

```
sh kafka-topics.sh --create --topic Topic name --partitions Number of partitions occupied by the topic --replication-factor Number of replicas of the topic --zookeeper IP address of the node where the ZooKeeper instance resides:clientPort/kafka
```

Step 8 Run the following command to view the topic information in the cluster:

```
sh kafka-topics.sh --list --zookeeper IP address of the node where the ZooKeeper instance resides:clientPort/kafka
```

Step 9 Delete the topic created in [Step 7](#).

```
sh kafka-topics.sh --delete --topic Topic name --zookeeper IP address of the node where the ZooKeeper instance resides:clientPort/kafka
```

Type **y** and press **Enter**.

```
----End
```

Using the Kafka Client (MRS 3.x or Later)

Step 1 Access the ZooKeeper instance page.

Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **ZooKeeper** > **Instance**.

Step 2 View the IP addresses of the ZooKeeper role instance.

Record any IP address of the ZooKeeper instance.

Step 3 Log in to the node where the client is installed.

Step 4 Run the following command to switch to the client directory, for example, **/opt/hadoopclient/Kafka/kafka/bin**.

```
cd /opt/hadoopclient/Kafka/kafka/bin
```

Step 5 Run the following command to configure environment variables:

```
source /opt/hadoopclient/bigdata_env
```

Step 6 If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the current user. If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit Kafka user
```

Step 7 Create a topic.

```
sh kafka-topics.sh --create --topic Topic name --partitions Number of partitions occupied by the topic --replication-factor Number of replicas of the topic --
```

zookeeper *IP address of the node where the ZooKeeper instance resides:clientPort/kafka*

Step 8 Run the following command to view the topic information in the cluster:

```
sh kafka-topics.sh --list --zookeeper IP address of the node where the ZooKeeper instance resides:clientPort/kafka
```

Step 9 Delete the topic created in [Step 7](#).

```
sh kafka-topics.sh --delete --topic Topic name --zookeeper IP address of the node where the ZooKeeper instance resides:clientPort/kafka
```

Type **y** and press **Enter**.

----End

14.2 Managing Kafka Topics

Scenario

You can manage Kafka topics on a cluster client based on service requirements. Management permission is required for clusters with Kerberos authentication enabled.

Prerequisites

You have installed the Kafka client.

Procedure

Step 1 Access the ZooKeeper instance page.

- For versions earlier than MRS 3.x, click the cluster name to go to the cluster details page and choose **Components** > **ZooKeeper** > **Instances**.

NOTE

- If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)
- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **ZooKeeper** > **Instance**.

Step 2 View the IP addresses of the ZooKeeper role instance.

Record any IP address of the ZooKeeper instance.

Step 3 Prepare the client based on service requirements. Log in to the node where the client is installed.

Log in to the node where the client is installed. For details, see [Installing a Client](#).

Step 4 Run the following command to switch to the client directory, for example, `/opt/client/Kafka/kafka/bin`.

```
cd /opt/client/Kafka/kafka/bin
```

Step 5 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 6 Run the following command to perform user authentication (skip this step in normal mode):

```
kinit Component service user
```

Step 7 For versions earlier than MRS 3.x, run the following commands to manage Kafka topics:

- Creating a topic

```
sh kafka-topics.sh --create --topic Topic name --partitions Number of partitions occupied by the topic --replication-factor Number of replicas of the topic --zookeeper IP address of the node where the ZooKeeper instance resides:clientPort/kafka
```

- Deleting a topic

```
sh kafka-topics.sh --delete --topic Topic name --zookeeper IP address of the node where the ZooKeeper instance resides:clientPort/kafka
```

 **NOTE**

- The number of topic partitions or topic backup replicas cannot exceed the number of Kafka instances.
- By default, the value of **clientPort** of ZooKeeper is **2181**.
- There are three ZooKeeper instances. Use the IP address of any one.
- For details about managing messages in Kafka topics, see [Managing Messages in Kafka Topics](#).

Step 8 MRS 3.x and later versions: Use **kafka-topics.sh** to manage Kafka topics.

- Creating a topic:

By default, partitions of a topic are distributed based on the number of partitions on the node and disk. To distribute partitions based on the disk capacity, set **log.partition.strategy** to **capacity** for the Kafka service.

When a topic is created in Kafka, partitions and copies can be generated based on the combination of rack awareness and cross-AZ feature. The **--zookeeper** and **--bootstrap-server** modes are supported.

- Disable the rack policy and cross-AZ feature (default policy).

Copies of topics created based on this policy are randomly allocated to any node in the cluster.

```
./kafka-topics.sh --create --topic topic name --partitions number of partitions occupied by the topic --replication-factor number of replicas of the topic --zookeeper IP address of any ZooKeeper node:clientPort/kafka
```

```
./kafka-topics.sh --create --topic topic name --partitions number of partitions occupied by the topic --replication-factor number of replicas of the topic --bootstrap-server IP address of the Kafka cluster:21007 --command-config ../config/client.properties
```

If you use **--bootstrap-server** to create a topic, set **rack.aware.enable** and **az.aware.enable** to **false**.

- Enable the rack policy and disable the cross-AZ feature.

The leader of each partition of the topic created based on this policy is randomly allocated on the cluster node. However, different replicas of the

same partition are allocated to different racks. Therefore, when this policy is used, ensure that the number of nodes in each rack is the same, otherwise, the load of nodes in the rack with fewer nodes is much higher than the average load of the cluster.

```
./kafka-topics.sh --create --topic topic name --partitions number of partitions occupied by the topic --replication-factor number of replicas of the topic --zookeeper IP address of any ZooKeeper node:clientPort/kafka --enable-rack-aware
```

```
./kafka-topics.sh --create --topic topic name --partitions number of partitions occupied by the topic --replication-factor number of replicas of the topic --bootstrap-server IP address of the Kafkacluster:21007 --command-config ../config/client.properties
```

If you use **--bootstrap-server** to create a topic, set **rack.aware.enable** to **true** and **az.aware.enable** to **false**.

- Disable the rack policy and enable the cross-AZ feature.

The leader of each partition of the topic created based on this policy is randomly allocated on the cluster node. However, different replicas of the same partition are allocated to different AZs. Therefore, when this policy is used, ensure that the number of nodes in each AZ is the same, otherwise, the load of nodes in the AZ with fewer nodes is much higher than the average load of the cluster.

```
./kafka-topics.sh --create --topic topic name --partitions number of partitions occupied by the topic --replication-factor number of replicas of the topic --zookeeper IP address of any ZooKeeper node:clientPort/kafka --enable-az-aware
```

```
./kafka-topics.sh --create --topic topic name --partitions number of partitions occupied by the topic --replication-factor number of replicas of the topic --bootstrap-server IP address of the Kafkacluster:21007 --command-config ../config/client.properties
```

If you use **--bootstrap-server** to create a topic, set **rack.aware.enable** to **false** and **az.aware.enable** to **true**.

- Enable the rack policy and cross-AZ feature.

The leader of each partition of the topic created based on this policy is randomly allocated on the cluster node. However, different replicas of the same partition are allocated to different racks in different AZs. This policy ensures that the number of nodes on each rack in each AZ is the same, otherwise, the load in the cluster is unbalanced.

```
./kafka-topics.sh --create --topic topic name --partitions number of partitions occupied by the topic --replication-factor number of replicas of the topic --zookeeper IP address of any ZooKeeper node:clientPort/kafka --enable-rack-aware --enable-az-aware
```

```
./kafka-topics.sh --create --topic topic name --partitions number of partitions occupied by the topic --replication-factor number of replicas of the topic --bootstrap-server IP address of the Kafkacluster:21007 --command-config ../config/client.properties
```

If you use **--bootstrap-server** to create a topic, set **rack.aware.enable** and **az.aware.enable** to **true**.

 NOTE

- Kafka supports topic creation in either of the following modes:
 - In **--zookeeper** mode, the client generates a copy allocation scheme. The community supports this mode from the beginning. To reduce the dependency on the ZooKeeper component, the community will delete the support for this mode in later versions. When creating a topic in this mode, you can select a copy allocation policy by combining the **--enable-rack-aware** and **--enable-az-aware** options. Note: The **--enable-az-aware** option can be used only when the cross-AZ feature is enabled on the server, that is, **az.aware.enable** is set to **true**. Otherwise, the execution fails.
 - In **--bootstrap-server** mode, the server generates a copy allocation solution. In later versions, the community supports only this mode for topic management. When a topic is created in this mode, the **--enable-rack-aware** and **--enable-az-aware** options cannot be used to control the copy allocation policy. The **rack.aware.enable** and **az.aware.enable** parameters can be used together to control the copy allocation policy. Note that the **az.aware.enable** parameter cannot be modified; if the cross-AZ feature is enabled during cluster creation, this parameter is automatically set to **true**; the **rack.aware.enable** parameter can be customized.
- List of topics:
 - `./kafka-topics.sh --list --zookeeper service IP address of any ZooKeeper node:clientPort/kafka`
 - `./kafka-topics.sh --list --bootstrap-server IP address of the Kafkacluster:21007 --command-config ../config/client.properties`
- Viewing the topic:
 - `./kafka-topics.sh --describe --zookeeper service IP address of any ZooKeeper node:clientPort/kafka --topic topic name`
 - `./kafka-topics.sh --describe --bootstrap-server IP address of the Kafkacluster:21007 --command-config ../config/client.properties --topic topic name`
- Modifying a topic:
 - `./kafka-topics.sh --alter --topic topic name --config configuration item=configuration value --zookeeper service IP address of any ZooKeeper node:clientPort/kafka`
- Expanding partitions:
 - `./kafka-topics.sh --alter --topic topic name --zookeeper service IP address of any ZooKeeper node:clientPort/kafka --command-config Kafka/kafka/config/client.properties --partitions number of partitions after the expansion`
 - `./kafka-topics.sh --alter --topic topic name --bootstrap-server IP address of the Kafka cluster:21007 --command-config Kafka/kafka/config/client.properties --partitions number of partitions after the expansion`
- Deleting a topic
 - `./kafka-topics.sh --delete --topic topic name --zookeeper Service IP address of any ZooKeeper node:clientPort/kafka`

- `./kafka-topics.sh --delete --topic topic name--bootstrap-server IP address of the Kafka cluster:21007 --command-config ./config/client.properties`

----End

14.3 Querying Kafka Topics

Scenario

You can query existing Kafka topics on MRS.

Procedure

Step 1 Go to the Kafka service page.

- For versions earlier than MRS 3.x, click the cluster name to go to the cluster details page and choose **Components** > **Kafka**.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services** > **Kafka**.

Step 2 Click **KafkaTopicMonitor**.

All topics are displayed in the list by default. You can view the number of partitions and replicas of the topics.

Step 3 Click the desired topic in the list to view its details.

----End

14.4 Managing Kafka User Permissions

Scenario

For clusters with Kerberos authentication enabled, using Kafka requires relevant permissions. MRS clusters can grant the use permission of Kafka to different users.

Table 14-1 lists the default Kafka user groups.

 **NOTE**

In MRS 3.x or later, Kafka supports two types of authentication plug-ins: Kafka open-source authentication plug-in and Ranger authentication plug-in.

This section describes the user permission management based on the Kafka open source authentication plug-in. For details about how to use the Ranger authentication plug-in, see [Adding a Ranger Access Permission Policy for Kafka](#).

Table 14-1 Default Kafka user groups

User Group	Description
kafkaadmin	Kafka administrator group. Users in this group have the permissions to create, delete, read, and write all topics, and authorize other users.
kafkasuperuser	Kafka super user group. Users in this group have the permissions to read and write all topics.
kafka	Kafka common user group. Users in this group can access a topic only when they are granted with the read and write permissions of the topic by a user in the kafkaadmin group.

Prerequisites

- You have installed the Kafka client.
- A user in the **kafkaadmin** group, for example **admin**, has been prepared.

Procedure

Step 1 Access the ZooKeeper instance page.

- For versions earlier than MRS 3.x, click the cluster name to go to the cluster details page and choose **Components > ZooKeeper > Instances**.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster > Name of the desired cluster > Services > ZooKeeper > Instance**.

Step 2 View the IP addresses of the ZooKeeper role instance.

Record the IP address of any ZooKeeper instance.

Step 3 Prepare the client based on service requirements. Log in to the node where the client is installed.

Log in to the node where the client is installed. For details, see [Installing a Client](#).

Step 4 Run the following command to switch to the client directory, for example, **/opt/client/Kafka/kafka/bin**.

```
cd /opt/client/Kafka/kafka/bin
```

Step 5 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 6 Run the following command to authenticate the user (skip this step in normal mode):

kinit *Component service user*

Step 7 Versions earlier than MRS 3.x: Select the scenario required by the service and manage Kafka user permissions.

- Querying the permission list of a topic
sh kafka-acls.sh --authorizer-properties zookeeper.connect=IP address of the node where the ZooKeeper instance resides.2181/kafka --list --topic Topic name
- Adding producer permission to a user
sh kafka-acls.sh --authorizer-properties zookeeper.connect=IP address of the node where the ZooKeeper instance resides.2181/kafka --add --allow-principal User:Username --producer --topic Topic name
- Removing producer permission of a user
sh kafka-acls.sh --authorizer-properties zookeeper.connect=IP address of the node where the ZooKeeper instance resides.2181/kafka --remove --allow-principal User:Username --producer --topic Topic name
- Adding consumer permission to a user
sh kafka-acls.sh --authorizer-properties zookeeper.connect=IP address of the node where the ZooKeeper instance resides.2181/kafka --add --allow-principal User:Username --consumer --topic Topic name --group Consumer group name
- Removing consumer permission of a user
sh kafka-acls.sh --authorizer-properties zookeeper.connect=IP address of the node where the ZooKeeper instance resides.2181/kafka --remove --allow-principal User:Username --consumer --topic Topic name --group Consumer group name

 **NOTE**

You need to enter **y** twice to confirm the removal of permission.

Step 8 MRS 3.x and later versions: The following table lists the common commands used for user authorization when **kafka-acl.sh** is used.

- View the permission control list of a topic:
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP address of any ZooKeeper node:2181/kafka > --list --topic <topicname>
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --command-config ../config/client.properties --list --topic <topic name>
- Add the Producer permission for a user:
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP address of any ZooKeeper node:2181/kafka > --add --allow-principal User:<username> --producer --topic <topic name>
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --command-config ../config/client.properties --add --allow-principal User:<username> --producer --topic <topic name>
- Assign the Producer permission to a user in batches.
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP address of any ZooKeeper node:2181/kafka > --add --allow-principal User:<username> --producer --topic <topic name> --resource-pattern-type prefixed

```
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --  
command-config ../config/client.properties --add --allow-principal  
User:<username> --producer --topic <topic name>--resource-pattern-type  
prefixed
```

- Remove the Producer permission from a user:

```
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP  
address of any ZooKeeper node:2181/kafka > --remove --allow-principal  
User:<username> --producer --topic <topic name>
```

```
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --  
command-config ../config/client.properties --remove --allow-principal  
User:<username> --producer --topic <topic name>
```

- Delete the Producer permission of a user in batches:

```
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP  
address of any ZooKeeper node:2181/kafka > --remove --allow-principal  
User:<username> --producer --topic <topic name> --resource-pattern-type  
prefixed
```

```
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --  
command-config ../config/client.properties --remove --allow-principal  
User:<username> --producer --topic <topic name>--resource-pattern-type  
prefixed
```

- Add the Consumer permission for a user:

```
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP  
address of any ZooKeeper node:2181/kafka > --add --allow-principal  
User:<user name> --consumer --topic <topic name> --group <consumer  
group name>
```

```
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --  
command-config ../config/client.properties --add --allow-principal  
User:<username> --consumer --topic <topicname> --group <consumer  
group name>
```

- Add consumer permissions to a user in batches:

```
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP  
address of any ZooKeeper node:2181/kafka > --add --allow-principal  
User:<username> --consumer --topic <topic name> --group <consumer  
group name> --resource-pattern-type prefixed
```

```
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --  
command-config ../config/client.properties --add --allow-principal  
User:<username> --consumer --topic <topicname> --group <consumer  
group name> --resource-pattern-type prefixed
```

- Remove the consumer permission from a user:

```
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP  
address of any ZooKeeper node:2181/kafka > --remove --allow-principal  
User:<username> --consumer --topic <topic name> --group <consumer  
group name>
```

```
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --  
command-config ../config/client.properties --remove --allow-principal  
User:<username> --consumer --topic <topic name> --group <consumer  
group name>
```

- Delete the consumer permission of a user in batches:

```
./kafka-acls.sh --authorizer-properties zookeeper.connect=<service IP  
address of any ZooKeeper node:2181/kafka > --remove --allow-principal  
User:<username> --consumer --topic <topic name> --group <consumer  
group name> --resource-pattern-type prefixed
```

```
./kafka-acls.sh --bootstrap-server <IP address of the Kafkacluster:21007> --  
command-config ../config/client.properties --remove --allow-principal  
User:<username> --consumer --topic <topicname> --group <consumer  
group name> --resource-pattern-type prefixed
```

----End

14.5 Managing Messages in Kafka Topics

Scenario

You can produce or consume messages in Kafka topics using the MRS cluster client. For clusters with Kerberos authentication enabled, you must have the permission to perform these operations.

Prerequisites

You have installed the Kafka client.

Procedure

Step 1 Go to the Kafka service page.

- For versions earlier than MRS 3.x, click the cluster name to go to the cluster details page and choose **Components** > **Kafka**.

NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager and choose **Cluster** > *Name of the desired cluster* > **Services** > **Kafka**.

Step 2 Click **instance**. Query the IP addresses of the Kafka instances.

Record the IP address of any Kafka instance.

Step 3 Prepare the client based on service requirements. Log in to the node where the client is installed.

Log in to the node where the client is installed. For details, see [Installing a Client](#).

Step 4 Run the following command to switch to the client directory, for example, `/opt/client/Kafka/kafka/bin`.

```
cd /opt/client/Kafka/kafka/bin
```

Step 5 Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
```

Step 6 For clusters with Kerberos authentication enabled, run the following command to authenticate the user. For clusters with Kerberos authentication disabled, skip this step.

kinit *Kafka user*

Example:

kinit admin

Step 7 Manage messages in Kafka topics using the following commands:

- Producing messages

```
sh kafka-console-producer.sh --broker-list IP address of the node where the Kafka instance resides:9092 --topic Topic name --producer.config /opt/client/Kafka/kafka/config/producer.properties
```

You can input specified information as the messages produced by the producer and then press **Enter** to send the messages. To end message producing, press **Ctrl + C** to exit.

- Consuming messages

```
sh kafka-console-consumer.sh --topic Topic name --bootstrap-server IP address of the node where the Kafka instance resides:9092 --consumer.config /opt/client/Kafka/kafka/config/consumer.properties
```

In the configuration file, **group.id** (indicating the consumer group) is set to **example-group1** by default. Users can change the value as required. The value takes effect each time consumption occurs.

By default, the system reads unprocessed messages in the current consumer group when the command is executed. If a new consumer group is specified in the configuration file and the **--from-beginning** parameter is added to the command, the system reads all messages that have not been automatically deleted in Kafka.

 **NOTE**

----End

14.6 Synchronizing Binlog-based MySQL Data to the MRS Cluster

This section describes how to use the Maxwell data synchronization tool to migrate offline binlog-based data to an MRS Kafka cluster.

Maxwell is an open source application that reads MySQL binlogs, converts operations, such as addition, deletion, and modification, into a JSON format, and sends them to an output end, such as a console, a file, and Kafka. For details about Maxwell, visit <https://maxwells-daemon.io>. Maxwell can be deployed on a MySQL server or on other servers that can communicate with MySQL.

Maxwell runs on a Linux server, including EulerOS, Ubuntu, Debian, CentOS, and OpenSUSE. Java 1.8+ must be supported.

The following provides details about data synchronization.

1. [Configuring MySQL](#)
2. [Installing Maxwell](#)
3. [Configuring Maxwell](#)
4. [Starting Maxwell](#)
5. [Verifying Maxwell](#)
6. [Stopping Maxwell](#)
7. [Format of the Maxwell Generated Data and Description of Common Fields](#)

Configuring MySQL

- Step 1** Start the binlog, open the **my.cnf** file in MySQL, and check whether **server_id**, **log-bin**, and **binlog_format** are configured in the **[mysqld]** block. If they are not configured, run the following command to add configuration items and restart MySQL. If they are configured, skip this step.

```
$ vi my.cnf

[mysqld]
server_id=1
log-bin=master
binlog_format=row
```

- Step 2** Maxwell needs to connect to MySQL, create a database named **maxwell** for storing metadata, and access the database to be synchronized. Therefore, you are advised to create a MySQL user for Maxwell to use. Log in to MySQL as user **root** and run the following commands to create a user named **maxwell** (**XXXXXX** indicates the password and needs to be replaced with actual one).

- If Maxwell is deployed on a non-MySQL server, the created user **maxwell** must have a permission to remotely log in to the database. In this case, run the following command to create the user:

```
mysql> GRANT ALL on maxwell.* to 'maxwell'@'%' identified by 'XXXXXX';
```

```
mysql> GRANT SELECT, REPLICATION CLIENT, REPLICATION SLAVE on *.* to 'maxwell'@'%';
```

- If Maxwell is deployed on the MySQL server, the created user **maxwell** can be configured to log in to the database only on the local host. In this case, run the following command:

```
mysql> GRANT SELECT, REPLICATION CLIENT, REPLICATION SLAVE on *.* to 'maxwell'@'localhost' identified by 'XXXXXX';
```

```
mysql> GRANT ALL on maxwell.* to 'maxwell'@'localhost';
```

----End

Installing Maxwell

- Step 1** Download the installation package at <https://github.com/zendesk/maxwell/releases> and select the **maxwell-XXX.tar.gz** binary file for download. In the file name, **XXX** indicates a version number.
- Step 2** Upload the **tar.gz** package to any directory (the **/opt** directory of the Master node used as an example here).

Step 3 Log in to the server where Maxwell is deployed and run the following command to go to the directory where the **tar.gz** package is stored.

```
cd /opt
```

Step 4 Run the following commands to decompress the **maxwell-XXX.tar.gz** package and go to the **maxwell-XXX** directory:

```
tar -zxvf maxwell-XXX.tar.gz
```

```
cd maxwell-XXX
```

```
----End
```

Configuring Maxwell

If the **conf** directory exists in the **maxwell-XXX** folder, configure the **config.properties** file. For details about the configuration items, see [Table 14-2](#). If the **conf** directory does not exist, change **config.properties.example** in the **maxwell-XXX** folder to **config.properties**.

Table 14-2 Maxwell configuration item description

Parameter	Mandatory	Description	Default Value
user	Yes	Name of the user for connecting to MySQL, that is, the user created in Step 2 .	-
password	Yes	Password for connecting to MySQL	-
host	No	MySQL address	localhost
port	No	MySQL port	3306
log_level	No	Log print level. The options are as follows: <ul style="list-style-type: none"> • debug • info • warn • error 	info
output_ddl	No	Whether to send a DDL (modified based on definitions of the database and data table) event <ul style="list-style-type: none"> • true: Send DDL events. • false: Do not send DDL events. 	false
producer	Yes	Producer type. Set this parameter to kafka . <ul style="list-style-type: none"> • stdout: Log the generated events. • kafka: Send the generated events to Kafka. 	stdout

Parameter	Mandatory	Description	Default Value
producer_partition_by	No	Partition policy used to ensure that data of the same type is written to the same partition of Kafka. <ul style="list-style-type: none"> • database: Events of the same database are written to the same partition of Kafka. • table: Events of the same table are written to the same partition of Kafka. 	database
ignore_producer_error	No	Specifies whether to ignore the error that the producer fails to send data. <ul style="list-style-type: none"> • true: The error information is logged and the error data is skipped. The program continues to run. • false: The error information is logged and the program is terminated. 	true
metrics_slf4j_interval	No	Interval for outputting statistics on data successfully uploaded or failed to be uploaded to Kafka in logs. The unit is second.	60
kafka.bootstrap.servers	Yes	Address of the Kafka proxy node. The value is in the format of HOST:PORT[,HOST:PORT] .	-
kafka_topic	No	Name of the topic that is written to Kafka	maxwell
dead_letter_topic	No	Kafka topic used to record the primary key of the error log record when an error occurs when the record is sent	-
kafka_version	No	Kafka producer version used by Maxwell, which cannot be configured in the config.properties file. You need to use the --kafka_version xxx parameter to import the version number when starting the command.	-
kafka_partition_hash	No	Kafka topic partitioning algorithm. The value can be default or murmur3 .	default
kafka_key_format	No	Key generation method of the Kafka record. The value can be array or Hash .	Hash
ddl_kafka_topic	No	Topic that is written to the DDL operation when output_ddl is set to true	{kafka_topic}

Parameter	Mandatory	Description	Default Value
filter	No	<p>Used to filter databases or tables.</p> <ul style="list-style-type: none"> If only the mydatabase database needs to be collected, set this parameter to the following: exclude: *.*;include: mydatabase.* If only the mydatabase.mytable table needs to be collected, set this parameter to the following: exclude: *.*;include: mydatabase.mytable If only the mytable, mydate_123, and mydate_456 tables in the mydatabase database need to be collected, set this parameter to the following: exclude: *.*;include: mydatabase.mytable, include: mydatabase./mydate_\\d*/ 	-

Starting Maxwell

Step 1 Log in to the server where Maxwell is deployed.

Step 2 Run the following command to go to the Maxwell installation directory:

```
cd /opt/maxwell-1.21.0/
```

NOTE

For the first time to use Maxwell, you are advised to change **log_level** in **conf/config.properties** to **debug** (debug level) so that you can check whether data can be obtained from MySQL and sent to Kafka after startup. After the entire process is debugged, change **log_level** to **info**, and then restart Maxwell for the modification to take effect.

```
# log level [debug | info | warn | error]
```

```
log_level=debug
```

Step 3 Run the following commands to start Maxwell:

```
source /opt/client/bigdata_env
```

```
bin/Maxwell
```

```
bin/maxwell --user='maxwell' --password='XXXXXX' --host='127.0.0.1' \
```

```
--producer=kafka --kafka.bootstrap.servers=kafkahost:9092 --  
kafka_topic=Maxwell
```

In the preceding commands, **user**, **password**, and **host** indicate the username, password, and IP address of MySQL, respectively. You can configure the three parameters by modifying configurations of the configuration items or using the preceding commands. **kafkahost** indicates the IP address of the Core node in the streaming cluster.

If information similar to the following appears, Maxwell has started successfully:

```
Success to start Maxwell [78092].
```

----End

Verifying Maxwell

- Step 1** Log in to the server where Maxwell is deployed.
- Step 2** View the logs. If the log file does not contain an ERROR log and the following information is displayed, the connection between Maxwell and MySQL is normal:
- ```
BinlogConnectorLifecycleListener - Binlog connected.
```

- Step 3** Log in to the MySQL database and update, create, or delete test data. The following provides operation statement examples for your reference.

```
--Creating a database
create database test;
--Creating a table
create table test.e (
 id int(10) not null primary key auto_increment,
 m double,
 c timestamp(6),
 comment varchar(255) charset 'latin1'
);
-- Adding a record
insert into test.e set m = 4.2341, c = now(3), comment = 'I am a creature of light.';
--Updating a record
update test.e set m = 5.444, c = now(3) where id = 1;
--Deleting a record
delete from test.e where id = 1;
--Modifying a table
alter table test.e add column torvalds bigint unsigned after m;
--Deleting a table
drop table test.e;
-- Deleting a database
drop database test;
```

- Step 4** Check the Maxwell logs. If no WARN/ERROR is displayed, Maxwell is installed and configured properly.

To check whether the data is successfully uploaded, set **log\_level** in the **config.properties** file to **debug**. When the data is successfully uploaded, the following JSON data is printed immediately. For details about the fields, see [Format of the Maxwell Generated Data and Description of Common Fields](#).

```
{"database":"test","table":"e","type":"insert","ts":1541150929,"xid":60556,"commit":true,"data":{"id":1,"m":4.2341,"c":"2018-11-02 09:28:49.297000","comment":"I am a creature of light."}}
.....
```

### NOTE

After the entire process is debugged, you can change the value of **log\_level** in the **config.properties** file to **info** to reduce the number of logs to be printed and restart Maxwell for the modification to take effect.

```
log level [debug | info | warn | error]
log_level=info
```

----End

## Stopping Maxwell

- Step 1** Log in to the server where Maxwell is deployed.
- Step 2** Run the command to obtain the Maxwell process ID (PID). The second field in the command output is PID.

```
ps -ef | grep Maxwell | grep -v grep
```

**Step 3** Run the following command to forcibly stop the Maxwell process:

```
kill -9 PID
```

```
----End
```

## Format of the Maxwell Generated Data and Description of Common Fields

The data generated by Maxwell is in JSON format. The common fields are described as follows:

- **type**: operation type. The options are **database-create**, **database-drop**, **table-create**, **table-drop**, **table-alter**, **insert**, **update**, and **delete**.
- **database**: name of the database to be operated
- **ts**: operation time, which is a 13-digit timestamp
- **table**: name of the table to be operated
- **data**: content after data is added, deleted, or modified
- **old**: content before data is modified or schema definition before a table is modified
- **sql**: SQL statement for DDL operations
- **def**: schema definition for table creation and modification
- **xid**: unique ID of an object
- **commit**: check whether such operations as data addition, deletion, and modification have been submitted.

## 14.7 Creating a Kafka Role

### Scenario

This section describes how to create and configure a Kafka role.

This section applies to MRS 3.x or later.

#### NOTE

Users can create Kafka roles only in security mode.

If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. For details, see [Adding a Ranger Access Permission Policy for Kafka](#).

### Prerequisites

You have understood the service requirements.

### Procedure

**Step 1** Log in to FusionInsight Manager and choose **System > Permission > Role**.

**Step 2** On the displayed page, click **Create Role** and enter a **Role Name** and **Description**.

- Step 3** On the **Configure Resource Permission** page, choose *Name of the desired cluster* > **Kafka**.
- Step 4** Select permissions based on service requirements. For details about configuration items, see [Table 14-3](#).

**Table 14-3** Description

| Scenario                                                | Role Authorization                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Setting the Kafka administrator permissions             | In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> > <b>Kafka</b> > <b>Kafka Manager Privileges</b> .<br><br><b>NOTE</b><br>This permission allows you to create and delete topics, but does not allow you to produce or consume any topics.                                                     |
| Setting the production permission of a user on a topic  | <ol style="list-style-type: none"> <li>1. In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> &gt; <b>Kafka</b> &gt; <b>Kafka Topic Producer And Consumer Privileges</b>.</li> <li>2. In the <b>Permission</b> column of the specified topic, select <b>Kafka Producer Permission</b>.</li> </ol> |
| Setting the consumption permission of a user on a topic | <ol style="list-style-type: none"> <li>1. In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> &gt; <b>Kafka</b> &gt; <b>Kafka Topic Producer And Consumer Privileges</b>.</li> <li>2. In the <b>Permission</b> column of the specified topic, select <b>Kafka Consumer Privileges</b>.</li> </ol> |

- Step 5** Click **OK**, and return to the **Role** page.
- End

## 14.8 Kafka Common Parameters

This section applies to MRS 3.x or later.

### Navigation path for setting parameters:

For details about how to set parameters, see [Modifying Cluster Service Configuration Parameters](#).

## Common Parameters

**Table 14-4** Parameter description

| Parameter                  | Description                                                                                                                                                          | Default Value                                |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------|
| log.dirs                   | List of Kafka data storage directories. Use commas (,) to separate multiple directories.                                                                             | %<br>{@auto.detect.datapart.b<br>k.log.logs} |
| KAFKA_HEAP_OPTS            | Specifies the JVM option used for Kafka to start broker. It is recommended that you set this parameter based on service requirements.                                | -Xmx6G -Xms6G                                |
| auto.create.topics.enable  | Indicates whether a topic is automatically created. If this parameter is set to <b>false</b> , you need to run a command to create a topic before sending a message. | true                                         |
| default.replication.factor | Default number of replicas of a topic is automatically created.                                                                                                      | 2                                            |
| monitor.preInitDelay       | Delay of the first health check after the server is started. If the startup takes a long time, increase the value of the parameter. Unit: millisecond                | 600,000                                      |

## Timeout Parameters

**Table 14-5** Broker-related timeout parameters

| Parameter                    | Description                                                                  | Default Value | Impact                                                 |
|------------------------------|------------------------------------------------------------------------------|---------------|--------------------------------------------------------|
| controller.socket.timeout.ms | Specifies the timeout for connecting controller to broker. Unit: millisecond | 30,000        | Generally, retain the default value of this parameter. |

| Parameter                    | Description                                                                                                                                                                                                                                                                           | Default Value | Impact                                                                                                                                                                                                             |
|------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| group.max.session.timeout.ms | Specifies the maximum session timeout during the consumer registration. Unit: millisecond                                                                                                                                                                                             | 180000        | The configured value must be less than the value of this parameter.                                                                                                                                                |
| group.min.session.timeout.ms | Specifies the minimum session timeout during the consumer registration. Unit: millisecond                                                                                                                                                                                             | 6000          | The configured value must be greater than the value of this parameter.                                                                                                                                             |
| offsets.commit.timeout.ms    | Specifies the timeout for the Offset to submit requests. Unit: millisecond                                                                                                                                                                                                            | 5000          | This parameter specifies the maximum delay for processing an Offset request.                                                                                                                                       |
| replica.socket.timeout.ms    | Specifies the timeout of the request for synchronizing replica data. Its value must be greater than or equal to that of the <b>replica.fetch.wait.max.ms</b> parameter. Unit: millisecond                                                                                             | 30000         | Specifies the maximum timeout for establishing a channel before the synchronization thread sends a synchronization request. The value must be greater than that of the <b>replica.fetch.wait.max.ms</b> parameter. |
| request.timeout.ms           | Specifies the timeout for waiting for a response after the client sends a connection request. If no response is received within the timeout, the client resends the request. A request failure is returned after the maximum retry times is reached. Unit: millisecond                | 30000         | This parameter is configured when the networkclient connection is transferred in the controller and replica threads on the broker node.                                                                            |
| transaction.max.timeout.ms   | Specifies the maximum timeout allowed by the transaction. If the client request time exceeds the value of this parameter, broker returns an error in InitProducerIdRequest. This prevents a long client request timeout, ensuring that consumer can receive topics. Unit: millisecond | 900000        | Specifies the maximum timeout for transactions.                                                                                                                                                                    |

| Parameter                        | Description                                                                             | Default Value | Impact                                                                                                                                                                                                                                                |
|----------------------------------|-----------------------------------------------------------------------------------------|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| user.group.cache.timeout.seconds | Specifies the time when the user group information is stored in the cache. Unit: second | 300           | Specifies the time for caching the mapping between users and user groups. If time exceeds the threshold, the system automatically runs the <b>id -Gn</b> command to query the user information. During this period, the mapping in the cache is used. |
| zookeeper.connection.timeout.ms  | Specifies the timeout for connecting to ZooKeeper. Unit: millisecond                    | 45,000        | This parameter specifies the duration for connecting the ZooKeeper and zkclient for the first time. If the duration exceeds the value of this parameter, the zkclient automatically disconnects the connection.                                       |

| Parameter                    | Description                                                                                                                                                                         | Default Value | Impact                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| zookeeper.session.timeout.ms | Specifies the ZooKeeper session timeout duration. During this period, ZooKeeper disconnects the connection if broker does not report its heartbeats to ZooKeeper. Unit: millisecond | 45,000        | ZooKeeper session timeout has the following functions:<br>1) Based on value of this parameter and the number of ZooKeeper URLs in ZKURL, if the connection duration exceeds the node timeout value (sessionTimeout/ Number of transferred ZooKeeper URLs), the connection fails and the system attempts to connect to the next node.<br>2) After the connection is established, a session (for example, the temporary BrokerId node registered on the ZooKeeper) is cleared by the ZooKeeper a session timeout later if the broker is stopped. |

**Table 14-6** Producer-related timeout parameters

| Parameter          | Description                                 | Default Value | Impact                                                                                                               |
|--------------------|---------------------------------------------|---------------|----------------------------------------------------------------------------------------------------------------------|
| request.timeout.ms | Specifies the timeout of a message request. | 30,000        | If a network fault occurs, increase the value of this parameter. If the value is too small, the Batch Expire occurs. |



**Table 14-7** Consumer-related timeout parameters

| Parameter               | Description                                                  | Default Value | Impact                                                                                                                                        |
|-------------------------|--------------------------------------------------------------|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| connections.max.idle.ms | Specifies the maximum retention period for idle connections. | 600,000       | If the idle connection time is greater than this parameter value, this connection is disconnected. If necessary, a new connection is created. |
| request.timeout.ms      | Specifies the timeout for consumer requests.                 | 30,000        | If the request times out, the request will fail and be sent again.                                                                            |

## 14.9 Safety Instructions on Using Kafka

This section applies to MRS 3.x or later.

### Brief Introduction to Kafka APIs

- **Producer API**  
Indicates the API defined in **org.apache.kafka.clients.producer.KafkaProducer**. When **kafka-console-producer.sh** is used, the API is used by default.
- **Consumer API**  
Indicates the API defined in **org.apache.kafka.clients.consumer.KafkaConsumer**. When **kafka-console-consumer.sh** is used, the API is used by default.

 **NOTE**

In versions later than MRS 3.x, Kafka does not support old Producer APIs and Consumer APIs.

### Protocol Description for Accessing Kafka

The protocols used to access Kafka are as follows: PLAINTEXT, SSL, SASL\_PLAINTEXT, and SASL\_SSL.

When Kafka service is started, the listeners using the PLAINTEXT and SASL\_PLAINTEXT protocols are started. You can set **ssl.mode.enable** to **true** in Kafka service configuration to start listeners using SSL and SASL\_SSL protocols. The following table describes the four protocols:

| Protocol       | Description                                                 | Default Port |
|----------------|-------------------------------------------------------------|--------------|
| PLAINTEXT      | Supports plaintext access without authentication.           | 9092         |
| SASL_PLAINTEXT | Supports plaintext access with Kerberos authentication.     | 21007        |
| SSL            | Supports SSL-encrypted access without authentication.       | 9093         |
| SASL_SSL       | Supports SSL-encrypted access with Kerberos authentication. | 21009        |

## ACL Settings for a Topic

To view and set topic permission information, run the `kafka-acls.sh` script on the Linux client. For details, see [Managing Kafka User Permissions](#).

## Use of Kafka APIs in Different Scenarios

- Scenario 1: accessing the topic with an ACL

| Used API | User Group                                                                                                                                                                                                                                                                 | Client Parameter                                                                    | Server Parameter                             | Accessed Port                                   |
|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|----------------------------------------------|-------------------------------------------------|
| API      | Users need to meet one of the following conditions: <ul style="list-style-type: none"> <li>In the administrator group</li> <li>In the <b>kafkaadmin</b> group</li> <li>In the <b>kafka_superuser</b> group</li> <li>In the <b>kafka</b> group and be authorized</li> </ul> | security.inter.broker.protocol=SASL_PLAINTEXT<br>sasl.kerberos.service.name = kafka | -                                            | sasl.port<br>(The default number is 21007.)     |
|          |                                                                                                                                                                                                                                                                            | security.protocol=SASL_SSL<br>sasl.kerberos.service.name = kafka                    | Set <b>ssl.mode.enabled</b> to <b>true</b> . | sasl-ssl.port<br>(The default number is 21009.) |

- Scenario 2: accessing the topic without an ACL

| Used API | User Group                                                                                                                                                                                                                  | Client Parameter                                                               | Server Parameter                                                                                                                                                                                 | Accessed Port                                           |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------|
| API      | <p>Users need to meet one of the following conditions:</p> <ul style="list-style-type: none"> <li>In the administrator group</li> <li>In the <b>kafkaadmin</b> group</li> <li>In the <b>kafkasuperuser</b> group</li> </ul> | <p>security.protocol=SASL_PLAINTEXT<br/>sasl.kerberos.service.name = kafka</p> | -                                                                                                                                                                                                | <p>sasl.port<br/>(The default number is 21007.)</p>     |
|          | <p>Users are in the <b>kafka</b> group.</p>                                                                                                                                                                                 |                                                                                | <p>Set <b>allow.everyone.if.no.acl.found</b> to <b>true</b>.</p> <p><b>NOTE</b><br/>In normal mode, the server parameter <b>allow.everyone.if.no.acl.found</b> does not need to be modified.</p> | <p>sasl.port<br/>(The default number is 21007.)</p>     |
|          | <p>Users need to meet one of the following conditions:</p> <ul style="list-style-type: none"> <li>In the administrator group</li> <li>In the <b>kafkaadmin</b> group</li> <li>In the <b>kafkasuperuser</b> group</li> </ul> | <p>security.protocol=SASL_SSL<br/>sasl.kerberos.service.name = kafka</p>       | <p>Set <b>ssl.mode.enabled</b> to <b>true</b>.</p>                                                                                                                                               | <p>sasl-ssl.port<br/>(The default number is 21009.)</p> |

| Used API | User Group                           | Client Parameter            | Server Parameter                                                                                                                                                      | Accessed Port                               |
|----------|--------------------------------------|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------|
|          | Users are in the <b>kafka</b> group. |                             | <ol style="list-style-type: none"> <li>1. Set <b>allow.everyone.if.no.acl.found</b> to <b>true</b>.</li> <li>2. Set <b>ssl.mode.enable</b> to <b>true</b>.</li> </ol> | ssl-ssl.port (The default number is 21009.) |
|          | -                                    | security.protocol=PLAINTEXT | Set <b>allow.everyone.if.no.acl.found</b> to <b>true</b> .                                                                                                            | port (The default number is 9092.)          |
|          | -                                    | security.protocol=SSL       | <ol style="list-style-type: none"> <li>1. Set <b>allow.everyone.if.no.acl.found</b> to <b>true</b>.</li> <li>2. Set <b>ssl.mode.enable</b> to <b>true</b>.</li> </ol> | ssl.port (The default number is 9063.)      |

## 14.10 Kafka Specifications

This section applies to MRS 3.x or later.

### Upper Limit of Topics

The maximum number of topics depends on the number of file handles (mainly used by data and index files on site) opened in the process.

1. Run the **ulimit -n** command to view the maximum number of file handles that can be opened in the process.
2. Run the **lsof -p <Kafka PID>** command to view the file handles (which may keep increasing) that are opened in the Kafka process on the current single node.
3. Determine whether the maximum number of file handles will be reached and whether the running of Kafka is affected after required topics are created, and estimate the maximum size of data that each partition folder can store and the number of data (\*.log file, whose default size is 1 GB and can be adjusted by modifying **log.segment.bytes**) and index (\*.index file, whose default size is 10 MB and can be adjusted by modifying **log.index.size.max.bytes**) files that will be produced after required topics are created.

## Number of Concurrent Consumers

In an application, it is recommended that the number of concurrent consumers in a group be the same as the number of partitions in a topic, ensuring that a consumer consumes data in only a specified partition. If the number of concurrent consumers is more than the number of partitions, the redundant consumers have no data to consume.

## Relationship Between Topic and Partition

- If  $K$  Kafka nodes are deployed in the cluster, each node is configured with  $N$  disks, the size of each disk is  $M$ , the cluster contains  $n$  topics (named as  $T_1, T_2, \dots, T_n$ ), the data input traffic per second of the  $m$  topic is  $X(T_m)$  MB/s, the number of configured replicas is  $R(T_m)$ , and the configured data retention time is  $Y(T_m)$  hour, the following requirement must be met:

$$M \times N \times K > \sum_{i=T_1}^{T_n} (X(i)R(i)Y(i) \times 3600)$$

- If the size of a disk is  $M$ , the disk has  $n$  partitions (named as  $P_0, P_1, \dots, P_n$ ), the data write traffic per second of the  $m$  partition is  $Q(P_m)$  MB/s (calculation method: data traffic of the topic to which the  $m$  partition belongs divided by the number of partitions), and the data retention time is  $T(P_m)$  hours, the following requirement must be met for the disk:

$$M > \sum_{i=P_0}^{P_n} (Q(i)T(i) \times 3600)$$

- Based on the throughput, if the throughput that can be reached by the producer is  $P$ , the throughput that can be reached by the consumer is  $C$ , and the expected throughput of Kafka is  $T$ , it is recommended that the number of partitions of the topic be set to  $\text{Max}(T/P, T/C)$ .

### NOTE

- In a Kafka cluster, more partitions mean higher throughput. However, too many partitions also pose potential impacts, such as a file handle increase, unavailability increase (for example, if a node is faulty, the time window becomes large after the leader is reselected in some partitions), and end-to-end latency increase.
- Suggestion: The disk usage of a partition is smaller than or equal to 100 GB; the number of partitions on a node is smaller than or equal to 3,000; the number of partitions in the entire cluster is smaller than or equal to 10,000.

## 14.11 Using the Kafka Client

### Scenario

This section guides users to use a Kafka client in an O&M or service scenario.

This section applies to MRS 3.x or later.

## Prerequisites

- The client has been installed. For example, the installation directory is **/opt/client**.
- Service component users are created by the MRS cluster administrator as required. Machine-machine users need to download the keytab file. A human-machine user must change the password upon the first login. (Not involved in normal mode)
- After changing the domain name of a cluster, redownload the client to ensure that the **kerberos.domain.name** value in the configuration file of the client is set to the correct server domain name.

## Procedure

**Step 1** Log in to the node where the client is installed as the client installation user.

**Step 2** Run the following command to go to the client installation directory:

```
cd /opt/client
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 4** Run the following command to perform user authentication (skip this step in normal mode):

```
kinit Component service user
```

**Step 5** Run the following command to switch to the Kafka client installation directory:

```
cd Kafka/kafka/bin
```

**Step 6** Run the following command to use the client tool to view and use the help information:

- **./kafka-console-consumer.sh**: Kafka message reading tool
- **./kafka-console-producer.sh**: Kafka message publishing tool
- **./kafka-topics.sh**: Kafka topic management tool

----End

## 14.12 Configuring Kafka HA and High Reliability Parameters

### Scenario

For the Kafka message transmission assurance mechanism, different parameters are available for meeting different performance and reliability requirements. This section describes how to configure Kafka high availability (HA) and high reliability parameters.

This section applies to MRS 3.x or later.

## Impact on the System

- Impact of HA and high performance configurations:

### NOTICE

After HA and high performance are configured, the data reliability decreases. Specifically, data may be lost if disks or nodes are faulty.

- Impact of high reliability configurations:
  - Deteriorated performance  
If **ack** is set to **-1**, data written is considered as successful only when data is written to multiple replicas. As a result, the delay of a single message increases and the client processing capability decreases. The impact is subject to the actual test data.
  - Reduced availability  
A replica that is not in the ISR list cannot be elected as a leader. If the leader goes offline and other replicas are not in the ISR list, the partition remains unavailable until the leader node recovers. When the node where a replica of a partition is located is faulty, the minimum number of successful replicas cannot be met. As a result, service writing fails.
- If parameters are at the service level, Kafka needs to be restarted. You are advised to modify the service-level configuration in the change window.

## Parameter Description

- If services require high availability and high performance, set the parameters listed in [Table 14-8](#) on the server. For details about the parameter configuration entry, see [Modifying Cluster Service Configuration Parameters](#).

**Table 14-8** Server HA and high performance parameters

| Parameter                      | Default Value | Description                                                                                                                                                                                                                                                |
|--------------------------------|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| unclean.leader.election.enable | true          | Specifies whether a replica that is not in the ISR can be selected as the leader. If this parameter is set to <b>true</b> , data may be lost.                                                                                                              |
| auto.leader.rebalance.enable   | true          | Specifies whether the leader automated balancing function is used.<br>If this parameter is set to <b>true</b> , the controller periodically balances the leader of each partition on all nodes and assigns the leader to a replica with a higher priority. |

| Parameter           | Default Value | Description                                                                                                              |
|---------------------|---------------|--------------------------------------------------------------------------------------------------------------------------|
| min.insync.replicas | 1             | Specifies the minimum number of replicas to which data is written when <b>acks</b> is set to <b>-1</b> for the Producer. |

Set the parameters listed in [Table 14-9](#) in the client configuration file **producer.properties**. The path for storing **producer.properties** is **/opt/client/Kafka/kafka/config/producer.properties**, where **/opt/client** indicates the installation directory of the Kafka client.



**Table 14-9** Client HA and high performance parameters

| Parameter | Default Value | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| acks      | 1             | <p>The leader needs to check whether the message has been received and determine whether the required operation has been processed. This parameter affects message reliability and performance.</p> <ul style="list-style-type: none"> <li>• If this parameter is set to <b>0</b>, the producer does not wait for any response from the server, and the message is considered successful.</li> <li>• If this parameter is set to <b>1</b>, when the leader of the replica verifies that data has been written into the cluster, the leader returns a response without waiting for data to be written to all replicas. In this case, if the leader is abnormal when the leader makes the confirmation but replica synchronization is not complete, data will be lost.</li> <li>• If this parameter is set to <b>-1</b>, the message is considered to be successfully received only when all synchronized replicas are confirmed. If the <b>min.insync.replicas</b></li> </ul> |

| Parameter | Default Value | Description                                                                                                                                             |
|-----------|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
|           |               | parameter is also configured, data can be written into multiple replicas. In this case, records will not be lost as long as one replica remains active. |

- To ensure high data reliability for services, set the parameters listed in [Table 14-10](#) on the server. For details about the parameter configuration entry, see [Modifying Cluster Service Configuration Parameters](#).

**Table 14-10** Server HA parameters

| Parameter                      | Recommended Value | Description                                                                                                                                                                                                                                  |
|--------------------------------|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| unclean.leader.election.enable | false             | A replica that is not in the ISR list cannot be elected as a leader.                                                                                                                                                                         |
| min.insync.replicas            | 2                 | Specifies the minimum number of replicas to which data is written when <b>acks</b> is set to <b>-1</b> for the Producer.<br>Ensure that the value of <b>min.insync.replicas</b> is equal to or less than that of <b>replication.factor</b> . |

Set the parameters listed in [Table 14-11](#) in the client configuration file **producer.properties**. The path for storing **producer.properties** is **/opt/client/Kafka/kafka/config/producer.properties**, where **/opt/client** indicates the installation directory of the Kafka client.

**Table 14-11** Server HA parameters

| Parameter | Recommended Value | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-----------|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| acks      | -1                | <p>The leader needs to check whether the message has been received and determine whether the required operation has been processed.</p> <p>If this parameter is set to <b>-1</b>, the message is considered to be successfully received only when all replicas in the ISR list have confirmed to receive the message. This parameter is used along with <b>min.insync.replicas</b> to ensure that multiple copies are successfully written. As long as one copy is active, the record will not be lost. If this parameter is set to <b>-1</b>, the production performance deteriorates. Therefore, you need to set this parameter based on the actual situation.</p> |

## Configuration Suggestions

Configure parameters based on requirements on reliability and performance in the following service scenarios:

- For valued data, you are advised to configure RAID1 or RAID5 for Kafka data directory disks to improve data reliability when a single disk is faulty.
- For parameters that can be modified at the topic level, the service level configurations are used by default.

These parameters can be separately configured based on topic reliability requirements. For example, log in to the Kafka client as user **root**, and run the following command to configure the reliability parameter with topic named test in the client installation directory:

```
cd Kafka/kafka/bin
```

```
kafka-topics.sh --zookeeper 192.168.1.205:240022181/kafka --alter --topic test --config unclean.leader.election.enable=false --config min.insync.replicas=2
```

**192.168.1.205** indicates the ZooKeeper service IP address.

- If parameters are at the service level, Kafka needs to be restarted. You are advised to modify the service-level configuration in the change window.

## 14.13 Changing the Broker Storage Directory

### Scenario

This section applies to MRS 3.x or later.

When a broker storage directory is added, the MRS cluster administrator needs to change the broker storage directory on FusionInsight Manager, to ensure that the Kafka can work properly. The new topic partition will be generated in the directory that has fewest partitions. Changing the ZooKeeper storage directory includes the following scenarios:

#### NOTE

Because Kafka does not detect disk capacity, ensure that the disk quantity and capacity configured for each Broker instance are the same.

- Change the storage directory of the Broker role. In this way, the storage directories of all Broker instances are changed.
- Change the storage directory of a single Broker instance. In this way, only the storage directory of this Broker instance is changed, and the storage directories of other Broker instances remain the same.

### Impact on the System

- Changing the Broker role storage directory requires the restart of services. The services cannot be accessed during the restart.
- The storage directory of a single Broker instance can be changed only after the instance is restarted. The instance cannot provide services during the restart.
- The directory for storing service parameter configurations must also be updated.

### Prerequisites

- New disks have been prepared and installed on each data node, and the disks are formatted.
- The Kafka client has been installed.
- When you change the storage directory of a single Broker instance, the number of active Broker instances must be greater than the number of backups specified during topic creation.

### Procedure

#### Changing the storage directory of the Kafka role

- Step 1** Log in as user **root** to each node on which the Kafka service is installed, and perform the following operations:

1. Create a target directory.  
For example, to create the target directory `${BIGDATA_DATA_HOME}/kafka/data2`, run the following command:  

```
mkdir ${BIGDATA_DATA_HOME}/kafka/data2
```
2. Mount the directory to the new disk. For example, mount `${BIGDATA_DATA_HOME}/kafka/data2` to the new disk.
3. Modify permissions on the new directory.  
For example, to modify permissions on the `${BIGDATA_DATA_HOME}/kafka/data2` directory, run the following commands:  

```
chmod 700 ${BIGDATA_DATA_HOME}/kafka/data2 -R and chown omm:wheel ${BIGDATA_DATA_HOME}/kafka/data2 -R
```

**Step 2** Log in to FusionInsight Manager for clusters of MRS 3.x or later and choose **Cluster > Services > Kafka > Configurations**.

**Step 3** Add a new directory to the end of the default value of **log.dirs**.

Enter **log.dirs** in the search box and add the new directory to the end of the default value of the **log.dirs** configuration item. Use commas (,) to separate multiple directories. For example:

```
${BIGDATA_DATA_HOME}/kafka/data1/kafka-logs,${BIGDATA_DATA_HOME}/kafka/data2/kafka-logs
```

**Step 4** Click **Save**, and then click **OK**. When **Operation succeeded** is displayed, click **Finish**.

**Step 5** Choose **Cluster > Services > Kafka**. In the upper right corner, choose **More > Restart Service** to restart the Kafka service.

### Changing the storage directory of a single Kafka instance

**Step 6** Log in to the Broker node as user **root** and perform the following operations:

1. Create a target directory.  
For example, to create the target directory `${BIGDATA_DATA_HOME}/kafka/data2`, run the following command:  

```
mkdir ${BIGDATA_DATA_HOME}/kafka/data2
```
2. Mount the directory to the new disk. For example, mount `${BIGDATA_DATA_HOME}/kafka/data2` to the new disk.
3. Modify permissions on the new directory.  
For example, to modify permissions on the `${BIGDATA_DATA_HOME}/kafka/data2` directory, run the following commands:  

```
chmod 700 ${BIGDATA_DATA_HOME}/kafka/data2 -R and chown omm:wheel ${BIGDATA_DATA_HOME}/kafka/data2 -R
```

**Step 7** Log in to FusionInsight Manager for MRS 3.x or later, and choose **Cluster > Services > Kafka > Instance**.

**Step 8** Click the specified broker instance and switch to **Instance Configurations**.

Enter **log.dirs** in the search box and add the new directory to the end of the default value of the **log.dirs** configuration item. Use commas (,) to separate

multiple directories, for example, `${BIGDATA_DATA_HOME}/kafka/data1/kafka-logs`, `${BIGDATA_DATA_HOME}/kafka/data2/kafka-logs`.

**Step 9** Click **Save**, and then click **OK**. A message is displayed, indicating that the operation is successful. Click **Finish**.

**Step 10** On the Broker instance page, choose **More > Restart Instance** to restart the Broker instance.

----End

## 14.14 Checking the Consumption Status of Consumer Group

### Scenario

This section describes how to view the current expenditure on the client based on service requirements.

This section applies to MRS 3.x or later.

### Prerequisites

- You have understood service requirements and prepared a system user.
- The Kafka client has been installed.

### Procedure

**Step 1** Log in as a client installation user to the node on which the Kafka client is installed.

**Step 2** Switch to the Kafka client installation directory, for example, `/opt/kafkaclient`.  
**cd /opt/kafkaclient**

**Step 3** Run the following command to configure environment variables:  
**source bigdata\_env**

**Step 4** Run the following command to perform user authentication (skip this step in normal mode):  
**kinit Component service user**

**Step 5** Run the following command to switch to the Kafka client installation directory:  
**cd Kafka/kafka/bin**

**Step 6** Run the `kafka-consumer-groups.sh` command to check the current consumption status.

- Check the Consumer Group list on Kafka saved by Offset:  
**./kafka-consumer-groups.sh --list --bootstrap-server <Service IP address of any broker node:21007> --command-config ../config/consumer.properties**  
eg: `./kafka-consumer-groups.sh --bootstrap-server 192.168.1.1:21007 --list --command-config ../config/consumer.properties`

- Check the consumption status of Consumer Group on Kafka saved by Offset:  
**./kafka-consumer-groups.sh --describe --bootstrap-server <Service IP address of any broker node:21007> --group Consumer group name --command-config ../config/consumer.properties**  
eg:./kafka-consumer-groups.sh --describe --bootstrap-server 192.168.1.1:21007 --group example-group --command-config ../config/consumer.properties

---

#### NOTICE

1. Ensure that the current consumer is online and consumes data.
2. Configure the **group.id** in the **consumer.properties** configuration file and **--group** in the command to the group to be queried.
3. The Kafka cluster's IP port number is 21007 in security mode and 9092 in normal mode.

---

----End

## 14.15 Kafka Balancing Tool Instructions

### Scenario

This section describes how to use the Kafka balancing tool on a client to balance the load of the Kafka cluster based on service requirements in scenarios such as node decommissioning, node recommissioning, and load balancing.

This section applies to MRS 3.x or later.

### Prerequisites

- You have understood service requirements and prepared a Kafka administrator (belonging to the **kafkaadmin** group. It is not required for the normal mode.).
- The Kafka client has been installed.

### Procedure

- Step 1** Log in as a client installation user to the node on which the Kafka client is installed.
- Step 2** Switch to the Kafka client installation directory, for example, **/opt/kafkaclient**.
- Step 3** Run the following command to configure environment variables:  
**source bigdata\_env**
- Step 4** Run the following command to authenticate the user (skip this step in normal mode):  
**kinit Component service user**

**Step 5** Run the following command to switch to the Kafka client installation directory:

```
cd Kafka/kafka
```

**Step 6** Run the **kafka-balancer.sh** command to balance user cluster. The commonly used commands are:

- Run the **--run** command to perform cluster balancing:

```
./bin/kafka-balancer.sh --run --zookeeper <ZooKeeper service IP address of any ZooKeeper node:zkPort/kafka> --bootstrap-server <Kafka cluster IP:port> --throttle 1000000 --consumer-config config/consumer.properties --enable-az-aware --show-details
```

This command consists of generation and execution of the balancing solution. **--show-details** is optional, indicating whether to print the solution details. **--throttle** indicates the bandwidth limit during the execution of the balancing solution. The unit is bytes per second (bytes/sec). **--enable-az-aware** indicates that the cross-AZ feature is enabled when the balancing solution is generated. When this parameter is used, ensure that the cross-AZ feature has been enabled for the cluster.

- Run the **--run** command to decommission a node:

```
./bin/kafka-balancer.sh --run --zookeeper <Service IP address of any ZooKeeper node:zkPort/kafka> --bootstrap-server <Kafka cluster IP address:port> --throttle 1000000 --consumer-config config/consumer.properties --remove-brokers <BrokerId list> --enable-az-aware --force
```

In the command, **--remove-brokers** indicates the list of broker IDs to be deleted. Multiple broker IDs are separated by commas (.). **--force** is optional, indicating that the disk usage alarm is ignored and the migration solution is forcibly generated. **-enable-az-aware** is optional, indicating that the cross-AZ feature is enabled when the balancing solution is generated. When this parameter is used, ensure that the cross-AZ feature has been enabled for the cluster.

- Run the following command to view the execution status:

```
./bin/kafka-balancer.sh --status --zookeeper <Service IP address of any ZooKeeper node:zkPort/kafka>
```

- Run the following command to generate a balancing solution:

```
./bin/kafka-balancer.sh --generate --zookeeper <Service IP address of any ZooKeeper node:zkPort/kafka> --bootstrap-server <Kafka cluster IP address:port> --consumer-config config/consumer.properties --enable-az-aware
```

This command is used to generate a migration solution based on the current cluster status and print the solution to the console. **--enable-az-aware** is optional, indicating that the cross-AZ feature is enabled when a migration solution is generated. If this parameter is used, ensure that the cross-AZ feature has been enabled for the cluster.

- Clearing the intermediate status

```
./bin/kafka-balancer.sh --clean --zookeeper <Service IP address of any ZooKeeper node:zkPort/kafka>
```

This command is used to clear the intermediate status information on the ZooKeeper when the migration is not complete.



---

**NOTICE**

The port number of the Kafka cluster's IP address is 21007 in security mode and 9092 in normal mode.

---

----End

## Troubleshooting

During partition migration using the Kafka balancing tool, if the execution progress of the balancing tool is blocked due to a Broker fault in the cluster, you need to manually rectify the fault. The scenarios are as follows:

- The Broker is faulty because the disk usage reaches 100%.
  - a. Log in to FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Kafka** > **Instance**, stop the Broker instance in the **Restoring** state, and record the management IP address of the node where the instance resides and the corresponding **broker.id**. You can click the role name to view the value, on the **Instance Configurations** page, select **All Configurations** and search for the **broker.id** parameter.
  - b. Log in to the recorded management IP address as user **root**, and run the **df -lh** command to view the mounted directory whose disk usage is 100%, for example, **`\${BIGDATA\_DATA\_HOME}/kafka/data1**.
  - c. Go to the directory, run the **du -sh \*** command to view the size of each file in the directory, Check whether files other than files in the **kafka-logs** directory exist, and determine whether these files can be deleted or migrated.
    - If yes, delete or migrate the related data and go to **8**.
    - If no, go to **4**.
  - d. Go to the **kafka-logs** directory, run the **du -sh \*** command, select a partition folder to be moved. The naming rule is **Topic name-Partition ID**. Record the topic and partition.
  - e. Modify the **recovery-point-offset-checkpoint** and **replication-offset-checkpoint** files in the **kafka-logs** directory in the same way.
    - i. Decrease the number in the second line in the file. (To remove multiple directories, the number deducted is equal to the number of files to be removed.
    - ii. Delete the line of the to-be-removed partition. (The line structure is "*Topic name Partition ID Offset*". Save the data before deletion. Subsequently, the content must be added to the file of the same name in the destination directory.)
  - f. Modify the **recovery-point-offset-checkpoint** and **replication-offset-checkpoint** files in the destination data directory (for example, **`\${BIGDATA\_DATA\_HOME}/kafka/data2/kafka-logs**) in the same way.
    - Increase the number in the second line in the file. (To move multiple directories, the number added is equal to the number of files to be moved.

- Add the to-be moved partition to the end of the file. (The line structure is "*Topic name Partition ID Offset*". You can copy the line data saved in 5.)
- g. Move the partition to the destination directory. After the partition is moved, run the **chown omm:wheel -R *Partition directory*** command to modify the directory owner group for the partition.
- h. Log in to FusionInsight Manager and choose **Cluster > Name of the desired cluster > Services > Kafka > Instance** to start the stopped Broker instance.
- i. Wait for 5 to 10 minutes and check whether the health status of the Broker instance is **Good**.
  - If yes, resolve the disk capacity insufficiency problem according to the handling method of "ALM-38001 Insufficient Kafka Disk Capacity" after the alarm is cleared.
  - If no, contact O&M support.

After the faulty Broker is recovered, the blocked balancing task continues. You can run the **--status** command to view the task execution progress.

- The Broker fault occurs because of other causes, the fault scenario is clear, and the fault can be rectified within a short period of time.
  - a. Restore the faulty Broker according to the root cause.
  - b. After the faulty Broker is recovered, the blocked balancing task continues. You can run the **--status** command to view the task execution progress.
- The Broker fault occurs because of other causes, the fault scenario is complex, and the fault cannot be rectified within a short period of time.
  - a. Run the **kinit *Kafka administrator account*** command (skip this step in normal mode).
  - b. Run the **zkCli.sh -server <ZooKeeper cluster service IP address.zkPort/ kafka>** command to log in to ZooKeeper Shell.
  - c. Run the **addauth krbgroup** command (skip this step in normal mode).
  - d. Delete the **/admin/reassign\_partitions** and **/controller** directories.
  - e. Perform the preceding steps to forcibly stop the migration. After the cluster recovers, run the **kafka-reassign-partitions.sh** command to delete redundant copies generated during the intermediate process.

## 14.16 Balancing Data After Kafka Node Scale-Out

### Scenario

This section describes how to use the Kafka balancing tool on the client to balance the load of the Kafka cluster after Kafka nodes are scaled out.

This section applies to versions earlier than MRS 3.x.

## Prerequisites

- You have understood service requirements and prepared a Kafka administrator (belonging to the **kafkaadmin** group and not required for the normal mode).
- The Kafka client has been installed, for example, in the **/opt/kafkaclient** directory.
- Two topics named **test\_2** and **test\_3** has been created by referring to **Step 7**. The **move-kafka-topic.json** file has been created in the **/opt/kafkaclient/Kafka/kafka** directory. The topic format is as follows:

```
{
 "topics":
 [{"topic":"test_2"}, {"topic":"test_3"}],
 "version":1
}
```

## Procedure

**Step 1** Log in to the node where the Kafka client is installed as the client installation user.

**Step 2** Run the following command to switch to the client installation directory:

```
cd /opt/kafkaclient
```

**Step 3** Run the following command to set environment variables:

```
source bigdata_env
```

**Step 4** Run the following command to perform user authentication (skip this step if the cluster is in normal mode):

```
kinit Component service user
```

**Step 5** Run the following command to go to the **bin** directory of the Kafka client:

```
cd Kafka/kafka/bin
```

**Step 6** Run the following command to generate an execution plan:

```
./kafka-reassign-partitions.sh --zookeeper 172.16.0.119:2181/kafka --topics-to-move-json-file ../move-kafka-topic.json --broker-list "1,2,3" --generate
```

### NOTE

- **172.16.0.119**: service IP address of the ZooKeeper instance
- **--broker-list "1,2,3"**: list of broker instances. **1,2,3** indicates all broker IDs after a scale-out.

```
[root@node-master1SPXC bin]# ./kafka-reassign-partitions.sh --zookeeper 172.16.0.119:2181/kafka --topics-to-move-json-file ../move-kafka-topic.json --broker-list "1,2,3" --generate
Current partition replica assignment
{"version":1,"partitions":[{"topic":"test_2","partition":3,"replicas":["any","any"]}, {"topic":"test_2","partition":4,"replicas":["any","any"]}, {"topic":"test_2","partition":5,"replicas":["any","any"]}, {"topic":"test_2","partition":6,"replicas":["any","any"]}, {"topic":"test_3","partition":0,"replicas":["any","any"]}, {"topic":"test_3","partition":1,"replicas":["any","any"]}, {"topic":"test_3","partition":2,"replicas":["any","any"]}, {"topic":"test_3","partition":3,"replicas":["any","any"]}, {"topic":"test_3","partition":4,"replicas":["any","any"]}, {"topic":"test_3","partition":5,"replicas":["any","any"]}, {"topic":"test_3","partition":6,"replicas":["any","any"]}]}
Proposed partition reassignment configuration
{"version":1,"partitions":[{"topic":"test_3","partition":0,"replicas":["any","any"]}, {"topic":"test_2","partition":1,"replicas":["any","any"]}, {"topic":"test_2","partition":2,"replicas":["any","any"]}, {"topic":"test_2","partition":3,"replicas":["any","any"]}, {"topic":"test_2","partition":4,"replicas":["any","any"]}, {"topic":"test_2","partition":5,"replicas":["any","any"]}, {"topic":"test_2","partition":6,"replicas":["any","any"]}, {"topic":"test_3","partition":1,"replicas":["any","any"]}, {"topic":"test_3","partition":2,"replicas":["any","any"]}, {"topic":"test_3","partition":3,"replicas":["any","any"]}, {"topic":"test_3","partition":4,"replicas":["any","any"]}, {"topic":"test_3","partition":5,"replicas":["any","any"]}, {"topic":"test_3","partition":6,"replicas":["any","any"]}]}
[root@node-master1SPXC bin]#
```

**Step 7** Run the `vim ../reassignment.json` command to create the `reassignment.json` file and save it to the `/opt/kafkaclient/Kafka/kafka` directory.

Copy the content under **Proposed partition reassignment configuration** generated in **Step 6** to the `reassignment.json` file, as shown in the follows:

```
{
 "version": 1,
 "partitions": [
 {
 "topic": "test",
 "partition": 4,
 "replicas": [1, 2],
 "log_dirs": ["any", "any"]
 },
 {
 "topic": "test",
 "partition": 1,
 "replicas": [1, 3],
 "log_dirs": ["any", "any"]
 },
 {
 "topic": "test",
 "partition": 3,
 "replicas": [3, 1],
 "log_dirs": ["any", "any"]
 },
 {
 "topic": "test",
 "partition": 0,
 "replicas": [3, 2],
 "log_dirs": ["any", "any"]
 },
 {
 "topic": "test",
 "partition": 2,
 "replicas": [2, 1],
 "log_dirs": ["any", "any"]
 }
]
}
```

**Step 8** Run the following command to redistribute partitions:

```
./kafka-reassign-partitions.sh --zookeeper 172.16.0.119:2181/kafka --
reassignment-json-file ../reassignment.json --execute --throttle 50000000
```

**NOTE**

**--throttle 50000000**: The maximum bandwidth is 50 MB/s. You can change the bandwidth based on the data volume and the customer's requirements on the balancing time. If the data volume is 5 TB, the bandwidth is 50 MB/s and the data balancing takes about 8 hours.

```
[root@node-master1SPXC bin]# vim ../reassignment.json
[root@node-master1SPXC bin]# ./kafka-reassign-partitions.sh --zookeeper 172.16.0.119:2181/kafka --reassignment-json-file ../reassignment.json --execute --throttle 50000000
Current partition replica assignment

{"version":1,"partitions":[{"topic":"test_2","partition":3,"replicas":[1,2],"log_dirs":["any","any"]},{"topic":"test_2","partition":4,"replicas":[2,1],"log_dirs":["any","any"]},{"topic":"test_3","partition":5,"replicas":[2,1],"log_dirs":["any","any"]},{"topic":"test_3","partition":3,"replicas":[2,1],"log_dirs":["any","any"]},{"topic":"test_2","partition":2,"replicas":[2,1],"log_dirs":["any","any"]},{"topic":"test_3","partition":2,"replicas":[1,2],"log_dirs":["any","any"]},{"topic":"test_3","partition":0,"replicas":[1,2],"log_dirs":["any","any"]},{"topic":"test_2","partition":6,"replicas":[2,1],"log_dirs":["any","any"]},{"topic":"test_3","partition":4,"replicas":[1,2],"log_dirs":["any","any"]},{"topic":"test_2","partition":0,"replicas":[2,1],"log_dirs":["any","any"]},{"topic":"test_3","partition":1,"replicas":[2,1],"log_dirs":["any","any"]},{"topic":"test_2","partition":1,"replicas":[1,2],"log_dirs":["any","any"]},{"topic":"test_2","partition":5,"replicas":[1,2],"log_dirs":["any","any"]},{"topic":"test_3","partition":6,"replicas":[1,2],"log_dirs":["any","any"]}]}

Save this to use as the --reassignment-json-file option during rollback
Warning: You must run Verify periodically, until the reassignment completes, to ensure the throttle is removed. You can also alter the throttle by rerunning the Execute command passing a new value.
The inter-broker throttle limit was set to 50000000 B/s
Successfully started reassignment of partitions.
[root@node-master1SPXC bin]#
```

**Step 9** Run the following command to check the data migration status:

```
./kafka-reassign-partitions.sh --zookeeper 172.16.0.119:2181/kafka --
reassignment-json-file ../reassignment.json --verify
```

```
drwx----- 2 omm wheel 4096 Sep 14 21:12 test_3-5
drwx----- 2 omm wheel 4096 Sep 14 21:12 test_3-6
[root@node-str-coreRuzk0001 kafka-logs]# ll
total 56
-rw----- 1 omm wheel 4 Sep 14 21:30 cleaner-offset-check
-rw----- 1 omm wheel 4 Sep 14 21:31 log-start-offset-check
-rw----- 1 omm wheel 54 Sep 14 19:39 meta.properties
-rw----- 1 omm wheel 103 Sep 14 21:31 recovery-point-offset-check
-rw----- 1 omm wheel 103 Sep 14 21:32 replication-offset-check
drwx----- 2 omm wheel 4096 Sep 14 21:11 test_2-0
drwx----- 2 omm wheel 4096 Sep 14 21:11 test_2-1
drwx----- 2 omm wheel 4096 Sep 14 21:11 test_2-4
drwx----- 2 omm wheel 4096 Sep 14 21:11 test_2-5
drwx----- 2 omm wheel 4096 Sep 14 21:11 test_2-6
drwx----- 2 omm wheel 4096 Sep 14 21:12 test_3-1
drwx----- 2 omm wheel 4096 Sep 14 21:12 test_3-2
drwx----- 2 omm wheel 4096 Sep 14 21:12 test_3-3
drwx----- 2 omm wheel 4096 Sep 14 21:12 test_3-5
[root@node-str-coreRuzk0001 kafka-logs]#

[] Disable this terminal from "MultiExec" mode
[root@node-str-coreCDNo data1]# cd kafka-logs/
[root@node-str-coreCDNo kafka-logs]# ll
total 60
-rw----- 1 omm wheel 4 Sep 14 21:18 cleaner-offset-check
-rw----- 1 omm wheel 4 Sep 14 21:31 log-start-offset-check
-rw----- 1 omm wheel 54 Sep 14 21:18 meta.properties
-rw----- 1 omm wheel 115 Sep 14 21:31 recovery-point-offset-check
-rw----- 1 omm wheel 115 Sep 14 21:32 replication-offset-check
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_2-0
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_2-2
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_2-3
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_2-4
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_2-6
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_3-0
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_3-1
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_3-4
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_3-5
drwx----- 2 omm wheel 4096 Sep 14 21:30 test_3-6
[root@node-str-coreCDNo kafka-logs]#

[] Disable this terminal from "MultiExec" mode
[root@node-master1SPXC bin]# ./kafka-reassign-partitions.sh --reassignment-json-file ../reassignment.json --verify
Status of partition reassignment:
Reassignment of partition test_2-3 completed successfully
Reassignment of partition test_2-4 completed successfully
Reassignment of partition test_3-5 completed successfully
Reassignment of partition test_3-3 completed successfully
Reassignment of partition test_2-2 completed successfully
Reassignment of partition test_3-2 completed successfully
Reassignment of partition test_2-6 completed successfully
Reassignment of partition test_3-4 completed successfully
Reassignment of partition test_2-0 completed successfully
Reassignment of partition test_3-1 completed successfully
Reassignment of partition test_2-1 completed successfully
Reassignment of partition test_2-5 completed successfully
Reassignment of partition test_2-1 completed successfully
Reassignment of partition test_2-5 completed successfully
Reassignment of partition test_3-6 completed successfully
Throttle was removed.
[root@node-master1SPXC bin]#
```

----End

## 14.17 Kafka Token Authentication Mechanism Tool Usage

### Scenario

Operations need to be performed on tokens when the token authentication mechanism is used.

This section applies to MRS 3.x or later.

### Prerequisites

- You have understood service requirements and prepared a system user.
- The Kafka client has been installed.

### Procedure

**Step 1** Log in as a client installation user to the node on which the Kafka client is installed.

**Step 2** Switch to the Kafka client installation directory, for example, **/opt/kafkaclient**.

```
cd /opt/kafkaclient
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 4** Run the following command to perform user authentication (skip this step in normal mode):

```
kinit Component service user
```

**Step 5** Run the following command to switch to the Kafka client installation directory:

```
cd Kafka/kafka/bin
```

**Step 6** Use **kafka-delegation-tokens.sh** to perform operations on tokens.

- Generate a token for a user.  

```
./kafka-delegation-tokens.sh --create --bootstrap-server <IP1:PORT, IP2:PORT,...> --max-life-time-period <Long: max life period in milliseconds> --command-config <config file> --renewer-principal User:<user name>
```

Example: 

```
./kafka-delegation-tokens.sh --create --bootstrap-server 192.168.1.1:21007,192.168.1.2:21007,192.168.1.3:21007 --command-config ../config/producer.properties --max-life-time-period -1 --renewer-principal User:username
```

- List information about all tokens of a specified user.  

```
./kafka-delegation-tokens.sh --describe --bootstrap-server <IP1:PORT, IP2:PORT,...> --command-config <config file> --owner-principal User:<user name>
```

Example: `./kafka-delegation-tokens.sh --describe --bootstrap-server 192.168.1.1:21007,192.168.1.2:21007,192.168.1.3:21007 --command-config ../config/producer.properties --owner-principal User:username`

- Update the token validity period.

`./kafka-delegation-tokens.sh --renew --bootstrap-server <IP1:PORT, IP2:PORT,...> --renew-time-period <Long: renew time period in milliseconds> --command-config <config file> --hmac <String: HMAC of the delegation token>`

Example: `./kafka-delegation-tokens.sh --renew --bootstrap-server 192.168.1.1:21007,192.168.1.2:21007,192.168.1.3:21007 --renew-time-period -1 --command-config ../config/producer.properties --hmac ABCDEFG`

- Destroy a token.

`./kafka-delegation-tokens.sh --expire --bootstrap-server <IP1:PORT, IP2:PORT,...> --expiry-time-period <Long: expiry time period in milliseconds> --command-config <config file> --hmac <String: HMAC of the delegation token>`

Example: `./kafka-delegation-tokens.sh --expire --bootstrap-server 192.168.1.1:21007,192.168.1.2:21007,192.168.1.3:21007 --expiry-time-period -1 --command-config ../config/producer.properties --hmac ABCDEFG`

----End

## 14.18 Introduction to Kafka Logs

This section applies to MRS 3.x or later.

### Log Description

**Log paths:** The default storage path of Kafka logs is `/var/log/Bigdata/kafka`. The default storage path of audit logs is `/var/log/Bigdata/audit/kafka`.

- Broker: `/var/log/Bigdata/kafka/broker` (run logs)

**Log archive rule:** The automatic Kafka log compression function is enabled. By default, when the size of logs exceeds 30 MB, logs are automatically compressed into a log file named in the following format: `<Original log file name>-<yyyy-mm-dd_hh-mm-ss>.[ID].log.zip`. A maximum of 20 latest compressed files are retained by default. You can configure the number of compressed files and the compression threshold.

**Table 14-12** Broker log list

| Type    | Log File Name  | Description                              |
|---------|----------------|------------------------------------------|
| Run log | server.log     | Server run log of the broker process     |
|         | controller.log | Controller run log of the broker process |

| Type | Log File Name                              | Description                                                                                                                         |
|------|--------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
|      | kafka-request.log                          | Request run log of the broker process                                                                                               |
|      | log-cleaner.log                            | Cleaner run log of the broker process                                                                                               |
|      | state-change.log                           | State-change run log of the broker process                                                                                          |
|      | kafkaServer-<SSH_USER>-<DATE>-<PID>-gc.log | GC log of the broker process                                                                                                        |
|      | postinstall.log                            | Work log after broker installation                                                                                                  |
|      | prestart.log                               | Work log before broker startup                                                                                                      |
|      | checkService.log                           | Log that records whether broker starts successfully                                                                                 |
|      | start.log                                  | Startup log of the broker process                                                                                                   |
|      | stop.log                                   | Stop log of the broker process                                                                                                      |
|      | checkavailable.log                         | Log that records the health check details of the Kafka service                                                                      |
|      | checkInstanceHealth.log                    | Log that records the health check details of broker instances                                                                       |
|      | kafka-authorizer.log                       | Broker authorization log                                                                                                            |
|      | kafka-root.log                             | Broker basic log                                                                                                                    |
|      | cleanup.log                                | Cleanup log of broker uninstallation                                                                                                |
|      | metadata-backup-recovery.log               | Broker backup and recovery log                                                                                                      |
|      | ranger-kafka-plugin-enable.log             | Log that records the Ranger plug-ins enabled by brokers                                                                             |
|      | server.out                                 | Broker JVM log                                                                                                                      |
|      | audit.log                                  | Authentication log of the Ranger authentication plug-in. This log is archived in the <b>/var/log/Bigdata/audit/kafka</b> directory. |

## Log Level

**Table 14-13** describes the log levels supported by Kafka.

Levels of run logs are ERROR, WARN, INFO, and DEBUG from the highest to the lowest priority. Run logs of equal or higher levels are recorded. The higher the specified log level, the fewer the logs recorded.

**Table 14-13** Log levels

| Level | Description                                                                              |
|-------|------------------------------------------------------------------------------------------|
| ERROR | Logs of this level record error information about system running.                        |
| WARN  | Logs of this level record exception information about the current event processing.      |
| INFO  | Logs of this level record normal running status information about the system and events. |
| DEBUG | Logs of this level record the system information and system debugging information.       |

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page. See [Modifying Cluster Service Configuration Parameters](#).
- Step 2** On the menu bar on the left, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.

----End

## Log Format

The following table describes the Kafka log format.



**Table 14-14** Log formats

| Type    | Format                                                                                                                                                     | Example                                                                                           |
|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Run log | <yyyy-MM-dd HH:mm:ss,SSS> <Log Level> <Thread that generates the log> <Message in the log> <Full name of the log event invocation class>(<Log file>:<Row>) | 2015-08-08 11:09:53,483   INFO   [main]   Loading logs.   kafka.log.LogManager (Logging.scala:68) |
|         | <yyyy-MM-dd HH:mm:ss><HostName><Component name><LogLevel><Message>                                                                                         | 2015-08-08 11:09:51 10-165-0-83 Kafka INFO Running kafka-start.sh.                                |

## 14.19 Performance Tuning

### 14.19.1 Kafka Performance Tuning

#### Scenario

You can modify Kafka server parameters to improve Kafka processing capabilities in specific service scenarios.

#### Parameter Tuning

Modify the service configuration parameters. For details, see [Modifying Cluster Service Configuration Parameters](#). For details about the tuning parameters, see [Table 14-15](#).

**Table 14-15** Tuning parameters

| Parameter                         | Default Value | Scenario                                                                                                                                                                                         |
|-----------------------------------|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| num.recovery.threads.per.data.dir | 10            | During the Kafka startup process, if a large volume of data exists, you can increase the value of this parameter to accelerate the startup.                                                      |
| background.threads                | 10            | Specifies the number of threads processed by a broker background task. If a large volume of data exists, you can increase the value of this parameter to improve broker processing capabilities. |

| Parameter            | Default Value | Scenario                                                                                                                                                                                 |
|----------------------|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| num.replica.fetchers | 1             | Specifies the number of threads used when a replica requests to the Leader for data synchronization. If the value of this parameter is increased, the replica I/O concurrency increases. |
| num.io.threads       | 8             | Specifies the number of threads used by the broker to process disk I/O. It is recommended that the number of threads be greater than or equal to the number of disks.                    |
| KAFKA_HEAP_OPTS      | -Xmx6G -Xms6G | Specifies the Kafka JVM heap memory setting. If the data volume on the broker is large, adjust the heap memory size.                                                                     |

## 14.20 Common Issues About Kafka

### 14.20.1 How Do I Solve the Problem that Kafka Topics Cannot Be Deleted?

#### Question

How do I delete a Kafka topic if it fails to be deleted?

#### Answer

- Possible cause 1: The **delete.topic.enable** configuration item is not set to **true**. The deletion can be performed only when the configuration item is set to **true**.
- Possible cause 2: The **auto.create.topics.enable** configuration parameter is set to **true**, which is used by other applications and is always running in the background.

Solution:

- For cause 1: Set **delete.topic.enable** to **true** on the configuration page.
- For cause 2: Stop the application that uses the topic in the background, or set **auto.create.topics.enable** to **false** (restart the Kafka service), and then delete the topic.

## 14.21 Kafka Feature Description

### Kafka Idempotent Feature

Feature description: The function of creating idempotent producers is introduced in Kafka 0.11.0.0. After this function is enabled, producers are automatically upgraded to idempotent producers. When producers send messages with the same field values, brokers automatically detect whether the messages are duplicate to avoid duplicate data. Note that this feature can only ensure idempotence in a single partition. That is, an idempotent producer can ensure that no duplicate messages exist in a partition of a topic. Only idempotence on a single session can be implemented. The session refers to the running of the producer process. That is, idempotence cannot be ensured after the producer process is restarted.

Method for enabling this feature:

1. Add **props.put("enable.idempotence", true)** to the secondary development code.
2. Add **enable.idempotence = true** to the client configuration file.

### Kafka Transaction Feature

Feature description: Kafka 0.11 introduces the transaction feature. The Kafka transaction feature indicates that a series of producer message production and consumer offset submission operations are in the same transaction, or are regarded as an atomic operation. Message production and offset submission succeed or fail at the same time. This feature provides transactions at the Read Committed isolation level to ensure that multiple messages are written to the target partition atomically and that the consumer can view only the transaction messages that are successfully submitted. The transaction feature of Kafka is used in the following scenarios:

1. Multiple pieces of data sent by a producer can be encapsulated in a transaction to form an atomic operation. All messages are successfully sent or fail to be sent.
2. read-process-write mode: Message consumption and production are encapsulated in a transaction to form an atomic operation. In a streaming application, a service usually needs to receive messages from the upstream system, process the messages, and then send the processed messages to the downstream system. This corresponds to message consumption and production.

Example of secondary development code:

```
// Initialize the configuration and enable the transaction feature.
Properties props = new Properties();
props.put("enable.idempotence", true);
props.put("transactional.id", "transaction1");
...

KafkaProducer producer = new KafkaProducer<String, String>(props);

// init transaction
producer.initTransactions();
try {
```

```
// Start a transaction.
producer.beginTransaction();
producer.send(record1);
producer.send(record2);
// Stop a transaction.
producer.commitTransaction();
} catch (KafkaException e) {
// Abort a transaction.
producer.abortTransaction();
}
```

## Single-Cluster Cross-AZ Feature

Feature description: Kafka 2.4.0 and later versions support the cross-AZ feature of a single cluster. After the cross-AZ feature is enabled, different copies of the same partition for a new topic are stored in different AZs. When an AZ node is abnormal, nodes in other AZs can still provide services to external systems. By default, the cross-AZ feature is disabled during cluster installation. You need to manually enable it.

## Nearby Consumption

Feature description: In versions earlier than Kafka 2.4.0, the production and consumption of the client are leader copies oriented to each partition. Follower copies are used only for data redundancy and do not provide services for external systems. As a result, the leader copy has high pressure. In addition, in cross-DC and cross-rack consumption scenarios, a large volume of data is transmitted between DCs and between racks. In Kafka 2.4.0 and later versions, the Kafka kernel can consume data from follower replicas, which greatly reduces the data transmission volume and reduces the network bandwidth pressure in cross-DC and cross-rack scenarios. The community opens the ReplicaSelector API to support this feature. By default, MRS Kafka provides two methods to use this API.

1. **RackAwareReplicaSelector**: indicates that replicas in the same rack are preferentially consumed (nearby consumption in a rack).
2. **AzAwareReplicaSelector**: indicates that copies from nodes in the same AZ are preferentially consumed (nearby consumption in an AZ).

The following uses **RackAwareReplicaSelector** as an example to describe how to consume the closest replica.

```
public class RackAwareReplicaSelector implements ReplicaSelector {

 @Override
 public Optional<ReplicaView> select(TopicPartition topicPartition,
 ClientMetadata clientMetadata,
 PartitionView partitionView) {
 if (clientMetadata.rackId() != null && !clientMetadata.rackId().isEmpty()) {
 Set<ReplicaView> sameRackReplicas = partitionView.replicas().stream()
 // Filter the replicas that are in the same rack as the client.
 .filter(replicaInfo -> clientMetadata.rackId().equals(replicaInfo.endpoint().rack()))
 .collect(Collectors.toSet());
 if (sameRackReplicas.isEmpty()) {
 // If no replicas are in the same rack as the client, the leader replica is returned.
 return Optional.of(partitionView.leader());
 } else {
 // It shows that a replica that is in the same rack as the client exists.
 if (sameRackReplicas.contains(partitionView.leader())) {
 // If the client and the leader replica are in the same rack, the leader replica returns first.
 return Optional.of(partitionView.leader());
 } else {
 // Otherwise, the latest replica synchronized with the leader is returned.
 }
 }
 }
 }
}
```

```
 return sameRackReplicas.stream().max(ReplicaView.comparator());
 }
}
} else {
 // If the rack information is not contained in the client request, the leader replica is returned first.
 return Optional.of(partitionView.leader());
}
}
```

Method for enabling this feature:

1. Server: Update the **replica.selector.class** configuration item based on different features.
  - To enable "nearby consumption in a rack", set this parameter to **org.apache.kafka.common.replica.RackAwareReplicaSelector**.
  - To enable "nearby consumption in an AZ", set this parameter to **org.apache.kafka.common.replica.AzAwareReplicaSelector**.
2. Client: Add the **client.rack** configuration item to the **consumer.properties** file in the *{Client installation directory}/Kafka/kafka/config* directory.
  - If the "nearby consumption in a rack" is enabled on the server, add the information about the rack where the client is located, for example, **client.rack = /default0/rack1**.
  - If the "nearby consumption in an AZ" is enabled on the server, add the information about the rack where the client is located, for example, **client.rack = /AZ1/rack1**.

## Ranger Unified Authentication

Feature description: In versions earlier than Kafka 2.4.0, Kafka supports only the SimpleAclAuthorizer authentication plugin provided by the community. In Kafka 2.4.0 and later versions, MRS Kafka supports both the Ranger authentication plugin and the authentication plugin provided by the community. Ranger authentication is used by default. Based on the Ranger authentication plugin, fine-grained Kafka ACL management can be performed.

### NOTE

If the Ranger authentication plugin is used on the server and **allow.everyone.if.no.acl.found** is set to **true**, all actions are allowed when a non-secure port is used for access. You are advised to disable **allow.everyone.if.no.acl.found** for security clusters that use the Ranger authentication plugin.

# 15 Using KafkaManager

---

## 15.1 Introduction to KafkaManager

KafkaManager is a tool for managing Apache Kafka and provides GUI-based metric monitoring and management of Kafka clusters.

KafkaManager supports the following functions:

- Manage multiple Kafka clusters.
- Check cluster status (topics, consumers, offsets, partitions, replicas, and nodes)
- Run preferred replica election.
- Generate partition assignments with option to select brokers to use.
- Run reassignment of partitions (based on generated assignments).
- Create a topic with optional topic configurations (Multiple Kafka cluster versions are supported).
- Delete a topic (only supported on 0.8.2+ and **delete.topic.enable = true** is set in broker configuration).
- Batch generate partition assignments for multiple topics with option to select brokers to use.
- Batch run reassignment of partitions for multiple topics.
- Add partitions to an existing topic.
- Update configurations for an existing topic.
- Optionally enable JMX polling for broker-level and topic-level metrics.
- Optionally filter out consumers that do not have `ids/owner/&offsets/` directories in ZooKeeper.

## 15.2 Accessing the KafkaManager Web UI

You can monitor and manage Kafka clusters on the graphical KafkaManager web UI.

## Prerequisites

- KafkaManager has been installed in a cluster.
- The password of user **admin** has been obtained. The password of user **admin** is specified by the user during MRS cluster creation.

## Accessing the KafkaManager Web UI

**Step 1** Go to the cluster details page and choose **Components > KafkaManager**.

 **NOTE**

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

**Step 2** In the **KafkaManager Summary** area, click any UI link in **KafkaManager WebUI** to access the KafkaManager web UI.

You can view the following information on the KafkaManager web UI.

- Kafka cluster list
- Broker node list and metric monitoring information of Kafka clusters
- Kafka cluster replica monitoring information
- Kafka cluster consumer monitoring information

 **NOTE**

You can click the KafkaManager logo in the upper left corner on any sub-page of KafkaManager to return to the homepage of the KafkaManager web UI, where a cluster list is displayed.

----End

## 15.3 Managing Kafka Clusters

Kafka cluster management includes the following operations:

- [Adding a Cluster on the KafkaManager Web UI](#)
- [Updating Cluster Parameters](#)
- [Deleting a Cluster on the KafkaManager Web UI](#)

### Adding a Cluster on the KafkaManager Web UI

After a Kafka cluster is created for the first time, a default Kafka cluster named **my-cluster** is created on the KafkaManager web UI. You can also add Kafka clusters that have been created on the MRS management console on the KafkaManager web UI to manage multiple Kafka clusters.

**Step 1** Log in to the KafkaManager web UI.

**Step 2** In the upper part of the page, choose **Cluster > Add Cluster**.

**Step 3** Set the cluster parameters. For the following parameters, refer to their example values. Retain the default values for other parameters.

**Table 15-1** Cluster parameters to be modified

| Parameter                                                                   | Example Value                             | Description                                                                                                                                                                                                                            |
|-----------------------------------------------------------------------------|-------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Cluster Name                                                                | mrs-demo                                  | Name of the cluster to be added on the KafkaManager web UI                                                                                                                                                                             |
| Cluster Zookeeper Hosts                                                     | zk1_ip:zk1_port,<br>zk2_ip:zk2_port/kafka | ZooKeeper address of the cluster to be added                                                                                                                                                                                           |
| Kafka Version                                                               | 1.1.0                                     | Kafka version of the cluster to be added. The default value is <b>1.1.0</b> .                                                                                                                                                          |
| Enable JMX Polling (Set JMX_PORT env variable before starting kafka server) | Selected                                  | -                                                                                                                                                                                                                                      |
| Poll consumer information (Not recommended for large # of consumers)        | Selected                                  | -                                                                                                                                                                                                                                      |
| Enable Active OffsetCache (Not recommended for large # of consumers)        | Selected                                  | -                                                                                                                                                                                                                                      |
| Display Broker and Topic Size (only works after applying this patch)        | Selected                                  | -                                                                                                                                                                                                                                      |
| Security Protocol                                                           | PLAINTEXT                                 | <ul style="list-style-type: none"> <li>For a Kafka cluster with Kerberos authentication enabled, select <b>SASL_PLAINTEXT</b>.</li> <li>For a Kafka cluster with Kerberos authentication disabled, select <b>PLAINTEXT</b>.</li> </ul> |

**Step 4** Click **Save**.

----End

## Updating Cluster Parameters

**Step 1** Log in to the KafkaManager web UI.

**Step 2** Click **Modify** in the **Operations** column of the cluster.



**Step 3** Go to the cluster configuration page and modify cluster parameters.

----End

## Deleting a Cluster on the KafkaManager Web UI

**Step 1** Log in to the KafkaManager web UI.

**Step 2** Click **Disable** in the **Operations** column of the cluster.

**Step 3** When **Delete** or **Enable** is displayed in the **Operations** column on the cluster list page, click **Delete** to delete the cluster. You can also click **Enable** to enable the cluster.

----End

## 15.4 Kafka Cluster Monitoring Management

The Kafka cluster monitoring management includes the following operations:

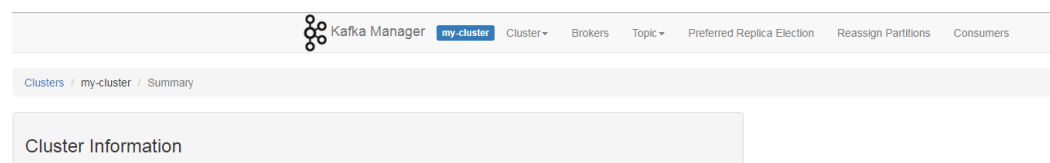
- [Viewing Broker Information](#)
- [Viewing Topic Information](#)
- [Viewing Consumers Information](#)
- [Modifying the Partition of a Topic Through KafkaManager](#)

### Viewing Broker Information

**Step 1** Log in to the KafkaManager web UI.

**Step 2** On the cluster list page, click a cluster name to access the Summary page of the cluster.

**Figure 15-1** Summary page of a cluster



**Step 3** Click **Brokers** to access the Broker monitoring page. The page displays the Broker list and I/O statistics of the Broker nodes.

**Figure 15-2** Broker monitoring page

| Brokers |                         |       |          |          |           |      | Combined Metrics            |      |       |       |        |
|---------|-------------------------|-------|----------|----------|-----------|------|-----------------------------|------|-------|-------|--------|
|         |                         |       |          |          |           |      | Rate                        | Mean | 1 min | 5 min | 15 min |
| Id      | Host                    | Port  | JMX Port | Bytes In | Bytes Out | Size |                             |      |       |       |        |
| 1       | SSL-9093-PLAINTEXT-9092 | 21006 | 0.00     | 0.00     | 0 B       |      | Messages in /sec            | 0.00 | 0.00  | 0.00  | 0.00   |
| 2       | SSL-9093-PLAINTEXT-9092 | 21006 | 0.00     | 0.00     | 0 B       |      | Bytes in /sec               | 0.05 | 0.00  | 0.00  | 0.00   |
| 3       | SSL-9093-PLAINTEXT-9092 | 21006 | 0.00     | 0.00     | 0 B       |      | Bytes out /sec              | 0.02 | 0.00  | 0.00  | 0.00   |
|         |                         |       |          |          |           |      | Bytes rejected /sec         | 0.00 | 0.00  | 0.00  | 0.00   |
|         |                         |       |          |          |           |      | Failed fetch request /sec   | 0.00 | 0.00  | 0.00  | 0.00   |
|         |                         |       |          |          |           |      | Failed produce request /sec | 0.00 | 0.00  | 0.00  | 0.00   |

----End

### Viewing Topic Information

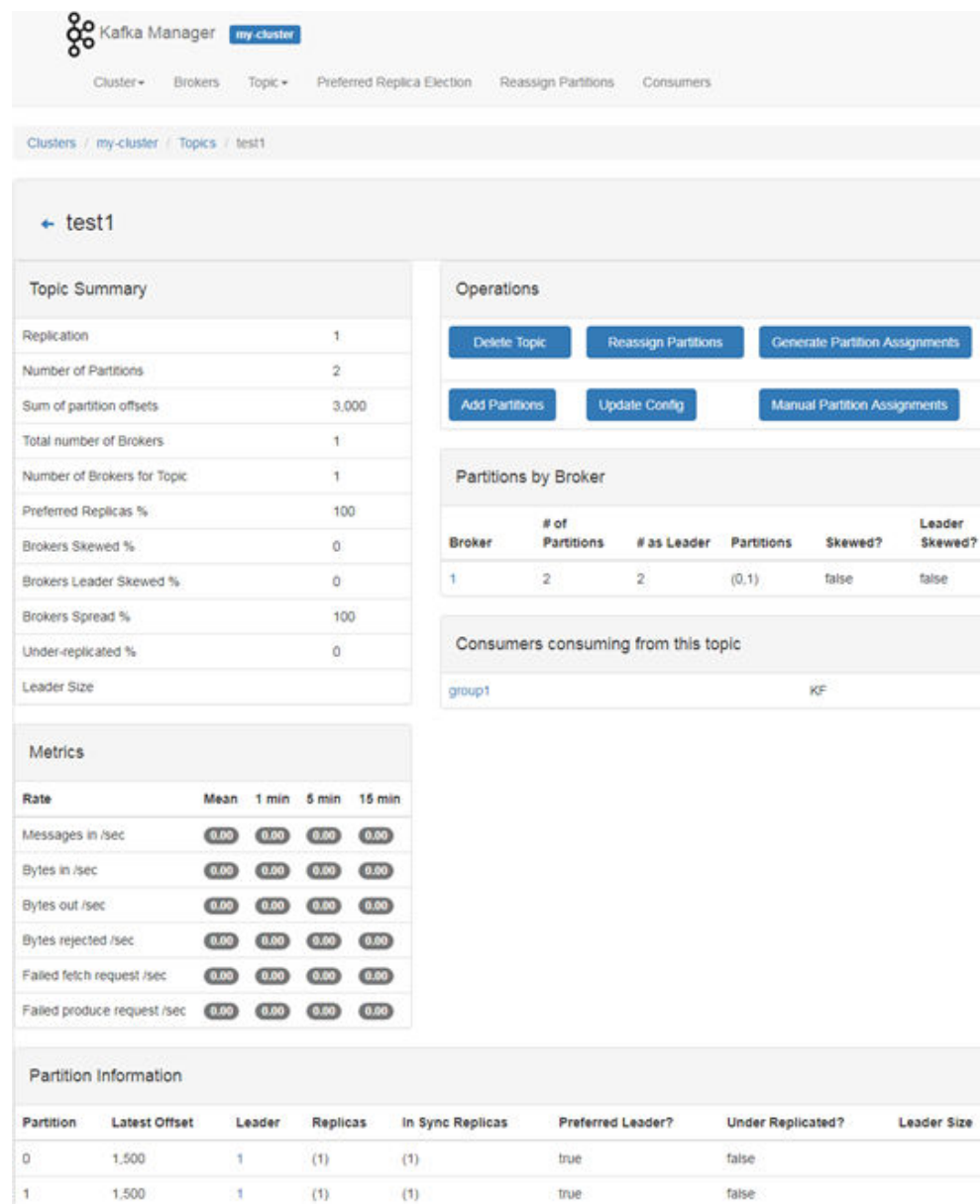
- Step 1** Log in to the KafkaManager web UI.
- Step 2** On the cluster list page, click a cluster name to access the **Summary** page of the cluster.
- Step 3** Choose **Topic > List** to view the topic list of the current cluster and information about each topic.

**Figure 15-3** Topic list

| Topic            | # Partitions | # Brokers | Brokers Spread % | Brokers Leader Skew % | Brokers Replicas | Under Replicated % | Leader Size | Producer Message/Sec |
|------------------|--------------|-----------|------------------|-----------------------|------------------|--------------------|-------------|----------------------|
| consumer_offsets | 50           | 1         | 100              | 0                     | 1                | 0                  |             | 0.00                 |
| test1            | 2            | 1         | 100              | 0                     | 1                | 0                  |             | 0.00                 |

- Step 4** Click a topic name to view details about the topic.

**Figure 15-4** Topic details

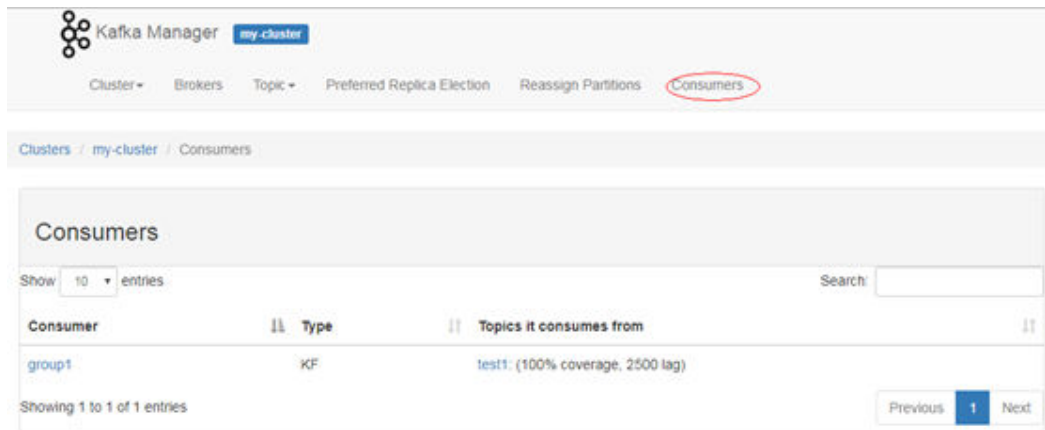


----End

## Viewing Consumers Information

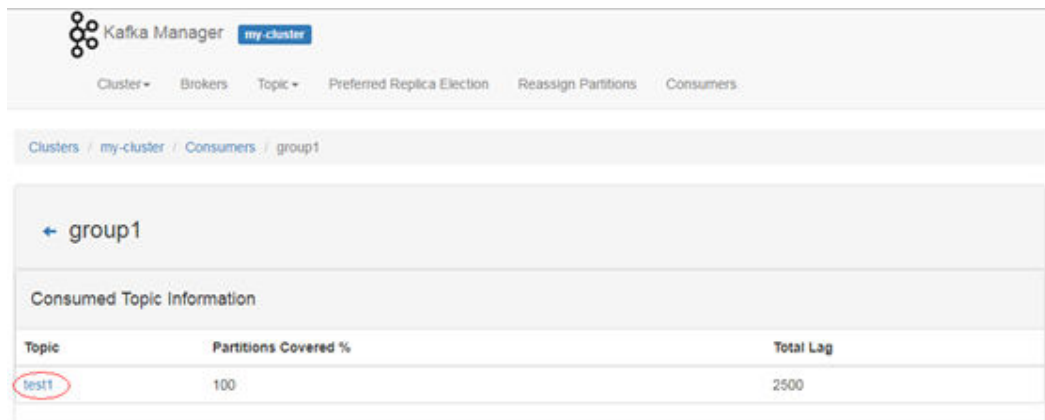
- Step 1** Log in to the KafkaManager web UI.
- Step 2** On the cluster list page, click a cluster name to access the **Summary** page of the cluster.
- Step 3** Click **Consumers** to view the consumers of the current cluster and each consumer's consumption information.

**Figure 15-5** Consumers



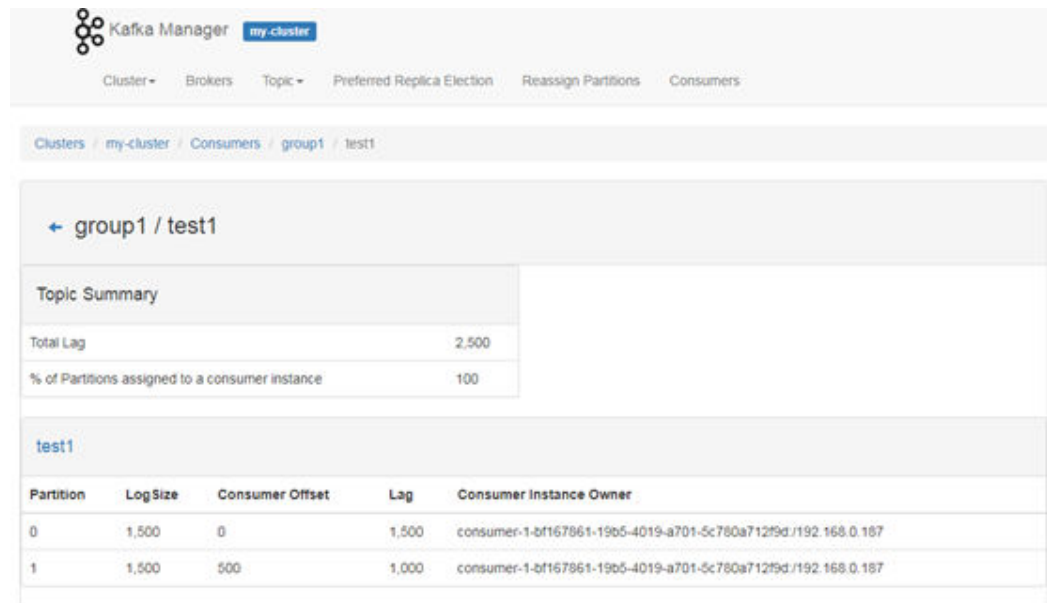
**Step 4** Click a consumer name to view the list of the consumed topics.

**Figure 15-6** List of topics consumed by the consumer



**Step 5** Click a topic name in the topic list of the consumer to view consumption information about the topic.

**Figure 15-7** Topic consumption details

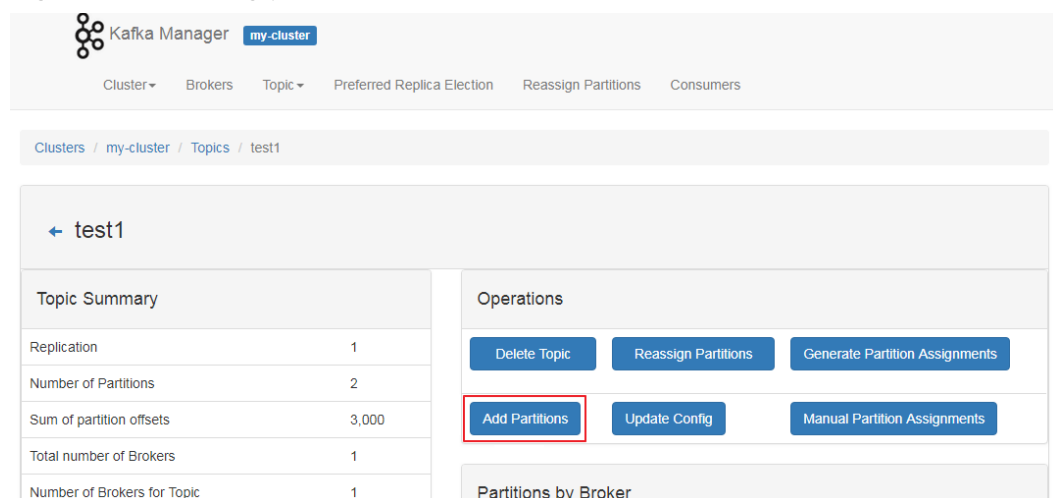


----End

## Modifying the Partition of a Topic Through KafkaManager

- Step 1** Log in to the KafkaManager web UI.
- Step 2** On the cluster list page, click a cluster name to access the **Summary** page of the cluster.
- Step 3** Choose **Topic > List** to access the topic list page of the current cluster.
- Step 4** Click a topic name to access the **Topic Summary** page.
- Step 5** Click **Add Partitions**. The page for adding partitions is displayed.

**Figure 15-8** Adding partitions



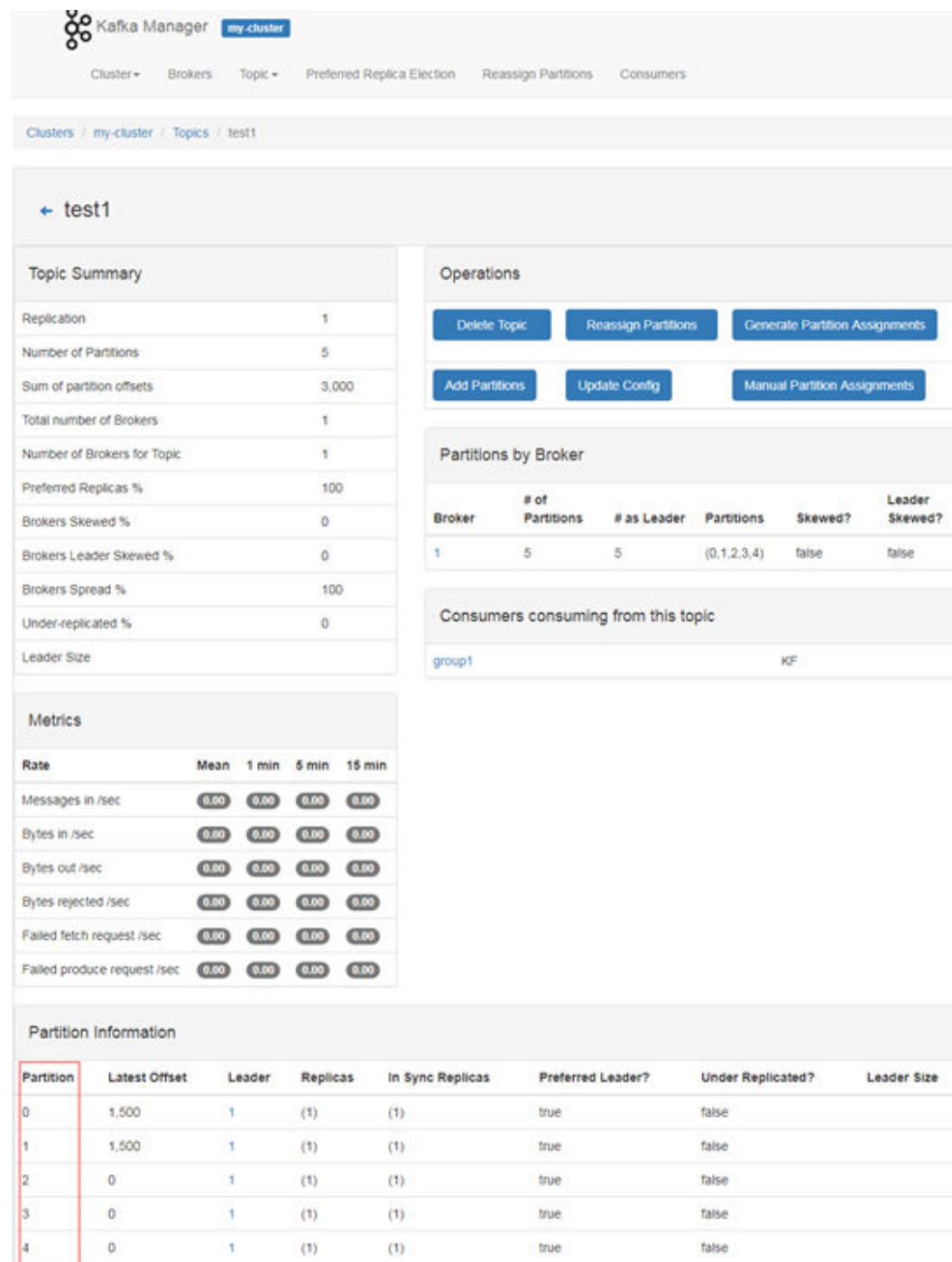
- Step 6** Confirm the topic name and modify the value of the **Partitions** parameter and click **Add Partitions** to add partitions.

**Figure 15-9** Modifying the number of partitions

The screenshot shows the 'Add Partitions' dialog in KafkaManager. The breadcrumb navigation at the top reads 'Clusters / my-cluster / Topics / test1 / Add Partitions'. The dialog title is 'Add Partitions'. It is divided into two sections: 'Add Partitions' and 'Brokers'. In the 'Add Partitions' section, the 'Topic' field contains 'test1'. In the 'Brokers' section, there are 'Select All' and 'Select None' buttons, and a list of brokers with the first one, '1 - 192.168.0.112', checked. The 'Partitions' field, which is highlighted with a red box, contains the number '2'. At the bottom of the dialog, there are 'Add Partitions' and 'Cancel' buttons.

- Step 7** After the partitions are added successfully, click **Go to topic view** to return to the **Topic Summary** page.
- Step 8** Check the number of partitions in **Partition Information** in the lower part of the **Topic Summary** page.

**Figure 15-10** Partition Information



**Step 9** (Optional) If you are not satisfied with the assigned partitions, you can use the partition reassignment function to automatically reassign partitions.

1. On the **Topic Summary** page, click **Generate Partition Assignments**.
2. Select the broker instance and click **Generate Partition Assignments** to generate a partition.
3. After partition generation, click **Go to topic view** to return to the **Topic Summary** page.

4. On the **Topic Summary** page, click **Reassign Partitions** to automatically assign partitions to the broker instance of the cluster.
5. Click **Go to reassign partitions** to view details about the reassigned partitions.

**Step 10** (Optional) If you are not satisfied with the automatically assigned partitions, you can manually assign the partitions.

1. On the **Topic Summary** page, click **Manual Partition Assignments** to access the page for manually assign partitions.
2. Manually assign a broker ID to each partition replica, and click **Save Partition Assignment** to save the changes.
3. Click **Go to topic view** to return to the **Topic Summary** page and view the partition details.

----End



# 16 Using Kudu

---

## 16.1 Using Kudu from Scratch

Kudu is a columnar storage manager developed for the Apache Hadoop platform. Kudu shares the common technical properties of Hadoop ecosystem applications. It is horizontally scalable and supports highly available operations.

### Prerequisites

The cluster client has been installed. For example, the client is installed in the `/opt/hadoopclient` directory. The client directory in the following operations is only an example. Change it to the actual installation directory.

### Procedure

**Step 1** Log in to the node where the client is installed as the client installation user.

**Step 2** Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 4** Run the Kudu command line tool.

Run the command line tool of the Kudu component to view help information.

```
kudu -h
```

The command output is as follows:

```
Usage: ./kudu <command> [<args>]

<command> can be one of the following:
 cluster Operate on a Kudu cluster
 diagnose Diagnostic tools for Kudu servers and clusters
 fs Operate on a local Kudu filesystem
 hms Operate on remote Hive Metastores
 local_replica Operate on local tablet replicas via the local filesystem
 master Operate on a Kudu Master
```

|                |                                                           |
|----------------|-----------------------------------------------------------|
| pbcc           | Operate on PBC (protobuf container) files                 |
| perf           | Measure the performance of a Kudu cluster                 |
| remote_replica | Operate on remote tablet replicas on a Kudu Tablet Server |
| table          | Operate on Kudu tables                                    |
| tablet         | Operate on remote Kudu tablets                            |
| test           | Various test actions                                      |
| tserver        | Operate on a Kudu Tablet Server                           |
| wal            | Operate on WAL (write-ahead log) files                    |

 **NOTE**

The Kudu command line tool does not support DDL and DML operations, but provides the refined query function for the **cluster**, **master**, **tserver**, **fs**, and **table** parameters.

**Common operations:**

- Check the tables in the current cluster.  
*./kudu table list KuduMaster instance IP1:7051, KuduMaster instance IP2:7051, KuduMaster instance IP3:7051*
- Query the configurations of the KuduMaster instance of the Kudu service.  
*./kudu master get\_flags KuduMaster instance IP:7051*
- Query the schema of a table.  
*./kudu table describe KuduMaster instance IP1:7051, KuduMaster instance IP2:7051, KuduMaster instance IP3:7051 table name*
- Delete a table.  
*./kudu table delete KuduMaster instance IP1:7051, KuduMaster instance IP2:7051, KuduMaster instance IP3:7051 table name*

 **NOTE**

To obtain the IP address of the KuduMaster instance, choose **Components > Kudu > Instances** on the cluster details page.

----End

## 16.2 Accessing the Kudu Web UI

You can view Kudu job information on the Kudu web UI.

### Prerequisites

Kudu has been installed in a cluster.

### Accessing KuduMaster WebUI (MRS 3.x or Later)

- Step 1** Log in to Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
- Step 2** Choose **Cluster > Services > Spark**.
- Step 3** In the **Dashboard** page of Kudu, click **KuduMaster(KuduMaster)** on the right side of **KuduMaster WebUI**. The KuduMaster web UI is displayed.

Figure 16-1 KuduMaster WebUI



----End

### Accessing KuduMaster WebUI (Versions Earlier Than MRS 3.x)

- Step 1** Access Manager. For details, see [Accessing MRS Manager \(Versions Earlier Than MRS 3.x\)](#).
- Step 2** choose **Services > Kudu**.
- Step 3** In **KuduMaster WebUI** of **Kudu Summary**, click **KuduMaster(KuduMaster)**. The KuduMaster web UI is displayed.

----End

# 17 Using Loader

---

## 17.1 Using Loader from Scratch

You can use Loader to import data from the SFTP server to HDFS.

This section applies to versions earlier than MRS 3.x.

### Prerequisites

- You have prepared service data.
- You have created an analysis cluster.

### Procedure

**Step 1** Access the Loader page.

1. Go to the cluster details page and choose **Services**.
2. Choose **Hue**. In **Hue Web UI** of **Hue Summary**, click **Hue (Active)**. The Hue web UI is displayed.
3. Choose **Data Browsers > Sqoop**.

The job management tab page is displayed by default on the Loader page.

**Step 2** On the Loader page, click **Manage links**.

**Step 3** Click **New link** and create **sftp-connector**. For details, see [File Server Link](#).

**Step 4** Click **New link**, enter the link name, select **hdfs-connector**, and create **hdfs-connector**.

**Step 5** On the Loader page, click **Manage jobs**.

**Step 6** Click **New Job**.

**Step 7** In **Connection**, set parameters.

1. In **Name**, enter a job name.
2. Select the source link created in [Step 3](#) and the target link created in [Step 4](#).

**Step 8** In **From**, configure the job of the source link.

For details, see [ftp-connector or sftp-connector](#).

**Step 9** In **To**, configure the job of the target link.

For details, see [hdfs-connector](#).

**Step 10** In **Task Config**, set job running parameters.

**Table 17-1** Loader job running properties

| Parameter                            | Description                                                                                                                                                                                                                                                                                                                                                          |
|--------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Extractors                           | Number of Map tasks                                                                                                                                                                                                                                                                                                                                                  |
| Loaders                              | Number of Reduce tasks<br>This parameter is displayed only when the destination field is HBase or Hive.                                                                                                                                                                                                                                                              |
| Max. Error Records in a Single Shard | Error record threshold. If the number of error records of a single Map task exceeds the threshold, the task automatically stops and the obtained data is not returned.<br><b>NOTE</b><br>Data is read and written in batches for <b>MYSQL</b> and <b>MPPDB</b> of <b>generic-jdbc-connector</b> by default. Errors are recorded once at most for each batch of data. |
| Dirty Data Directory                 | Directory for saving dirty data. If you leave this parameter blank, dirty data will not be saved.                                                                                                                                                                                                                                                                    |

**Step 11** Click **Save**.

----End

## 17.2 How to Use Loader

This section applies to versions earlier than MRS 3.x.

### Process

The process for migrating user data with Loader is as follows:

1. Access the Loader page of the Hue web UI.
2. Manage Loader links.
3. Create a job and select a data source link and a link for saving data.
4. Run the job to complete data migration.

### Loader Page

The Loader page is a graphical data migration management tool based on the open source Sqoop web UI and is hosted on the Hue web UI. Perform the following operations to access the Loader page:

1. Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

2. Choose **Data Browsers > Sqoop**.

The job management tab page is displayed by default on the Loader page.

## Loader Links

Loader links save data location information. Loader uses links to access data or save data to the specified location. Perform the following operations to access the Loader link management page:

1. Access the Loader page.
2. Click **Manage links**.  
The Loader link management page is displayed.  
Click **Manage jobs** to return to the job management page.
3. Click **New link** to go to the configuration page and set parameters to create a Loader link.

## Loader Jobs

Loader jobs are used to manage data migration tasks. Each job consists of a source data link and a destination data link. A job reads data from the source link and saves data to the destination link to complete a data migration task.

# 17.3 Loader Link Configuration

This section applies to versions earlier than MRS 3.x.

## Overview

Loader supports the following links. This section describes configurations of each link.

- obs-connector
- generic-jdbc-connector
- ftp-connector or sftp-connector
- hbase-connector, hdfs-connector, or hive-connector

## OBS Link

An OBS link is a data exchange channel between Loader and OBS. [Table 17-2](#) describes the configuration parameters.

**Table 17-2** obs-connector configuration

| Parameter | Description                 |
|-----------|-----------------------------|
| Name      | Name of a Loader connection |

| Parameter    | Description                                                                                                                                                                                                                             |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| OBS Server   | Enter an OBS endpoint. The common format is <b>OBS.Region.DomainName</b> .<br>Run the following command to query the endpoints of OBS:<br><b>cat /opt/Bigdata/apache-tomcat-7.0.78/webapps/web/WEB-INF/classes/cloud-obs.properties</b> |
| Port         | Specifies the port for accessing OBS data. The default value is <b>443</b> .                                                                                                                                                            |
| Access Key   | AK for a user to access OBS                                                                                                                                                                                                             |
| Security Key | SK corresponding to AK                                                                                                                                                                                                                  |

## Relational Database Link

A relational database link is a data exchange channel between Loader and a relational database. [Table 17-3](#) describes the configuration parameters.

### NOTE

Some parameters are hidden by default. They appear only after you click **Show Senior Parameter**.

**Table 17-3 generic-jdbc-connector** configuration

| Parameter     | Description                                                           |
|---------------|-----------------------------------------------------------------------|
| Name          | Name of a Loader link                                                 |
| Database Type | Data types supported by Loader links: <b>ORACLE, MYSQL, and MPPDB</b> |
| Host          | Database access address, which can be an IP address or domain name.   |
| Port          | Port for accessing the database                                       |
| Database      | Name of the database saving data                                      |
| Username      | Username for accessing the database                                   |
| Password      | Password of the user Use the actual password.                         |

**Table 17-4 Senior parameter** configuration

| Parameter  | Description                                                   |
|------------|---------------------------------------------------------------|
| Fetch Size | A maximum volume of data obtained during each database access |

| Parameter             | Description                                                                                                                                                                                           |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Connection Properties | Drive properties exclusive to the database link supported by databases of different types, for example, <b>autoReconnect</b> of MYSQL. If you want to define the drive properties, click <b>Add</b> . |
| Identifier Enclose    | Delimiter for reserving keywords in the database SQL. Delimiters defined in different databases vary.                                                                                                 |

## File Server Link

File server links include FTP and SFTP links and serve as a data exchange channel between Loader and a file server. [Table 17-5](#) describes the configuration parameters.

**Table 17-5 ftp-connector or sftp-connector configuration**

| Parameter   | Description                                                                                                                                             |
|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| Name        | Name of a Loader link                                                                                                                                   |
| Hostname/IP | Enter the file server access address, which can be a host name or IP address.                                                                           |
| Port        | Port for accessing the file server. <ul style="list-style-type: none"> <li>Use port <b>21</b> for FTP.</li> <li>Use port <b>22</b> for SFTP.</li> </ul> |
| Username    | Username for logging in to the file server                                                                                                              |
| Password    | Password of the user                                                                                                                                    |

## MRS Cluster Link

MRS cluster links include HBase, HDFS, and Hive links and serve as a data exchange channel between Loader and HBase, HDFS, or Hive.

When configuring an MRS cluster link, set the name, select a connector, for example, **hbase-connector**, **hdfs-connector**, or **hive-connector**, and save the settings.

# 17.4 Managing Loader Links (Versions Earlier Than MRS 3.x)

## Scenario

You can create, view, edit, and delete links on the Loader page.



This section applies to versions earlier than MRS 3.x.

## Prerequisites

You have accessed the Loader page. For details, see [Loader Page](#).

## Creating a Link

**Step 1** On the Loader page, click **Manage links**.

**Step 2** Click **New link** and configure link parameters.

For details about the parameters, see [Loader Link Configuration](#).

**Step 3** Click **Save**.

If link configurations, for example, IP address, port, and access user information, are incorrect, the link will fail to be verified and saved. In addition, VPC configurations may affect the network connectivity.

### NOTE

You can click **Test** to immediately check whether the link is available.

----End

## Viewing a Link

**Step 1** On the Loader page, click **Manage links**.

- If Kerberos authentication is enabled for the cluster, all links created by the current user are displayed by default and other users' links cannot be displayed.
- If Kerberos authentication is disabled for the cluster, all Loader links of the cluster are displayed.

**Step 2** In **Sqoop Links**, enter a link name to filter the link.

----End

## Editing a Link

**Step 1** On the Loader page, click **Manage links**.

**Step 2** Click the link name to go to the edit page.

**Step 3** Modify the link configuration parameters based on service requirements.

**Step 4** Click **Test**.

If the test is successful, go to [Step 5](#). If a message displays indicating that OBS server cannot be connected, repeat [Step 3](#).

**Step 5** Click **Save**.

If a Loader job has integrated into a Loader link, editing the link parameters may affect Loader running.

----End

## Deleting a Link

**Step 1** On the Loader page, click **Manage links**.

**Step 2** Locate the row that contains the target link, and click **Delete**.

**Step 3** In the dialog box, click **Yes, delete it**.

If a Loader job has integrated a Loader link, the link cannot be deleted.

----End

# 17.5 Source Link Configurations of Loader Jobs

## Overview

When Loader jobs obtain data from different data sources, a link corresponding to a data source type needs to be selected and the link properties need to be configured.

This section applies to versions earlier than MRS 3.x.

## obs-connector

**Table 17-6** Data source link properties of **obs-connector**

| Parameter                 | Description                                                                                                                                                                                                                                                                                                              |
|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Bucket Name               | OBS file system for storing source data.                                                                                                                                                                                                                                                                                 |
| Source Directory/<br>File | Actual storage form of source data. It can be either all data files in a directory or a single data file contained in the file system.                                                                                                                                                                                   |
| File Format               | Loader supports the following file formats of data stored in OBS: <ul style="list-style-type: none"> <li>• <b>CSV_FILE</b>: Specifies a text file. When the destination link is a database link, only the text file is supported.</li> <li>• <b>BINARY_FILE</b>: Specifies binary files excluding text files.</li> </ul> |
| Line Separator            | Identifier of each line end of source data                                                                                                                                                                                                                                                                               |
| Field Separator           | Identifier of each field end of source data                                                                                                                                                                                                                                                                              |
| Encoding Type             | Text encoding type of source data. It takes effect on text files only.                                                                                                                                                                                                                                                   |

| Parameter       | Description                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| File Split Type | <p>The following types are supported:</p> <ul style="list-style-type: none"> <li>• <b>File:</b> The number of files is assigned to a map task by the total number of files. The calculation formula is <b>Total number of files/Extractors</b>.</li> <li>• <b>Size:</b> A file size is assigned to a map task by the total file size. The calculation formula is <b>Total file size/Extractors</b>.</li> </ul> |

## generic-jdbc-connector

**Table 17-7** Data source link properties of **generic-jdbc-connector**

| Parameter             | Description                                                                               |
|-----------------------|-------------------------------------------------------------------------------------------|
| Schema/<br>Tablespace | Name of the database storing source data. You can query and select it on the interface.   |
| Table Name            | Data table storing the source data. You can query and select it on the interface.         |
| Partition Column      | If multiple columns need to be read, use this column to split the result and obtain data. |
| Where Clause          | Query statement used when accessing the database                                          |

## ftp-connector or sftp-connector

**Table 17-8** Data source link properties of **ftp-connector** or **sftp-connector**

| Parameter                 | Description                                                                                                                                                                                                                                                                                                                                 |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Source Directory/<br>File | Actual storage form of source data. It can be either all data files in a directory or single data file contained in the file server.                                                                                                                                                                                                        |
| File Format               | <p>Loader supports the following file formats of data stored in the file server:</p> <ul style="list-style-type: none"> <li>• <b>CSV_FILE:</b> Specifies a text file. When the destination link is a database link, only the text file is supported.</li> <li>• <b>BINARY_FILE:</b> Specifies binary files excluding text files.</li> </ul> |
| Line Separator            | <p>Identifier of each line end of source data</p> <p><b>NOTE</b><br/>If FTP or SFTP serves as a source link and <b>File Format</b> is set to <b>BINARY_FILE</b>, the value of <b>Line Separator</b> in the advanced properties is invalid.</p>                                                                                              |

| Parameter       | Description                                                                                                                                                                                                                                                                                                                                                                                             |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Field Separator | Identifier of each field end of source data<br><b>NOTE</b><br>If FTP or SFTP serves as a source link and <b>File Format</b> is set to <b>BINARY_FILE</b> , the value of <b>Field Separator</b> in the advanced properties is invalid.                                                                                                                                                                   |
| Encoding Type   | Text encoding type of source data. It takes effect on text files only.                                                                                                                                                                                                                                                                                                                                  |
| File Split Type | The following types are supported: <ul style="list-style-type: none"> <li>• <b>File</b>: The number of files is assigned to a map task by the total number of files. The calculation formula is <b>Total number of files/Extractors</b>.</li> <li>• <b>Size</b>: A file size is assigned to a map task by the total file size. The calculation formula is <b>Total file size/Extractors</b>.</li> </ul> |

## hbase-connector

**Table 17-9** Data source link properties of **hbase-connector**

| Parameter  | Description                     |
|------------|---------------------------------|
| Table Name | HBase table storing source data |

## hdfs-connector

**Table 17-10** Data source link properties of **hdfs-connector**

| Parameter                 | Description                                                                                                                                                                                                                                                                                                               |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Source Directory/<br>File | Actual storage form of source data. It can be either all data files in a directory or single data file contained in HDFS.                                                                                                                                                                                                 |
| File Format               | Loader supports the following file formats of data stored in HDFS: <ul style="list-style-type: none"> <li>• <b>CSV_FILE</b>: Specifies a text file. When the destination link is a database link, only the text file is supported.</li> <li>• <b>BINARY_FILE</b>: Specifies binary files excluding text files.</li> </ul> |
| Line Separator            | Identifier of each line end of source data<br><b>NOTE</b><br>If HDFS serves as a source link and <b>File Format</b> is set to <b>BINARY_FILE</b> , the value of <b>Line Separator</b> in the advanced properties is invalid.                                                                                              |

| Parameter       | Description                                                                                                                                                                                                                                                                                                                                                                                             |
|-----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Field Separator | Identifier of each field end of source data<br><b>NOTE</b><br>If HDFS serves as a source link and <b>File Format</b> is set to <b>BINARY_FILE</b> , the value of <b>Field Separator</b> in the advanced properties is invalid.                                                                                                                                                                          |
| File Split Type | The following types are supported: <ul style="list-style-type: none"> <li>• <b>File</b>: The number of files is assigned to a map task by the total number of files. The calculation formula is <b>Total number of files/Extractors</b>.</li> <li>• <b>Size</b>: A file size is assigned to a map task by the total file size. The calculation formula is <b>Total file size/Extractors</b>.</li> </ul> |

## hive-connector

**Table 17-11** Data source link properties of **hive-connector**

| Parameter     | Description                                                                                      |
|---------------|--------------------------------------------------------------------------------------------------|
| Database Name | Name of the Hive database storing the data source. You can query and select it on the interface. |
| Table         | Name of the Hive table storing the data source. You can query and select it on the interface.    |

# 17.6 Destination Link Configurations of Loader Jobs

## Overview

When Loader jobs save data to different storage locations, a destination link needs to be selected and the link properties need to be configured.

## obs-connector

**Table 17-12** Destination link properties of **obs-connector**

| Parameter        | Description                                                                         |
|------------------|-------------------------------------------------------------------------------------|
| Bucket Name      | OBS file system for storing final data.                                             |
| Output Directory | Directory for storing final data in the file system. A directory must be specified. |

| Parameter       | Description                                                                                                                                                                                                                                                                                                              |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| File Format     | Loader supports the following file formats of data stored in OBS: <ul style="list-style-type: none"> <li>• <b>CSV_FILE</b>: Specifies a text file. When the destination link is a database link, only the text file is supported.</li> <li>• <b>BINARY_FILE</b>: Specifies binary files excluding text files.</li> </ul> |
| Line Separator  | Identifier of each line end of final data                                                                                                                                                                                                                                                                                |
| Field Separator | Identifier of each field end of final data                                                                                                                                                                                                                                                                               |
| Encoding Type   | Text encoding type of final data. It takes effect on text files only.                                                                                                                                                                                                                                                    |

## generic-jdbc-connector

**Table 17-13** Destination link properties of **generic-jdbc-connector**

| Parameter   | Description                             |
|-------------|-----------------------------------------|
| Schema Name | Name of the database storing final data |
| Table       | Name of the table saving final data     |

## ftp-connector or sftp-connector

**Table 17-14** Destination link properties of **ftp-connector** or **sftp-connector**

| Parameter        | Description                                                                                                                                                                                                                                                                                                                          |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Output Directory | Directory for storing final data in the file server. A directory must be specified.                                                                                                                                                                                                                                                  |
| File Format      | Loader supports the following file formats of data stored in the file server: <ul style="list-style-type: none"> <li>• <b>CSV_FILE</b>: Specifies a text file. When the destination link is a database link, only the text file is supported.</li> <li>• <b>BINARY_FILE</b>: Specifies binary files excluding text files.</li> </ul> |
| Line Separator   | Identifier of each line end of final data<br><b>NOTE</b><br>If FTP or SFTP serves as a destination link and <b>File Format</b> is set to <b>BINARY_FILE</b> , the value of <b>Line Separator</b> in the advanced properties is invalid.                                                                                              |

| Parameter       | Description                                                                                                                                                                                                                               |
|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Field Separator | Identifier of each field end of final data<br><b>NOTE</b><br>If FTP or SFTP serves as a destination link and <b>File Format</b> is set to <b>BINARY_FILE</b> , the value of <b>Field Separator</b> in the advanced properties is invalid. |
| Encoding Type   | Text encoding type of final data. It takes effect on text files only.                                                                                                                                                                     |

## hbase-connector

**Table 17-15** Destination link properties of **hbase-connector**

| Parameter                | Description                                                                                                                                                                                                                                                                                                                             |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Table Name               | Name of the HBase table saving final data. You can query and select it on the interface.                                                                                                                                                                                                                                                |
| Method                   | Data can be imported to an HBase table using either <b>BULKLOAD</b> or <b>PUTLIST</b> .                                                                                                                                                                                                                                                 |
| Clear Data Before Import | Whether to clear data in the destination HBase table. Options are as follows: <ul style="list-style-type: none"> <li>• <b>True</b>: Clean up data in the table.</li> <li>• <b>False</b>: Do not clean up data in the table. If you select <b>False</b>, an error is reported during job running if data exists in the table.</li> </ul> |

## hdfs-connector

**Table 17-16** Destination link properties of **hdfs-connector**

| Parameter         | Description                                                                                                                                                                                                                                                                                                               |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Output Directory  | Directory for storing final data in HDFS. A directory must be specified.                                                                                                                                                                                                                                                  |
| File Format       | Loader supports the following file formats of data stored in HDFS: <ul style="list-style-type: none"> <li>• <b>CSV_FILE</b>: Specifies a text file. When the destination link is a database link, only the text file is supported.</li> <li>• <b>BINARY_FILE</b>: Specifies binary files excluding text files.</li> </ul> |
| Compression Codec | Compression mode used when a file is saved to HDFS. The following modes are supported: <b>NONE</b> , <b>DEFLATE</b> , <b>GZIP</b> , <b>BZIP2</b> , <b>LZ4</b> , and <b>SNAPPY</b> .                                                                                                                                       |

| Parameter       | Description                                                                                                                                                                                                                                                                                                                                        |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Overwrite       | How to process files in the output directory when files are imported to HDFS. Options are as follows: <ul style="list-style-type: none"> <li>• <b>True:</b> Clean up files in the directory and import new files by default.</li> <li>• <b>False:</b> Do not clean up files. If files exist in the output directory, job running fails.</li> </ul> |
| Line Separator  | Identifier of each line end of final data<br><b>NOTE</b><br>If HDFS serves as a destination link and <b>File Format</b> is set to <b>BINARY_FILE</b> , the value of <b>Line Separator</b> in the advanced properties is invalid.                                                                                                                   |
| Field Separator | Identifier of each field end of final data<br><b>NOTE</b><br>If HDFS serves as a destination link and <b>File Format</b> is set to <b>BINARY_FILE</b> , the value of <b>Field Separator</b> in the advanced properties is invalid.                                                                                                                 |

## hive-connector

**Table 17-17** Destination link properties of **hive-connector**

| Parameter | Description                                                                                 |
|-----------|---------------------------------------------------------------------------------------------|
| Database  | Name of the Hive database storing final data. You can query and select it on the interface. |
| Table     | Name of the Hive table saving final data. You can query and select it on the interface.     |

## 17.7 Managing Loader Jobs

### Scenario

You can create, view, edit, and delete jobs on the Loader page.

This section applies to versions earlier than MRS 3.x.

### Prerequisites

You have accessed the Loader page. For details, see [Loader Page](#).

### Creating a Job

**Step 1** On the Loader page, click **New job**.



**Step 2** In **Connection**, set parameters.

1. In **Name**, enter a job name.
2. In **From link** and **To link**, select links accordingly.

After you select a link of a type, data is obtained from the specified source and saved to the destination.

 **NOTE**

If no available link exists, click **Add a new link**.

**Step 3** In **From**, configure the job of the source link.

For details, see [Source Link Configurations of Loader Jobs](#).

**Step 4** In **To**, configure the job of the destination link.

For details, see [Destination Link Configurations of Loader Jobs](#).

**Step 5** Check whether a database link is selected in **To link**.

Database links include:

- generic-jdbc-connector
- hbase-connector
- hive-connector

If you set **To link** to a database link, you need to configure a mapping between service data and a field in the database table.

- If you set it to a database link, go to [Step 6](#).
- If you do not set it to a database link, go to [Step 7](#).

**Step 6** In **Field Mapping**, enter a field mapping. Then proceed to [Step 7](#).

**Field Mapping** specifies a mapping between each column of user data and a field in the database table.

**Table 17-18 Field Mapping properties**

| Parameter         | Description                                                                                               |
|-------------------|-----------------------------------------------------------------------------------------------------------|
| Column Num        | Field sequence of service data                                                                            |
| Sample            | First row of sample values of service data                                                                |
| Column Family     | When <b>To link</b> is <b>hbase-connector</b> , you can select a column family for storing data.          |
| Destination Field | Field for storing data                                                                                    |
| Type              | Type of the field selected by the user                                                                    |
| Row Key           | When <b>To link</b> is <b>hbase-connector</b> , you need to select <b>Destination Field</b> as a row key. |

 **NOTE**

If the value of **From** is a connector of a file type, for example, SFTP, FTP, OBS, and HDFS files, the value of **Field Mapping** is the first row of data in the file. Ensure that the first row of data is complete. Otherwise, the Loader job will not extract columns that are not mapped.

**Step 7** In **Task Config**, set job running parameters.

**Table 17-19** Loader job running properties

| Parameter                            | Description                                                                                                                                                                                                                                                                                                                                                          |
|--------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Extractors                           | Number of Map tasks                                                                                                                                                                                                                                                                                                                                                  |
| Loaders                              | Number of Reduce tasks<br>This parameter is displayed only when the destination field is HBase or Hive.                                                                                                                                                                                                                                                              |
| Max. Error Records in a Single Shard | Error record threshold. If the number of error records of a single Map task exceeds the threshold, the task automatically stops and the obtained data is not returned.<br><b>NOTE</b><br>Data is read and written in batches for <b>MYSQL</b> and <b>MPPDB</b> of <b>generic-jdbc-connector</b> by default. Errors are recorded once at most for each batch of data. |
| Dirty Data Directory                 | Specifies the directory for saving dirty data. If you leave this parameter blank, dirty data will not be saved.                                                                                                                                                                                                                                                      |

**Step 8** Click **Save**.

----End

## Viewing a Job

**Step 1** Access the Loader page. The Loader job management page is displayed by default.

- If Kerberos authentication is enabled for the cluster, all jobs created by the current user are displayed by default and other users' jobs cannot be displayed.
- If Kerberos authentication is disabled for the cluster, all Loader jobs of the cluster are displayed.

**Step 2** In **Sqoop Jobs**, enter a job name to filter the job.

**Step 3** Click **Refresh** to obtain the latest job status.

----End

## Editing a Job

**Step 1** Access the Loader page. The Loader job management page is displayed by default.

**Step 2** Click the job name to go to the edit page.

**Step 3** Modify the job configuration parameters based on service requirements.

**Step 4** Click **Save**.


 **NOTE**

Basic job operations in the navigation bar on the left are **Run**, **Copy**, **Delete**, **Disable**, **History Record**, and **Show Job JSON Definition**.

----End

## Deleting a Job

**Step 1** Access the Loader page.

**Step 2** In the row of the specified job, click .

You can also select one or more jobs and click **Delete Job** in the upper right corner of the job list.

**Step 3** In the dialog box, click **Yes, delete it**.

If the state of a Loader job is **Running**, the job fails to be deleted.

----End

# 17.8 Preparing a Driver for MySQL Database Link

## Scenario

As a component for batch data export, Loader can import and export data using a relational database.

## Prerequisites

You have prepared service data.

## Procedure

### Procedure for versions earlier than MRS cluster 3.x

**Step 1** Download the MySQL JDBC driver **mysql-connector-java-5.1.21.jar** from the MySQL official website. For details about how to select the MySQL JDBC driver, see the following table.

**Table 17-20** Version information

| JDBC Driver Version | MySQL Version                                                  |
|---------------------|----------------------------------------------------------------|
| Connector/J 5.1     | MySQL 4.1, MySQL 5.0, MySQL 5.1, and MySQL 6.0 alpha           |
| Connector/J 5.0     | MySQL 4.1, MySQL 5.0 servers, and distributed transaction (XA) |

| JDBC Driver Version | MySQL Version                                                                   |
|---------------------|---------------------------------------------------------------------------------|
| Connector/J 3.1     | MySQL 4.1, MySQL 5.0 servers, and MySQL 5.0 except distributed transaction (XA) |
| Connector/J 3.0     | MySQL 3.x and MySQL 4.1                                                         |

**Step 2** Upload **mysql-connector-java-5.1.21.jar** to the Loader installation directory on the active and standby MRS Master nodes.

- For versions earlier than MRS 3.x, upload the package to **/opt/Bigdata/MRS\_XXX/install/FusionInsight-Sqoop-1.99.7/FusionInsight-Sqoop-1.99.7/server/jdbc/**.

In the preceding path, **XXX** indicates the MRS version number. Change it based on site requirements.

**Step 3** Change the owner of the **mysql-connector-java-5.1.21.jar** package to **omm:wheel**.

**Step 4** Modify the **jdbc.properties** configuration file.

Change the key value of **MYSQL** to **mysql-connector-java-5.1.21.jar**, for example, **MYSQL=mysql-connector-java-5.1.21.jar**.

**Step 5** Restart the Loader service.

----End

**Procedure for MRS cluster 3.x and later versions:**

For MRS 3.x or later, change the permission on the driver JAR file of the relational database by referring to the following steps:

**Step 1** Log in to the active and standby management nodes of the Loader service, obtain the driver JAR package of the relational database, and save it to the following directory on the active and standby Loader nodes: **`\${BIGDATA\_HOME}/FusionInsight\_Porter\_8.0.2.1/install/FusionInsight-Sqoop-1.99.3/FusionInsight-Sqoop-1.99.3/server/webapps/loader/WEB-INF/ext-lib**

 **NOTE**

The version 8.0.2.1 is used as an example. Replace it with the actual version number.

**Step 2** Run the following commands as user **root** on the active and standby nodes of the Loader service to change the permission:

```
cd `${BIGDATA_HOME}/FusionInsight_Porter_8.0.2.1/install/FusionInsight-Sqoop-1.99.3/FusionInsight-Sqoop-1.99.3/server/webapps/loader/WEB-INF/ext-lib
```

```
chown omm:wheel JARpackage name
```

```
chmod 600 JARpackage name
```

**Step 3** Log in to FusionInsight Manager. Choose **Cluster** and click the target cluster name. In the navigation pane on the left, choose **Services > Loader**. In the upper

right corner, choose **More**, select **Restart Service**, and enter the password to restart the Loader service.

----End

## 17.9 Loader Log Overview

### Log Description

**Log path:** The default storage path of Loader log files is `/var/log/Bigdata/loader/Log category`.

- runlog: `/var/log/Bigdata/loader/runlog` (run logs)
- scriptlog: `/var/log/Bigdata/loader/scriptlog/` (script execution logs)
- catalina: `/var/log/Bigdata/loader/catalina` (Tomcat startup and stop logs)
- audit: `/var/log/Bigdata/loader/audit` (audit logs)

**Log archive rule:**

The automatic compression and archiving function are enabled for Loader run logs and audit logs. By default, when the size of a log file exceeds 10 MB, the log file is automatically compressed into a log file named in the following rule: `<Original log file name>-<yyyymmdd_hhmmss>.[ID].log.zip`. A maximum of 20 latest compressed files are reserved. The number of compressed files can be configured on the Manager portal.

**Table 17-21** Loader log list

| Log Type   | Log File Name                          | Description                                                                                                                   |
|------------|----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| Run log    | loader.log                             | Loader system log file that records most of the logs generated when the TelcoFS system is running.                            |
|            | loader-omm-***-pid***-gc.log.*.current | Loader process GC log file                                                                                                    |
|            | sqoopInstanceCheck.log                 | Loader instance health check log file                                                                                         |
| Audit log  | default.audit                          | Loader operation audit log file that records operations such as adding, deleting, modifying, and querying jobs and user login |
| Tomcat log | catalina.out                           | Tomcat run log file.                                                                                                          |
|            | catalina. <yyyy-mm-dd >.log            | Tomcat run log file                                                                                                           |

| Log Type   | Log File Name                           | Description                                                                                                                                                                                                                                                                        |
|------------|-----------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|            | host-manager. <yyyy-mm-dd >.log         | Tomcat run log file                                                                                                                                                                                                                                                                |
|            | localhost_access_log. <yyyy-mm-dd >.txt | Tomcat run log file                                                                                                                                                                                                                                                                |
|            | manager <yyyy-mm-dd >.log               | Tomcat run log file                                                                                                                                                                                                                                                                |
|            | localhost. <yyyy-mm-dd >.log            | Tomcat run log file                                                                                                                                                                                                                                                                |
| Script log | postInstall.log                         | Loader installation script log file<br>Log file generated during the execution of the Loader installation script ( <b>postInstall.sh</b> )                                                                                                                                         |
|            | preStart.log                            | Pre-startup script log file of the Loader service<br>During startup of the Loader service, a series of preparation operations are first performed (by executing <b>preStart.sh</b> ), such as generating the keytab file. This log file records information about these operations |
|            | loader_ctl.log                          | Log file generated when Loader executes the service start and stop script ( <b>sqoop.sh</b> )                                                                                                                                                                                      |

## Log Level

**Table 17-22** describes the log levels provided by Loader. The priorities of log levels are ERROR, WARN, INFO, and DEBUG in descending order. Logs whose levels are higher than or equal to the specified level are printed. The number of printed logs decreases as the specified log level increases.

**Table 17-22** Log levels

| Level | Description                                           |
|-------|-------------------------------------------------------|
| ERROR | Error information about the current event processing. |

| Level | Description                                                    |
|-------|----------------------------------------------------------------|
| WARN  | Exception information about the current event processing.      |
| INFO  | Normal running status information about the system and events. |
| DEBUG | System information and system debugging information.           |

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page of Loader by referring to [Modifying Cluster Service Configuration Parameters](#).
- Step 2** On the menu bar on the left, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.

 **NOTE**

The configurations take effect immediately without the need to restart the service.

----End

## Log Formats

The following table lists the Loader log formats.

**Table 17-23** Log formats

| Log Type | Format                                                                                                                 | Example                                                                                                                                                                                       |
|----------|------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Run log  | <yyyy-MM-dd HH:mm:ss,SSS> <Log Level> <Thread that generates the log> <Message in the log> <Location of the log event> | 2015-06-29 14:54:35,553   INFO   [localhost-startStop-1]   ConnectionRequestHandler initialized   org.apache.sqoop.handler.ConnectionRequestHandler.<init>(ConnectionRequestHandler.java:100) |

| Log Type  | Format                                                                                         | Example                                                                                                                                                                                                                                                                                        |
|-----------|------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Audit log | <yyyy-MM-dd HH:mm:ss,SSS> <Log Level> default <Message in the log> <Location of the log event> | 2015-06-29 15:35:40,969 INFO default: UserName=admin, UserIP=10.52.0.111, Time=2015-06-29 15:35:40,969, Operation=submit, Resource=submission@21, Result=Failure, Detail={ [reason:GET_SFTP_SESSION_FAILED:Failed to get sftp session - 10.162.0.35 (caused by: Auth cancel) ]; [config:null]} |

## 17.10 Example: Using Loader to Import Data from OBS to HDFS

### Scenario

If you need to import a large volume of data from the external cluster to the internal cluster, import it from OBS to HDFS.

### Prerequisites

- You have prepared service data.
- You have created an analysis cluster.

### Procedure

**Step 1** Upload service data to your OBS file system.

**Step 2** Obtain the AK/SK information and create an OBS and HDFS link.

For details, see [Loader Link Configuration](#).

**Step 3** Access the Loader page.

If Kerberos authentication is enabled in the analysis cluster, refer to instructions in [Accessing the Hue Web UI](#).

**Step 4** Click **New Job**.

**Step 5** In **Information**, set parameters.

1. In **Name**, enter a job name. For example, **obs2hdfs**.
2. In **From link**, select the OBS link you create.
3. In **To link**, select the HDFS link you create.



**Step 6** In **From**, set source link parameters.

1. In **Bucket Name**, enter a name of the OBS file system.
2. In **Input directory or file**, enter a detailed location of service data in the file system.  
If it is a single file, enter a complete path containing the file name. If it is a directory, enter the complete path of the directory.
3. In **File format**, enter the type of the service data file.

For details, see [obs-connector](#).

**Step 7** In **To**, set destination link parameters.

1. In **Output directory**, enter the directory for storing service data in HDFS.  
If Kerberos authentication is enabled in the cluster, the current user accessing Loader needs to have the permission to write data to the directory.
2. In **File format**, enter the type of the service data file.  
The type must correspond to the type in [Step 6.3](#).
3. In **Compression codec**, enter a compression algorithm. For example, if you do not compress data, select **NONE**.
4. In **Overwrite**, select **True**.
5. Click **Show Senior Parameter** and set **Line Separator**.
6. Set **Field Separator**.

For details, see [hdfs-connector](#).

**Step 8** In **Task Config**, set job running parameters.

1. In **Extractors**, enter the number of Map tasks.
2. In **Loaders**, enter the number of Reduce tasks.  
If the destination link is an HDFS link, **Loaders** is hidden.
3. In **Max error records in single split**, enter an error record threshold.
4. In **Dirty data directory**, enter a directory for saving dirty data, for example, /**user/sqoop/obs2hdfs-dd**.

**Step 9** Click **Save and execute**.

On the **Manage jobs** page, view the job running result. You can click **Refresh** to obtain the latest job status.

----End

## 17.11 Common Issues About Loader

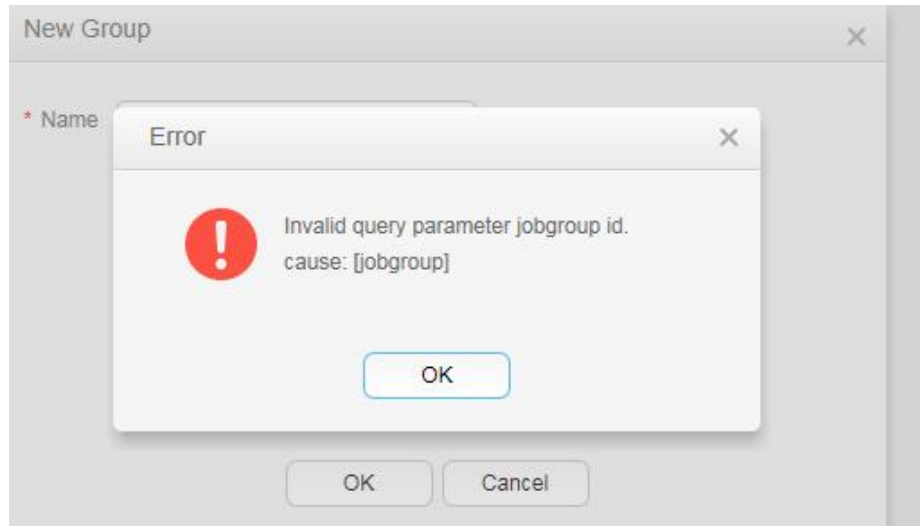
### 17.11.1 How to Resolve the Problem that Failed to Save Data When Using Internet Explorer 10 or Internet Explorer 11 ?

#### Question

Internet Explorer 11 or Internet Explorer 10 is used to access the web UI of Loader. After data is submitted, an error occurs.

## Answer

- Symptom
  - a. When the submitted data is saved, a similar error occurs: Invalid query parameter jobgroup id. cause: [jobgroup].



- Cause  
Some Internet Explorer 11 versions convert POST requests into GET requests after receiving the HTTP 307 response. As a result, POST data cannot be delivered to the server.
- Solution  
Use Google Chrome.

## 17.11.2 Differences Among Connectors Used During the Process of Importing Data from the Oracle Database to HDFS

### Question

Three types of connectors are available for importing data from the Oracle database to HDFS using Loader. That is, generic-jdbc-connector, oracle-connector, and oracle-partition-connector. Which one should I select? What are the differences between them?

### Answers

- generic-jdbc-connector  
Reads data from the Oracle database in JDBC mode. It is applicable to databases that support JDBC.  
In this mode, data loading performance of Loader is subject to data distribution in a partition column. When data skew occurs (data has only one value or several values) in a partition column, a few Maps process a significant portion of data. As a result, the index becomes invalid, causing a sharp decline in SQL query performance.  
**generic-jdbc-connector** supports view import and export, but oracle-partition-connector and oracle-connector do not support. Therefore, only this connector can be used to import views.

- Both **oracle-partition-connector** and **oracle-connector** can use the ROWID of Oracle for partitioning. **oracle-partition-connector** is self-developed and **oracle-connector** is an open-source edition. The two types of connectors share similar performance.  
**oracle-connector** requires more system table permissions. The following lists the read permissions required by the system tables of **oracle-connector** and **oracle-partition-connector**.
  - **oracle-connector**: dba\_tab\_partitions, dba\_constraints, dba\_tables t, dba\_segments, v\$instance, dba\_objects, v\$instance, SYS\_CONTEXT function, dba\_extents, and dba\_tab\_subpartitions
  - **oracle-partition-connector**: DBA\_OBJECTS and DBA\_EXTENTS

Compared with **generic-jdbc-connector**, **oracle-partition-connector** and **oracle-connector** have the following advantages:

- a. Load balancing: Number and scope of data segments are determined by the storage structure (data blocks) of the source table rather than the data on the source table. In terms of granularity, a data block can occupy a partition.
- b. Stable performance: Invalid index faults caused by data skew and bound variable snooping can be completely eliminated.
- c. Fast query speed: Using data segmentation delivers a higher query speed than that of using index.
- d. Excellent horizontal scalability: The number of generated segments increases with the increase of data volume. In this case, ideal performance can be delivered when you increase the number of concurrent tasks. Contrarily, decreasing concurrent tasks saves resources.
- e. Simplified data segmentation logic: Problems like precision loss, type compatibility, and bound variables can be prevented.
- f. Enhanced usability: Users do not need to create partition columns and tables for Loader.

# 18 Using MapReduce

---

## 18.1 Configuring the Log Archiving and Clearing Mechanism

### Scenario

Job and task logs are generated during execution of a MapReduce application.

- Job logs are generated by the MRApplicationMaster, which record details about the start and running time of jobs and each task, Counter value, and other information. After being analyzed by HistoryServer, the job logs are used to view job execution details.
- A task log records the log information generated by each task running in a container. By default, task logs are stored only on the local disk of each NodeManager. After the log aggregation function is enabled, the NodeManager merges local task logs and writes them into HDFS after job execution completes.

The job logs and task logs of the MapReduce are stored on HDFS (when the log aggregation function is enabled). If the mechanism for periodically archiving and deleting log files is not configured for a cluster with a large number of computation tasks, the log files will occupy large memory space of HDFS and increase the cluster load.

Log archive is implemented by Hadoop Archives. The number (number of Map tasks) of concurrent archiving tasks started by the Hadoop Archives is related to the total size of log files to be archived. The formula is as follows: Number of concurrent archive tasks = Total size of log files to be archived/Size of archive files.

### Configuration

Go to the **All Configurations** page of the MapReduce service. For details, see [Modifying Cluster Service Configuration Parameters](#).

Enter a parameter name in the search box. In addition, you need to configure the following information in the `mapred-site.xml` configuration file in the *Client*

*installation directory*/HDFS/hadoop/etc/hadoop/ **directory** on the MapReduce client node:

**Table 18-1** Parameter description

| Parameter                                | Description                                                                                                                                                               | Default Value              |
|------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| mapreduce.jobhistory.cleaner.enable      | Whether to enable the job log file deletion function.                                                                                                                     | true                       |
| mapreduce.jobhistory.cleaner.interval-ms | Period for starting a log file cleanup. Only log files whose retention period is longer than the time specified by <b>mapreduce.jobhistory.max-age-ms</b> can be deleted. | 86,400,000 ms (1 day)      |
| mapreduce.jobhistory.max-age-ms          | Log files whose retention period is longer than the retention period in milliseconds specified by this parameter will be deleted.                                         | 1,296,000,000 ms (15 days) |

You can configure the following parameters in the **yarn-site.xml** file on the ResourceManager, NodeManager, and MapReduce HistoryServer nodes. The **yarn.nodemanager.remote-app-log-dir** and **yarn.nodemanager.remote-app-log-archive-dir** parameters need to be configured on the Yarn client, and the configurations of the ResourceManager, NodeManager, and MapReduce HistoryServer nodes must be the same as those on the Yarn client.

**Table 18-2** Parameter description

| Parameter                                   | Description                                                                                                                                                                                                                                                                  | Default Value |
|---------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| yarn.nodemanager.remote-app-log-dir         | Indicates the HDFS path for aggregating the MapReduce job logs.                                                                                                                                                                                                              | /tmp/logs     |
| yarn.nodemanager.remote-app-log-archive-dir | Indicates the HDFS path for archiving the MapReduce job logs.                                                                                                                                                                                                                | /tmp/archived |
| yarn.log-aggregation.archive.files.minimum  | Indicates the minimum number of archived MapReduce job log files. The archiving task starts when the number of files in the <b>yarn.nodemanager.remote-app-log-dir</b> folder is greater than or equal to the value of this parameter.<br>This parameter applies to MRS 3.x. | 5,000         |

| Parameter                                           | Description                                                                                                                                                                                                                                                                                                                  | Default Value |
|-----------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| yarn.log-aggregation.archive-check-interval-seconds | Indicates the MapReduce job log archiving interval, in seconds. Log files are archived only when the number of log files reaches the value of <b>yarn.log-aggregation.archive.files.minimum</b> . The archiving function is disabled when the period is set to <b>0</b> or <b>-1</b> .<br>This parameter applies to MRS 3.x. | -1            |
| yarn.log-aggregation.retain-seconds                 | Indicates the retention period on HDFS for archiving the MapReduce job logs. The value <b>-1</b> indicates that log files are stored permanently.                                                                                                                                                                            | 1,296,000     |
| yarn.log-aggregation.retain-check-interval-seconds  | Indicates the check period (in seconds) of the MapReduce job log deletion task. If this parameter is set to <b>-1</b> , the check period is one tenth of the log retention period.                                                                                                                                           | 86400         |

## 18.2 Reducing Client Application Failure Rate

### Scenario

When the network is unstable or the cluster I/O and CPU are overloaded, client applications might encounter running failures.

### Configuration

Adjust the following parameters in the **mapred-site.xml** configuration file on the client to reduce the client application failure rate:

 **NOTE**

The **mapred-site.xml** configuration file is in the **conf** directory of the client installation path, for example, **/opt/client/Yarn/conf**.

**Table 18-3** Parameter description

| Parameter                                  | Description                                                                                                                                                                                                                           | Default Value |
|--------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.reduce.shuffle.max-host-failures | Indicates the number of allowed failures of an MR task to read remote shuffle data in the Reduce process. When the number is set to be over 5, the client application failure rate can be reduced. This parameter applies to MRS 3.x. | 5             |

| Parameter                                | Description                                                                                                                                                                                             | Default Value |
|------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.client.submit.file.replication | Indicates the backup of job files on HDFS. MR tasks are dependent on the job files during running. When the number of backups is set to be over 10, the client application failure rate can be reduced. | 10            |

## 18.3 Transmitting MapReduce Tasks from Windows to Linux

### Scenarios

When you transmit MapReduce tasks from Windows to Linux, the cluster does not support the running of the tasks.

 **NOTE**

This section applies to MRS 3.x or later.

### Configuration Description

Adjust the following parameter in the **mapred-site.xml** configuration file on the client to enable the running of MapReduce tasks: The **mapred-site.xml** configuration file is in the **conf** directory of the client installation path, for example, **/opt/client/Yarn/config**.

**Table 18-4** Parameters

| Parameter                               | Description                                                                                                                                                                                                                                                                                  | Default Value |
|-----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.app-submission.cross-platform | Indicates whether to support running of MapReduce tasks after they are transmitted from Windows to Linux. When the parameter value is <b>true</b> , the running of MapReduce tasks is supported. When the parameter value is <b>false</b> , the running of MapReduce tasks is not supported. | true          |

## 18.4 Configuring the Distributed Cache

### Scenarios

 **NOTE**

This section applies to MRS 3.x or later.

Distributed caching is useful in the following scenarios:

### Rolling Upgrade

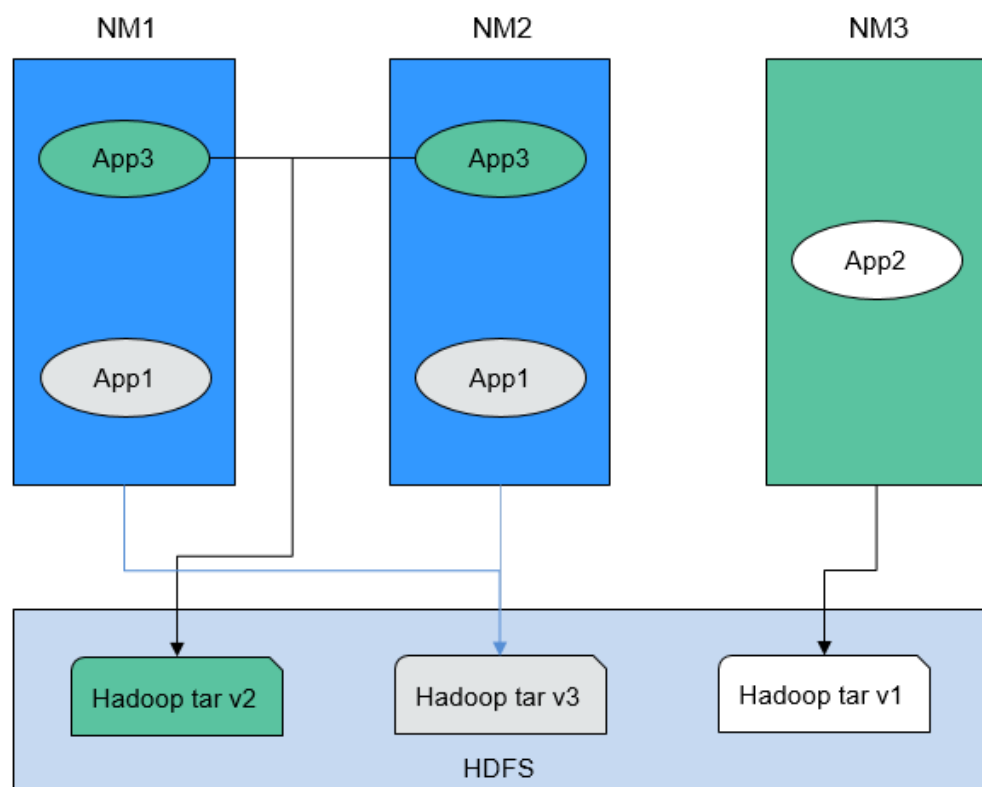
During the upgrade, applications must keep the text content (JAR file or configuration file) unchanged. The content is not based on Yarn of the current version, but on the version when it is submitted. This is a challenging issue. Generally, applications (such as MapReduce, Hive, and Tez) need to be installed locally. Libraries need to be installed on all cluster servers (clients and servers). When a rolling upgrade or downgrade starts in the cluster, the version of the locally installed library changes during application running. During the rolling upgrade, only a few NodeManagers are upgraded first. These NodeManagers obtain the software of the latest version. This leads to inconsistent behavior and can result in run-time errors.

### Co-existence of Multiple Yarn Versions

Cluster administrators may run tasks that use multiple versions of Yarn and Hadoop JARs in a cluster. However, this task is difficult to be implemented because the JARs have been localized and have only one version.

The MapReduce application framework can be deployed through the distributed cache and does not depend on the static version copied during installation. Therefore, you can store multiple versions of Hadoop in HDFS and configure the **mapred-site.xml** file to specify the default version used by the task. You can run different versions of MapReduce by setting proper configuration attributes without using the versions deployed in the cluster.

**Figure 18-1** Clusters with NodeManagers and Applications of multiple versions





As shown in [Figure 18-1](#), the application can use Hadoop JARs in HDFS instead of the local version. Therefore, during the rolling upgrade, even if NodeManager has been upgraded, the application can still run Hadoop of the earlier version.

## Configuration Description

**Step 1** Save the MapReduce **.tar** package of the specified version to a directory that can be accessed by applications in HDFS, as shown in the following command.

```
$HADOOP_HOME/bin/hdfs dfs -put hadoop-x.tar.gz /mapred/framework/
```

**Step 2** Set parameters in the **mapred-site.xml** file based on [Table 18-5](#).

**Table 18-5** Distributed cache parameters

| Parameter                            | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | Default Value |
|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.application.framework.path | <p>Indicates the URL directing to the archive location.</p> <p><b>NOTE</b><br/>This property can also create an alias for the archive if the URL fragment identity name is specified as follows. In this example, the alias is set to <b>mr-framework</b>.</p> <pre>&lt;property&gt; &lt;name&gt;mapreduce.application.framework.path&lt;/name&gt; &lt;value&gt;hdfs:/mapred/framework/hadoop-x.tar.gz#mr-framework&lt;/value&gt; &lt;/property&gt;</pre>                                                                                                                                                                                                                                                                                                                                                                     | NA            |
| mapreduce.application.classpath      | <p>Indicates the parameter property, which contains the MapReduce JARs in the class directory.</p> <p><b>NOTE</b><br/>For example, the alias <b>mr-framework</b> used in the framework path is used to match the directory.</p> <pre>&lt;property&gt; &lt;name&gt;mapreduce.application.classpath&lt;/name&gt; &lt;value&gt;\${PWD}/mr-framework/hadoop/share/hadoop/mapreduce/*: \${PWD}/mr-framework/hadoop/share/hadoop/mapreduce/lib/*: \${PWD}/mr-framework/hadoop/share/hadoop/common/*:\${PWD}/mr- framework/hadoop/share/hadoop/common/lib/*:\${PWD}/mr- framework/hadoop/share/hadoop/yarn/*:\${PWD}/mr-framework/ hadoop/share/hadoop/yarn/lib/*:\${PWD}/mr-framework/hadoop/share/ hadoop/hdfs/*:\${PWD}/mr-framework/hadoop/share/hadoop/ hdfs/lib/*:/etc/hadoop/conf/secure&lt;/value&gt;&lt;/property&gt;</pre> | N/A           |

You can upload MapReduce tarballs of multiple versions to HDFS. Different **mapred-site.xml** files indicate different locations. After that, you can run tasks for a specific **mapred-site.xml** file. The following is an example of running an MapReduce task for the MapReduce tarball of the *x* version:

```
hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar pi -conf etc/hadoop-x/mapred-site.xml 10 10
```

----End

## 18.5 Configuring the MapReduce Shuffle Address

### Scenario

When the MapReduce shuffle service is started, it attempts to bind an IP address based on local host. If the MapReduce shuffle service is required to connect to a specific IP address, no configuration is available. The following description allows you to configure a connection to a specific IP address.

### Configuration

To bind a specific IP address to the MapReduce shuffle service, set the following parameters in the **mapred-site.xml** configuration file of the node where the NodeManager instance resides:

**Table 18-6** Parameter description

| Parameter                 | Description                                                                                                                                                                                                                                                                                                                                                                                                   | Default Value |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.shuffle.address | <p>Indicates the specified address to run the shuffle service. The format is <i>IP:PORT</i>. The default value is empty. If this parameter is left empty, the local host IP address is bound. The default port number is 13562.</p> <p><b>NOTE</b><br/>If the value of <i>PORT</i> is different from that of <b>mapreduce.shuffle.port</b>, the <b>mapreduce.shuffle.port</b> value does not take effect.</p> | -             |

## 18.6 Configuring the Cluster Administrator List

### Scenario

This function is used to specify the MapReduce cluster administrator.

The administrator list is specified by **mapreduce.cluster.administrators**. The cluster administrator **admin** has all operation permissions.

### Configuration

On the **All Configurations** page of the MapReduce service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

**Table 18-7** Parameter description

| Parameter                        | Description                                                                                                                                                                                                                                                                                                    | Default Value                                                                                                        |
|----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| mapreduce.cluster.acls.enabled   | Indicates whether to enable permission control on Job History Server.                                                                                                                                                                                                                                          | true                                                                                                                 |
| mapreduce.cluster administrators | Indicates the administrator list of the MapReduce cluster. You can configure both users and user groups. Multiple users or user groups are separated by commas (,), and users and user groups are separated by spaces, for example, userA,userB groupA,groupB. The value * indicates all users or user groups. | For versions earlier than MRS 3.x: mapred<br>For MRS 3.x or later:<br>mapred<br>supergroup, System_administrator_186 |

## 18.7 Introduction to MapReduce Logs

### Log Description

#### Log paths:

- JobhistoryServer: `/var/log/Bigdata/mapreduce/jobhistory` (run log) and `/var/log/Bigdata/audit/mapreduce/jobhistory` (audit log)
- Container: `/srv/BigData/hadoop/data1/nm/containerlogs/application_${appid}/container_${$contid}`

#### NOTE

The logs of running tasks are stored in the preceding paths. After the running is complete, the system determines whether to aggregate the logs to an HDFS directory based on the Yarn configuration.

#### Log archive rule:

The automatic compression and archive function is enabled for MapReduce logs. By default, a log file is automatically compressed when the size of the log file is greater than 50 MB. The name of the compressed log file is in the following format: `<Name of the original log>-<yyyy-mm-dd_hh-mm-ss>.[NO.].log.zip`. A maximum of 100 latest compressed files are reserved. The number of compressed files can be configured on the parameter configuration page.

In MapReduce, JobhistoryServer cleans the old log files stored in HDFS periodically. The default storage directory is `/mr-history/done`. `mapreduce.jobhistory.max-age-ms` is used to set the cleanup interval. The default value of this parameter is 1,296,000,000 ms, which indicates 15 days.

**Table 18-8** MapReduce log list

| Type    | Name                                            | Description                                                                         |
|---------|-------------------------------------------------|-------------------------------------------------------------------------------------|
| Run log | jhs-daemon-start-stop.log                       | Startup log file of the daemon process                                              |
|         | hadoop-<SSH_USER>-jhshadaemon-<hostname>.log    | Run log file of the daemon process                                                  |
|         | hadoop-<SSH_USER>-<process_name>-<hostname>.out | Log that records the MapReduce running environment information                      |
|         | historyserver-<SSH_USER>-<DATE>-<PID>-gc.log    | Log that records the garbage collection of the MapReduce service                    |
|         | jhs-haCheck.log                                 | Log that records the active and standby status of MapReduce instances               |
|         | yarn-start-stop.log                             | Log that records the startup and stop of the MapReduce service                      |
|         | yarn-prestart.log                               | Log that records cluster operations before the MapReduce service startup            |
|         | yarn-postinstall.log                            | Work log before the MapReduce service startup and after the installation            |
|         | yarn-cleanup.log                                | Log that records the cleanup logs about the uninstallation of the MapReduce service |
|         | mapred-service-check.log                        | Log that records the health check details of the MapReduce service                  |
|         | container_{\$contid}                            | Container log                                                                       |
|         | hadoop-<SSH_USER>-<process_name>-<hostname>.log | MR run log                                                                          |
|         | mapred-switch-jhs.log                           | MR active/standby switchover log                                                    |

| Type      | Name                        | Description                                                           |
|-----------|-----------------------------|-----------------------------------------------------------------------|
|           | env.log                     | Environment information log before the instance is started or stopped |
| Audit log | mapred-audit-jobhistory.log | MapReduce operation audit log                                         |
|           | SecurityAuth.audit          | MapReduce security audit log                                          |

## Log Level

**Table 18-9** describes the log levels supported by MapReduce. The log levels are FATAL, ERROR, WARN, INFO, and DEBUG from high priority to low. Logs whose levels are higher than or equal to the specified level are printed. The number of printed logs decreases as the specified log level increases.

**Table 18-9** Log level

| Level | Description                                                                                |
|-------|--------------------------------------------------------------------------------------------|
| FATAL | Logs of this level record critical error information about the current event processing.   |
| ERROR | Logs of this level record error information about the current event processing.            |
| WARN  | Logs of this level record unexpected alarm information about the current event processing. |
| INFO  | Logs of this level record normal running status information about the system and events.   |
| DEBUG | Logs of this level record the system information and system debugging information.         |

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page of the MapReduce service. For details, see [Modifying Cluster Service Configuration Parameters](#).
- Step 2** On the left menu bar, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.

 NOTE

The configurations take effect immediately without restarting the service.

----End

## Log Format

The following table lists the MapReduce log formats.

**Table 18-10** Log format

| Type      | Format                                                                                                                                                                                                     | Example                                                                                                                                                                                                                                                                  |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Run log   | <i>&lt;yyyy-MM-dd<br/>HH:mm:ss,SSS&gt; &lt;Log<br/>level&gt; &lt;Name of the<br/>thread that generates the<br/>log&gt; &lt;Message in the log&gt; <br/>&lt;Location where the log<br/>event occurs&gt;</i> | 2020-01-26 14:18:59,109  <br>INFO   main   Client<br>environment:java.compiler=<N<br>A>  <br>org.apache.zookeeper.Environ<br>ment.logEnv(Environment.java<br>:100)                                                                                                       |
| Audit log | <i>&lt;yyyy-MM-dd<br/>HH:mm:ss,SSS&gt; &lt;Log<br/>level&gt; &lt;Name of the<br/>thread that generates the<br/>log&gt; &lt;Message in the log&gt; <br/>&lt;Location where the log<br/>event occurs&gt;</i> | 2020-01-26 14:24:43,605  <br>INFO   main-EventThread  <br>USER=omm<br>OPERATION=refreshAdminAcl<br>s TARGET=AdminService<br>RESULT=SUCCESS  <br>org.apache.hadoop.yarn.server.<br>resourcemanager.RMAuditLog<br>ger\$LogLevel<br>\$6.printLog(RMAuditLogger.ja<br>va:91) |

## 18.8 MapReduce Performance Tuning

### 18.8.1 Optimization Configuration for Multiple CPU Cores

#### Scenario

Optimization can be performed when the number of CPU cores is large, for example, the number of CPU cores is three times the number of disks.

#### Procedure

You can set the following parameters in either of the following ways:

- Configuration on the server:  
On the **All Configurations** page of the Yarn service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

- Configuration on the client:  
Modify the corresponding configuration file on the client.

 **NOTE**

- Path of configuration files on the HDFS client: *Client installation directory/HDFS/hadoop/etc/hadoop/hdfs-site.xml*
- Path of configuration files on the Yarn client: *Client installation directory/HDFS/hadoop/etc/hadoop/yarn-site.xml*.
- Path of configuration files on the MapReduce client: *Client installation directory/HDFS/hadoop/etc/hadoop/mapred-site.xml*.

**Table 18-11** Settings of multiple CPU cores

| Conf igation                         | Descriptio n                                                                                                                                                                                                                                                                                | Parameter                                                                                                                                                                                                                                 | Defa ult Valu e                                                | Serv er/ Clie nt | Impact                                                                                                                                                                                                                                                                                                            | Remarks                                                                                           |
|--------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------|
| Num ber of slots in a node container | The combination of the following parameters determines the number of concurrent tasks (Map and Reduce tasks) of each node: <ul style="list-style-type: none"> <li>• yarn.nodemanager.resource.memory-mb</li> <li>• mapreduce.map.memory.mb</li> <li>• mapreduce.reduce.memory.mb</li> </ul> | yarn.nodemanager.resource.memory-mb<br><b>NOTE</b><br>For versions earlier than MRS 3.x: You need to configure this parameter on the MRS console.<br>For MRS 3.x or later: You need to configure this parameter on FusionInsight Manager. | Versions earlier than MRS 3.x: 8192<br>MRS 3.x or later: 16384 | Server           | If data needs to be read from and written into disks for all tasks (Map/Reduce tasks), a disk may be accessed by multiple processes at the same time, which leads to poor disk I/O performance. To ensure disk I/O performance, the number of concurrent access requests from a client to a disk cannot exceed 3. | The maximum number of concurrent containers must be [2.5 x Number of disks configured in Hadoop]. |

| Conf igure tion | Descriptio n | Parameter                                                                                                                                                                                                                | Defa ult Valu e | Serv er/ Clie nt | Impact | Remarks |
|-----------------|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|------------------|--------|---------|
|                 |              | mapreduce.m<br>ap.memory.m<br>b<br><br><b>NOTE</b><br>You need to set this parameter in the configuration file on the client in the <i>Client installation directory/HDFS/hadoop/etc/hadoop/mapred-site.xml</i> path.    | 4096            | Clie nt          |        |         |
|                 |              | mapreduce.re<br>duce.memory.<br>mb<br><br><b>NOTE</b><br>You need to set this parameter in the configuration file on the client in the <i>Client installation directory/HDFS/hadoop/etc/hadoop/mapred-site.xml</i> path. | 4096            | Clie nt          |        |         |



| Conf igure tion               | Descriptio n                                                                                                                                                                                                                                                                                                                                                                                                                                | Parameter                                                                                                                                                                                                                   | Defa ult Valu e | Serv er/ Clie nt | Impact                                                                                               | Remarks                                                                                               |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|------------------|------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| Map outp ut and com press ion | <p>The Map task output before being written into disks can be compressed. This can save disk space, offer faster data write, and reduce the data traffic delivered to Reducer. You need to configure the following parameters on the client:</p> <ul style="list-style-type: none"> <li>• <b>mapreduce.map.output.compress</b>: The Map task output can be compressed before it is transmitted over the network. It is a per-job</li> </ul> | <p>mapreduce.map.output.compress</p> <p><b>NOTE</b><br/>You need to set this parameter in the configuration file on the client in the <i>Client installation directory/HDFS/hadoop/etc/hadoop/mapred-site.xml</i> path.</p> | true            | Client           | The disk I/O is the bottleneck. Therefore, use a compression algorithm with a high compression rate. | Snappy is used. The benchmark test results show that Snappy delivers high performance and efficiency. |

| Conf igitration | Description                                                                                                                                        | Parameter                                                                                                                                                                                                            | Default Value                          | Server/Client | Impact                                                                                                                              | Remarks |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------|---------------|-------------------------------------------------------------------------------------------------------------------------------------|---------|
|                 | configuration.<br><ul style="list-style-type: none"> <li><b>mapreduce.mapoutput.compress.codec</b>: the codec used for data compression</li> </ul> | mapreduce.mapoutput.compress.codec<br><b>NOTE</b><br>You need to set this parameter in the configuration file on the client in the <i>Client installation directory/HDFS/hadoop/etc/hadoop/mapred-site.xml</i> path. | org.apache.hadoop.io.compress.Lz4Codec | Client        |                                                                                                                                     |         |
| Spills          | mapreduce.map.sort.spill.percent                                                                                                                   | mapreduce.map.sort.spill.percent<br><b>NOTE</b><br>You need to set this parameter in the configuration file on the client in the <i>Client installation directory/HDFS/hadoop/etc/hadoop/mapred-site.xml</i> path.   | 0.8                                    | Client        | Disk I/Os are the bottleneck. You can set the value of <b>mapreduce.task.io.sort.mb</b> to minimize the memory spilled to the disk. | -       |

| Conf igure tion   | Descriptio n                                                                                                                                                                                                                                                      | Parameter                                                                                                                                                                                                            | Defa ult Valu e | Serv er/ Clie nt | Impact                                                                                                                                                                                                                                                   | Remarks                               |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------|
| Data pack et size | When the HDFS client writes data to a data node, the data will be accumulated until a packet is generated. Then, the packet is transmitted over the network. <b>dfs.client-write-packet-size</b> specifies the data packet size. It can be specified by each job. | <b>dfs.client-write-packet-size</b><br><b>NOTE</b><br>You need to set this parameter in the configuration file on the client in the <i>Client installation directory/HDFS/hadoop/etc/hadoop/hdfs-site.xml/</i> path. | 262144          | Clie nt          | The data node receives data packets from the HDFS client and writes data into disks through single threads. When disks are in the concurrent write state, increasing the data packet size can reduce the disk seek time and improve the I/O performance. | dfs.client-write-packet-size = 262144 |

## 18.8.2 Determining the Job Baseline

### Scenario

The performance optimization effect is verified by comparing actual values with the baseline data. Therefore, determining optimal job baseline is critical to performance optimization.

When determining the job baseline, comply with the following rules:

- Making full use of cluster resources
- Setting the number of Map and Reduce tasks appropriately
- Setting the runtime of each task appropriately

## Procedure

- **Rule 1: Making full use of cluster resources**

Enable all nodes to handle tasks as actively as they can when a job is executed. Maximizing the number of concurrent tasks helps make full use of resources. You can achieve this purpose by adjusting the data volume to be processed and the number of Map and Reduce tasks.

You can set **mapreduce.job.reduces** to control the number of Reduce tasks.

The number of Map tasks depends on the InputFormat type and whether the data file to be processed can be split. By default, TextFileInputFormat allocates Map tasks based on the number of blocks, that is, one Map task for each block. You can adjust the following parameters to improve resource utilization.

Parameter portal:

On the **All Configurations** page of the Yarn service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

| Parameter                                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Default Value |
|-----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.input.fileinputformat.split.maxsize | Indicates the maximum size of the data block into which the Map input information is to be split.<br><br>The shard size can be calculated based on its size customized by the user and the block size of each file. The formula is as follows:<br>splitSize = Math.max(minSize, Math.min(maxSize, blockSize))<br><br>If <b>maxSize</b> is bigger than <b>blockSize</b> , a block is a shard. If <b>maxSize</b> is smaller than <b>blockSize</b> , a block will be split into multiple shards. If the size of the remaining data in a block is smaller than <b>splitSize</b> , the remaining data will be treated as a separated shard. | -             |
| mapreduce.input.fileinputformat.split.minsize | Indicates the minimum size of a data shard.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | 0             |

- **Principle 2: Setting Reduce tasks to be executed in one round.**

Avoid the following scenarios:

- Most of Reduce tasks are completed in the first round, but there is still one Reduce task left running. The execution of the last Reduce task extends the runtime of the job. Therefore, reduce the number of Reduce tasks to enable all of them to run at the same time.

- All Map tasks are completed, but there are still Reduce tasks running on some nodes. In this case, the cluster resources are not fully utilized. You need to increase the number of Reduce tasks to enable each node to handle tasks.
- **Rule 3: Setting the runtime of each task appropriately**  
If each Map or Reduce task of a job takes only a few seconds, most time of the job is wasted on scheduling tasks and starting and stopping processes. Therefore, you need to increase the data volume to be processed in each task. The preferred processing time for each task is 1 minute.

You can configure the following parameters to adjust the processing time in a task.

Parameter portal:

On the **All Configurations** page of the Yarn service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

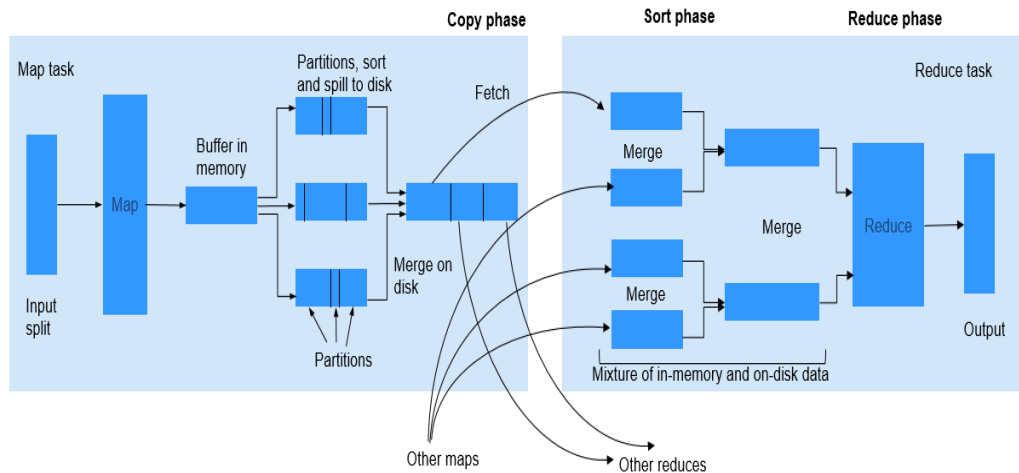
| Parameter                                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Default Value |
|-----------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.input.fileinputformat.split.maxsize | Indicates the maximum size of the data block into which the Map input information is to be split.<br><br>The shard size can be calculated based on its size customized by the user and the block size of each file. The formula is as follows:<br>$splitSize = \text{Math.max}(\text{minSize}, \text{Math.min}(\text{maxSize}, \text{blockSize}))$<br><br>If <b>maxSize</b> is bigger than <b>blockSize</b> , a block is a shard. If <b>maxSize</b> is smaller than <b>blockSize</b> , a block will be split into multiple shards. If the size of the remaining data in a block is smaller than <b>splitSize</b> , the remaining data will be treated as a separated shard. | -             |
| mapreduce.input.fileinputformat.split.minsize | Indicates the minimum size of a data shard.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 0             |

### 18.8.3 Streamlining Shuffle

#### Scenario

During the shuffle procedure of MapReduce, the Map task writes intermediate data into disks, and the Reduce task copies and adds the data to the reduce function. Hadoop provides lots of parameters for the optimization.

Figure 18-2 Shuffle process



## Procedure

### 1. Improving Performance in Map Phase

- Determine the memory used by Map.

To determine whether Map has sufficient memory, check the number of GCs and the ratio of the GC time over the total task time in counters of completed jobs. Normally, the GC time cannot exceed 10% of the task time (that is, GC time elapsed (ms)/CPU time spent (ms) < 10%).

You can improve Map performance by adjusting the following parameters.

Parameter portal:

On the **All Configurations** page of the Yarn service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

Table 18-12 Parameter description

| Parameter               | Description                                                                                                                                                                                                                                                                       | Default Value                                                                                                                                                                   |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| mapreduce.map.memory.mb | Memory restriction of a Map task.                                                                                                                                                                                                                                                 | 4096                                                                                                                                                                            |
| mapreduce.map.java.opts | JVM parameter of the Map subtask. If this parameter is set, it will replace the <b>mapred.child.java.opts</b> parameter. If <b>-Xmx</b> is not set, the value of <b>Xmx</b> is calculated based on <b>mapreduce.map.memory.mb</b> and <b>mapreduce.job.heap.memory-mb.ratio</b> . | For versions earlier than MRS 3.x: -Xmx2048M -Djava.net.preferIPv4Stack=true<br>For MRS 3.x or later: -Djava.net.preferIPv4Stack=true -Djava.security.krb5.conf=\${KRB5_CONFIG} |

It is recommended that the `-Xmx` in `mapreduce.map.java.opts` is 0.8 times the value of `mapreduce.map.memory.mb`.

– Using Combiner

Combiner is an optional procedure in the Map phase, in which the intermediate results with the same key value are combined. Generally, set the reduce class to combiner. Combiner helps reduce the intermediate result output of Map, thereby consuming less network bandwidth during the shuffle process. You can use the following API to set a combiner class for a specific job.

**Table 18-13** Combiner API

| Class                           | API                                                           | Description                                          |
|---------------------------------|---------------------------------------------------------------|------------------------------------------------------|
| org.apache.hadoop.mapreduce.Job | public void<br>setCombinerClass(Class<? extends Reducer> cls) | API used to set a combiner class for a specific job. |

2. **Improving Performance in Copy Phase**

– Compress data.

Compress the intermediate output of Map. Data compression reduces the data to be transferred over the network. However, data compression and decompression consume more CPU. Determine whether to compress the intermediate results of Map based on site requirements. If a task is bandwidth-intensive, data compression improves processing performance. As for the bulkload optimization, compression of the intermediate output improves the performance by 60%.

To improve copy performance, set `mapreduce.map.output.compress` to **true** and `mapreduce.map.output.compress.codec` to `org.apache.hadoop.io.compress.SnappyCodec`.

3. **Improving Performance in Merge Phase**

To improve merge performance, configure the following parameters to reduce the number of times that Reduce writes data to disks.

Parameter portal:

On the **All Configurations** page of the Yarn service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

**Table 18-14** Parameter description

| Parameter                                     | Description                                                                                                                                                                                                                                                                                                                                           | Default Value |
|-----------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.reduce.merge.inmem.threshold        | Threshold of the number of files for the in-memory merge process. When the accumulated number of files reaches the threshold, the process of in-memory merge and spilling to disks is initiated. If the value is less than or equal to <b>0</b> , the threshold does not take effect and the merge is triggered only based on the RAMFS memory usage. | 1000          |
| mapreduce.reduce.shuffle.merge.percent        | Usage threshold for initiating in-memory merge, indicating the percentage of memory allocated to the Map outputs (defined by <b>mapreduce.reduce.shuffle.input.buffer.percent</b> ).                                                                                                                                                                  | 0.66          |
| mapreduce.reduce.shuffle.input.buffer.percent | Percentage of memory to be allocated from the maximum heap size to storing Map outputs during the Shuffle.                                                                                                                                                                                                                                            | 0.70          |
| mapreduce.reduce.input.buffer.percent         | Percentage of memory (relative to the maximum heap size) to retain Map outputs during the Reduce. When the Shuffle is completed, all remaining Map outputs in memory must use less than this threshold before the Reduce begins.                                                                                                                      | 0.0           |

## 18.8.4 AM Optimization for Big Tasks

### Scenario

A big job containing 100,000 Map tasks fails. It is found that the failure is triggered by the slow response of ApplicationMaster (AM).

When the number of tasks increases, the number of objects managed by the AM increases, which requires much more memory for management. The default memory heap for AM is 1 GB.

### Procedure

You can improve the AM performance by setting the following parameters.



Navigation path for setting parameters:

Adjust the following parameters in the **mapred-site.xml** configuration file on the client to adjust the following parameters: The **mapred-site.xml** configuration file is in the **conf** directory of the client installation path, for example, **/opt/client/Yarn/config**.

| Parameter                          | Description                                                                                                         | Default Value                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------------------------------|---------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| yarn.app.mapreduce.am.resource.mb  | This parameter must be greater than the heap size specified by <b>yarn.app.mapreduce.am.command-opts</b> . Unit: MB | 1536                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| yarn.app.mapreduce.am.command-opts | Indicates the JVM startup parameters loaded to MapReduce ApplicationMaster.                                         | For versions earlier than MRS 3.x: -Xmx1024m -XX:CMSFullGCsBeforeCompaction=1 -XX:+UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled -XX:+UseCMSCompactAtFullCollection -verbose:gc<br><br>MRS 3.x or later: -Xmx1024m -XX:+UseConcMarkSweepGC -XX:+CMSParallelRemarkEnabled -verbose:gc -<br>Djava.security.krb5.conf=\${KRB5_CONFIG} -<br>Dhadoop.home.dir=\${BIGDATA_HOME}/<br>FusionInsight_HD_xxx/install/<br>FusionInsight-Hadoop-xxx/hadoop |

## 18.8.5 Speculative Execution

### Scenario

If a cluster has hundreds or thousands of nodes, the hardware or software fault of a node may prolong the execution time of the entire task (as most tasks are already completed, the system is still waiting for the task running on the faulty node). Speculative execution allows a task to be executed on multiple machines. You can disable speculative execution for small clusters.

### Procedure

Navigation path for setting parameters:

On the **All Configurations** page of the Yarn service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

| Parameter                    | Description                                                                                                                                | Default Value |
|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.map.speculative    | Sets whether to execute multiple instances of some map tasks concurrently. <b>true</b> indicates that speculative execution is enabled.    | false         |
| mapreduce.reduce.speculative | Sets whether to execute multiple instances of some reduce tasks concurrently. <b>true</b> indicates that speculative execution is enabled. | false         |

## 18.8.6 Using Slow Start

### Scenario

The Slow Start feature specifies the proportion of Map tasks to be completed before Reduce tasks are started. If the Reduce tasks are started too early, resources will be occupied, thereby reducing task running efficiency. However, if the Reduce tasks are started at an appropriate time, resource usage during shuffle and task running efficiency will be improved. For example, the MapReduce job includes 15 Map tasks and a cluster can start 10 Map tasks, there are 5 Map tasks remained after a round of Map tasks is completed and the cluster has available resources. In this case, you can configure the value of Slow Start to a value less than 1 (for example, 0.8), then the Reduce tasks can make use of the remaining cluster resources.

### Procedure

Parameter portal:

On the **All Configurations** page of the MapReduce service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

| Parameter                                    | Description                                                                                                                                                                            | Default Value |
|----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.job.reduce.slowstart.completedmaps | Fraction of the number of Maps in the job which should be completed before Reduces are scheduled for the job. By default, the Reduce tasks start when all the Map tasks are completed. | 1.0           |

## 18.8.7 Optimizing Performance for Committing MR Jobs

### Scenario

By default, if an MR job generates a large number of output files, it takes a long time for the job to commit the temporary outputs of a task to the final output

directory in the commit phase. In large clusters, the time-consuming commit process of jobs greatly affects the performance.

In this case, you can set the **mapreduce.fileoutputcommitter.algorithm.version** to **2** to improve the performance in the commit phase of MR jobs.

## Procedure

Navigation path for setting parameters:

On the **All Configurations** page of the Yarn service, enter a parameter name in the search box. For details, see [Modifying Cluster Service Configuration Parameters](#).

**Table 18-15** Parameter description

| Parameter                                       | Description                                                                                                                                                                                                                                                                                                                                               | Default Value |
|-------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| mapreduce.fileoutputcommitter.algorithm.version | <p>Indicates the algorithm version submitted by a job. The value is <b>1</b> or <b>2</b>.</p> <p><b>NOTE</b><br/> <b>2</b> is the recommended algorithm version. This algorithm enables tasks to directly commit the output results of each task to the final result output directory, reducing the time for the results of large jobs are committed.</p> | 2             |

## 18.9 Common Issues About MapReduce

### 18.9.1 Why Does It Take a Long Time to Run a Task Upon ResourceManager Active/Standby Switchover?

#### Question

MapReduce job takes a very long time (more than 10minutes) when the ResourceManager switch while the job is running.

#### Answer

This is because, ResorceManager HA is enabled but the ResorceManager work preserving restart is not enabled.

If ResorceManager work preserving restart is not enabled, then ResorceManager switch containers are killed which causes the ResorceManager to timeout the ApplicationMaster. For ResorceManager work preserving restart feature details, see <http://hadoop.apache.org/docs/r3.1.1/hadoop-yarn/hadoop-yarn-site/ResourceManagerRestart.html>.

The following method can be used to solve the issue:

Enable the ResourceManager work preserving restart feature by configuring the following parameter.

**yarn.resourcemanager.work-preserving-recovery.enabled=true**

## 18.9.2 Why Does a MapReduce Task Stay Unchanged for a Long Time?

### Question

MapReduce job is not progressing for long time

### Answer

This is because of less memory. When the memory is less, the time taken by the job to copy the map output increases significantly.

In order to reduce the waiting time, increase the heap memory.

The job configuration should be tuned according to number of mappers and data size processed by each mapper. Based on the input data size, tune the following configurations accordingly for feasible performance.

- **mapreduce.reduce.memory.mb**
- **mapreduce.reduce.java.opts**

**Example:** If the data size is 5 GB with 10 mappers, then the ideal heap memory would be 1.5 GB. Increase the heap memory size according with the increase in data size.

## 18.9.3 Why the Client Hangs During Job Running?

### Question

Why is the client unavailable when the MR ApplicationMaster or ResourceManager is moved to the D state during job running?

### Answer

When a task is running, the MR ApplicationMaster or ResourceManager is moved to D state (uninterrupted sleep state) or T state (stopped state). The client waits to return the task running state, but the MR ApplicationMaster does not return. Therefore, the client remains in the waiting state.

To avoid the preceding scenario, use the **ipc.client.rpc.timeout** configuration item in the **core-site.xml** file to set the client timeout interval.

The value of this parameter is millisecond. The default value is **0**, indicating that no timeout occurs. The client timeout interval ranges from 0 ms to 2,147,483,647 ms.

 NOTE

- If the Hadoop process is in the D state, restart the node where the process is located.
- The **core-site.xml** configuration file is stored in the **conf** directory of the client installation path, for example, **/opt/hadoopClient/Yarn/config**.

## 18.9.4 Why Cannot HDFS\_DELEGATION\_TOKEN Be Found in the Cache?

### Question

In security mode, why delegation token HDFS\_DELEGATION\_TOKEN is not found in the cache?

### Answer

In MapReduce, by default HDFS\_DELEGATION\_TOKEN will be canceled after the job completion. So if the token has to be re-used for the next job then the token will not be found in the cache.

To re-use the same token in subsequent job set the below parameter for the MR job configuration. When it is false the user can re-sue the same token.

```
jobConf.setBoolean("mapreduce.job.complete.cancel.delegation.tokens", false);
```

## 18.9.5 How Do I Set the Task Priority When Submitting a MapReduce Task?

### Question

How do I set the job priority when submitting a MapReduce task?

### Answer

You can add the parameter **-Dmapreduce.job.priority=<priority>** in the command to set task priority when submitting MapReduce tasks on the client. The format is as follows:

```
yarn jar <jar> [mainClass] -Dmapreduce.job.priority=<priority> [path1] [path2]
```

The parameters in the command are described as follows:

- **<jar>**: specifies the name of the JAR package to be run.
- **[mainClass]**: specifies the **main** method of the class for an application project in a JAR file.
- **<priority>**: specifies the priority of a task. The value can be **VERY\_HIGH**, **HIGH**, **NORMAL**, **LOW**, or **VERY\_LOW**.
- **[path1]**: specifies the data input path.
- **[path2]**: specifies the data output path.

For example, set the **/opt/client/HDFS/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-xxx.jar** package to a high-priority task.

```
yarn jar /opt/client/HDFS/hadoop/share/hadoop/mapreduce/hadoop-
mapreduce-examples-xxx.jar wordcount -
Dmapreduce.job.priority=VERY_HIGH /DATA.txt /out/
```

## 18.9.6 Why Physical Memory Overflow Occurs If a MapReduce Task Fails?

### Question

The HBase bulkload task has 210,000 Map tasks and 10,000 Reduce tasks. The MapReduce task fails to be executed, and the physical memory of ApplicationMaster overflows.

```
For more detailed output, check the application tracking page:https://bigdata-55:8090/cluster/app/
application_1449841777199_0003
Then click on links to logs of each attempt.
Diagnostics: Container [pid=21557,containerID=container_1449841777199_0003_02_000001] is running
beyond physical memory limits
Current usage: 1.0 GB of 1 GB physical memory used; 3.6 GB of 5 GB virtual memory used. Killing container.
Dump of the process-tree for container_1449841777199_0003_02_000001 :
|- PID PPID PGRPID SESSID CMD_NAME USER_MODE_TIME(MILLIS) SYSTEM_TIME(MILLIS)
VMEM_USAGE(BYTES) RSSMEM_USAGE(PAGES) FULL_CMD_LINE
|- 21584 21557 21557 21557 (java) 12342 1627 3871748096 271331 ${BIGDATA_HOME}/jdk1.8.0_51//bin/
java
-Djava.io.tmpdir=/srv/BigData/hadoop/data1/nm/localdir/usercache/hbase/appcache/
application_1449841777199_0003/container_1449841777199_0003_02_000001/tmp -
Dlog4j.configuration=container-log4j.properties
-Dyarn.app.container.log.dir=/srv/BigData/hadoop/data1/nm/containerlogs/
application_1449841777199_0003/container_1449841777199_0003_02_000001 -
Dyarn.app.container.log.filesize=0 -Dhadoop.root.logger=INFO,CLA
-Dhadoop.root.logfile=syslog -Xmx784m org.apache.hadoop.mapreduce.v2.app.MRAppMaster
|- 21557 21547 21557 21557 (bash) 0 0 13074432 368 /bin/bash -c ${BIGDATA_HOME}/jdk1.8.0_51//bin/
java
-Djava.io.tmpdir=/srv/BigData/hadoop/data1/nm/localdir/usercache/hbase/appcache/
application_1449841777199_0003/container_1449841777199_0003_02_000001/tmp -
Dlog4j.configuration=container-log4j.properties
-Dyarn.app.container.log.dir=/srv/BigData/hadoop/data1/nm/containerlogs/
application_1449841777199_0003/container_1449841777199_0003_02_000001 -
Dyarn.app.container.log.filesize=0 -Dhadoop.root.logger=INFO,CLA
-Dhadoop.root.logfile=syslog -Xmx784m org.apache.hadoop.mapreduce.v2.app.MRAppMaster 1>/srv/
BigData/hadoop/data1/nm/containerlogs/application_1449841777199_0003/
container_1449841777199_0003_02_000001/stdout
2>/srv/BigData/hadoop/data1/nm/containerlogs/application_1449841777199_0003/
container_1449841777199_0003_02_000001/stderr
Container killed on request. Exit code is 143
Container exited with a non-zero exit code 143
Failing this attempt. Failing the application.
```

### Answer

This is a performance specification problem. The root cause of the MapReduce task execution failure is the memory overflow of ApplicationMaster, that is, the NodeManager kills the task due to the physical memory overflow.

#### Solutions:

Increase the memory of ApplicationMaster and optimize the following parameters in the **mapred-site.xml** configuration file on the client:

- **yarn.app.mapreduce.am.resource.mb**
- **yarn.app.mapreduce.am.command-opts**. The recommended value of **-Xmx** is **0.8 x yarn.app.mapreduce.am.resource.mb**.

### Specification:

ApplicationMaster supports 24,000 concurrent containers when the configuration is as follows:

- `yarn.app.mapreduce.am.resource.mb=2048`
- In `yarn.app.mapreduce.am.command-opts`, `-Xmx` is `1638m`.

## 18.9.7 After the Address of MapReduce JobHistoryServer Is Changed, Why the Wrong Page is Displayed When I Click the Tracking URL on the ResourceManager WebUI?

### Question

After the address of MapReduce JobHistoryServer is changed, why the wrong page is displayed when I click the tracking URL on the ResourceManager WebUI?

### Answer

JobHistoryServer address (`mapreduce.jobhistory.address / mapreduce.jobhistory.webapp.<https.>address`) is the parameter of MapReduce. The MapReduce client will submit the address together with jobs to ResourceManager. After ResourceManager completing the jobs, the parameter is saved in `RMStateStore` as the target address for viewing history job information.

If the JobHistoryServer address is changed, update the address in the configuration file of the MapReduce client in time. If the address is not updated, the page of earlier JobHistoryServer is displayed when you click the tracking URL of the new job. The target address of information about MapReduce jobs running before the change of address cannot be changed, so the wrong page is also displayed when you click the tracking URL. You can check the history information by accessing the new JobHistoryServer address.

## 18.9.8 MapReduce Job Failed in Multiple NameService Environment

### Question

MapReduce or Yarn job fails in multiple nameService environment using viewFS.

### Answer

When using viewFS only the mount directories are accessible, so the most possible cause is that the path configured is not in one of the mounted paths. For example:

```
<property>
<name>fs.defaultFS</name>
<value>viewfs://ClusterX/</value>
</property>
<property>
<name>fs.viewfs.mounttable.ClusterX.link./folder1</name>
<value>hdfs://NS1/folder1</value>
</property>
</property>
```

```
<name>fs.viewfs.mounttable.ClusterX.link./folder2</name>
<value>hdfs://NS2/folder2</value>
</property>
```

For all the MR properties which depends on HDFS, should use the paths inside mount folders.

**Incorrect:**

```
<property>
<name>yarn.app.mapreduce.am.staging-dir</name>
<value>/tmp/hadoop-yarn/staging</value>
</property>
```

As the root folder (/) is not accessible in viewFS.

**Correct:**

```
<property>
<name>yarn.app.mapreduce.am.staging-dir</name>
<value>/folder1/tmp/hadoop-yarn/staging</value>
</property>
```

## 18.9.9 Why a Fault MapReduce Node Is Not Blacklisted?

### Question

MapReduce task fails and the ratio of fault nodes to all nodes is smaller than the blacklist threshold configured by **yarn.resourcemanager.am-scheduling.node-blacklisting-disable-threshold**. Why the fault node not be blacklisted?

### Answer

If the blacklisted percentage exceeds the threshold, all blacklisted nodes are released. Traditionally, the blacklist percentage is the ratio of fault nodes to all nodes in the cluster. Currently, each node has a label expression. Therefore, the blacklist percentage needs to be calculated based on the number of nodes related to valid node label expressions. In other way, the blacklist percentage is the ratio of fault nodes related to valid node label expressions.

Assume that there are 100 nodes in the cluster, including 10 nodes (labelA) related to valid node label expressions. Assume that all nodes related to valid node label expressions are faulty and default blacklist threshold is 0.33. In traditional calculation method,  $10/100 = 0.1$ , which is far smaller than the threshold (0.33). In this case, the 10 nodes will never get released. Therefore, MapReduce always cannot obtain nodes and applications cannot run properly. In practice, the blacklist percentage needs to be calculated based on the total number of nodes related to valid node label expressions:  $10/10 = 1$  is greater than the blacklist threshold and all nodes are released.

Therefore, even the ratio of fault nodes to all nodes in the cluster is below the threshold, all nodes in the blacklist are released.



# 19 Using Oozie

---

## 19.1 Using Oozie from Scratch

Oozie is an open-source workflow engine that is used to schedule and coordinate Hadoop jobs.

Oozie can be used to submit a wide array of jobs, such as Hive, Spark2x, Loader, MapReduce, Java, DistCp, Shell, HDFS, SSH, SubWorkflow, Streaming, and scheduled jobs.

This section describes how to use the Oozie client to submit a MapReduce job.

### Prerequisites

The client has been installed. For example, the installation directory is **/opt/client**. The client directory in the following operations is only an example. Change it based on the actual installation directory onsite.

### Procedure

- Step 1** Log in to the node where the client is installed as the client installation user.
- Step 2** Run the following command to go to the client installation directory. Assume that the client is installed in **/opt/client**.  

```
cd /opt/client
```
- Step 3** Run the following command to configure environment variables:  

```
source bigdata_env
```
- Step 4** Check the cluster authentication mode.
  - If the cluster is in security mode, run the following command to authenticate the user: *UserOozie* indicates the user who submits tasks.  

```
kinit UserOozie
```
  - If the cluster is in normal mode, go to [Step 5](#).
- Step 5** Upload the Oozie configuration file and JAR package to HDFS.

```
hdfs dfs -mkdir /user/UserOozie
```

```
hdfs dfs -put -f /opt/client/Oozie/oozie-client-*/examples /user/UserOozie/
```

*UserOozie* indicates the user who submits tasks.

**Step 6** Run the following commands to modify the job execution configuration file:

```
cd /opt/client/Oozie/oozie-client-*/examples/apps/map-reduce/
```

```
vi job.properties
```

```
nameNode=hdfs://hacluster
resourceManager=10.64.35.161:8032 (10.64.35.161 is the service plane IP address of the Yarn
resourceManager (active) node, and 8032 is the port number of yarn.resourcemanager.port)
queueName=default
examplesRoot=examples
user.name=admin
oozie.wf.application.path=${nameNode}/user/${user.name}/${examplesRoot}/apps/map-reduce#
HDFS upload path
outputDir=map-reduce
oozie.wf.rerun.failnodes=true
```

**Step 7** Run the following command to execute the Oozie job:

```
oozie job -oozie https://Host name of the Oozie role:21003/oozie/ -config
job.properties -run
```

```
[root@kwephispra44947 map-reduce]# oozie job -oozie https://kwephispra44948:21003/oozie/ -config
job.properties -run
INFO CMD=job -oozie https://kwephispra44948:21003/oozie/ -config job.properties -run
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/redis_client/Oozie/oozie-client-xxx-rc1-SNAPSHOT/lib/slf4j-
log4j12-1.7.26.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/redis_client/Oozie/oozie-client-xxx-rc1-SNAPSHOT/lib/slf4j-
simple-1.7.26.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
job: 0000000-200730163829770-oozie-omm-W
```

**Step 8** Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).

**Step 9** Choose **Cluster** > *Name of the desired cluster* > **Services** > **Oozie**, click the hyperlink next to **Oozie WebUI** to go to the Oozie page, and view the task execution result on the Oozie web UI.

**Figure 19-1** Task execution result

Job Id	Name	User	Group	Created	Started	Last Modified	Ended
1	-oozie-... map-reduce-wf			Thu, 30 Jul 2020 09:55:11 ...	Thu, 30 Jul 2020 09:55:12 ...	Thu, 30 Jul 2020 09:55:12 ...	Thu, 30 Jul 2020 09:55:12 ...
2	-oozie-... map-reduce-wf			Thu, 30 Jul 2020 09:59:07 ...	Thu, 30 Jul 2020 09:59:07 ...	Thu, 30 Jul 2020 09:59:07 ...	Thu, 30 Jul 2020 09:59:07 ...
3	-oozie-... map-reduce-wf			Thu, 30 Jul 2020 10:01:36 ...	Thu, 30 Jul 2020 10:01:36 ...	Thu, 30 Jul 2020 10:01:36 ...	Thu, 30 Jul 2020 10:01:36 ...

----End

## 19.2 Using the Oozie Client

### Scenario

This section describes how to use the Oozie client in an O&M scenario or service scenario.

## Prerequisites

- The client has been installed. For example, the installation directory is **/opt/hadoopclient**. The client directory in the following operations is only an example. Change it to the actual installation directory.
- Service component users are created by the MRS cluster administrator as required. In security mode, machine-machine users need to download the keytab file. A human-machine user must change the password upon the first login.

## Using the Oozie Client

**Step 1** Log in to the node where the client is installed as the client installation user.

**Step 2** Run the following command to go to the client installation directory.

```
cd /opt/hadoopclient
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 4** Check the cluster authentication mode.

- If the cluster is in security mode, run the following command to authenticate the user: *exampleUser* indicates the name of the user who submits tasks.

```
kinit exampleUser
```

- If the cluster is in normal mode, go to [Step 5](#).

**Step 5** Perform the following operations to configure Hue:

1. Configure the Spark2x environment (skip this step if the Spark2x task is not involved):

```
hdfs dfs -put /opt/hadoopclient/Spark2x/spark/jars/*.jar /user/oozie/
share/lib/spark2x/
```

2. Upload the Oozie configuration file and JAR package to HDFS.

```
hdfs dfs -mkdir /user/exampleUser
```

```
hdfs dfs -put -f /opt/hadoopclient/Oozie/oozie-client-*/examples /user/
exampleUser/
```

 NOTE

- *exampleUser* indicates the name of the user who submits tasks.
- If the user who submits the task and other files except **job.properties** are not changed, client installation directory **/Oozie/oozie-client-xxx/examples** can be repeatedly used after being uploaded to HDFS.
- When the JAR package in the HDFS directory **/user/oozie/share** changes, you need to restart the Oozie service.
- Resolve the JAR file conflict between Spark and Yarn about Jetty.

```
hdfs dfs -rm -f /user/oozie/share/lib/spark/jetty-all-9.2.22.v20170606.jar
```

- In normal mode, if **Permission denied** is displayed during the upload, run the following commands:

```
su - omm
source /opt/hadoopclient/bigdata_env
hdfs dfs -chmod -R 777 /user/oozie
exit
```

----End

## 19.3 Using Oozie Client to Submit an Oozie Job

### 19.3.1 Submitting a Hive Job

#### Scenario

This section describes how to use the Oozie client to submit a Hive job.

Hive jobs are divided into the following types:

- Hive job  
Hive job that is connected in JDBC mode
- Hive2 job  
Hive job that is connected in Beeline mode

This section describes how to submit a Hive job using the Oozie client.

 NOTE

- The procedure for submitting a Hive2 job using the Oozie client is the same as that for submitting a Hive job. You only need to change **/Hive** in the procedure to **/Hive2**.  
For example, if the Hive job running directory is **/opt/client/Oozie/oozie-client-xxx/examples/apps/hive/**, then the running directory of Hive2 is **/opt/client/Oozie/oozie-client-xxx/examples/apps/hive2/**.
- You are advised to download the latest client.

#### Prerequisites

- The Hive and Oozie components and clients have been installed and are running properly.
- You have created or obtained the human-machine account and password for accessing the Oozie service.

 **NOTE**

- This user must belong to the **hadoop**, **supergroup**, and **hive** groups and be assigned with the Oozie role operation permission. If the multi-instance function is enabled for Hive, the user must belong to a specific Hive instance group, for example, **hive3**.
- This user must also be assigned the **manager\_viewer** role at least.
- You have obtained the URL of the Oozie server (any instance) in the running state, for example, **https://10.1.130.10:21003/oozie**.
- You have obtained the name of the Oozie server, for example, **10-1-130-10**.
- You have obtained the IP address of the active Yarn ResourceManager, for example, **10.1.130.11**.

## Procedure

**Step 1** Log in to the node where the Oozie client is installed as the client installation user.

**Step 2** Run the following command to obtain the installation environment. In the command, **/opt/client/** indicates the client installation path.

```
source /opt/client/bigdata_env
```

**Step 3** Check the cluster authentication mode.

- If the cluster is in security mode, run the **kinit** command to authenticate users.

For example, the **oozieuser** user is authenticated using the following command:

```
kinit oozieuser
```

- If the cluster is in normal mode, go to [Step 4](#).

**Step 4** Run the following command to go to the example directory:

```
cd /opt/client/Oozie/oozie-client-*/examples/apps/hive/
```

[Table 19-1](#) lists the files that you need to pay attention to in the directory.

**Table 19-1** File description

File	Description
hive-site.xml	Configuration file of a Hive job
job.properties	Parameter definition file of a workflow
script.q	SQL script of a Hive job
workflow.xml	Rule definition file of a workflow

**Step 5** Run the following command to edit the **job.properties** file:

```
vi job.properties
```

Perform the following modifications:

Change the value of **userName** to the name of the human-machine user who submits the job, for example, **userName=oozieuser**.

**Step 6** Run the **oozie job** command to run the workflow file:

```
oozie job -oozie https://Host name of the Oozie role:21003/oozie/ -config
job.properties -run
```

 **NOTE**

- The command parameters are described as follows:
  - oozie** URL of the Oozie server that executes a job
  - config** Workflow property file
  - run** Executing a workflow
- If a job ID, for example, **job: 0000021-140222101051722-oozie-omm-W**, is displayed after the workflow file is executed, the job is successfully submitted. You can view the execution results on the Oozie management page.

Log in to the Oozie web UI at **https://IP address of the Oozie role:21003/oozie** as user **oozieuser**.

On the Oozie web UI, you can view the submitted workflow information based on the job ID in the table on the page.

----End

## 19.3.2 Submitting a Spark2x Job

### Scenario

This section describes how to submit a Spark2x job using the Oozie client.

 **NOTE**

You are advised to download the latest client.

### Prerequisites

- The Spark2x and Oozie components and clients have been installed and are running properly.

If the current client is an earlier version, you need to download and install the client again.
- You have created or obtained the human-machine account and password for accessing the Oozie service.

 **NOTE**

- This user must belong to the **hadoop**, **supergroup**, and **hive** groups and be assigned with the Oozie role operation permission. If the multi-instance function is enabled for Hive, the user must belong to a specific Hive instance group, for example, **hive3**.
- This user must also be assigned the **manager\_viewer** role at least.
- You have obtained the URL of the Oozie server (any instance) in the running state, for example, **https://10.1.130.10:21003/oozie**.
- You have obtained the name of the Oozie server, for example, **10-1-130-10**.

- You have obtained the IP address of the active Yarn ResourceManager, for example, **10.1.130.11**.

## Procedure

**Step 1** Log in to the node where the Oozie client is installed as the client installation user.

**Step 2** Run the following command to obtain the installation environment. In the preceding command, **/opt/client/** indicates the client installation path.

```
source /opt/client/bigdata_env
```

**Step 3** Check the cluster authentication mode.

- If the cluster is in security mode, run the **kinit** command to authenticate users.

For example, the **oozieuser** user is authenticated using the following command:

```
kinit oozieuser
```

- If the cluster is in normal mode, go to [Step 4](#).

**Step 4** Run the following command to go to the example directory:

```
cd /opt/client/Oozie/oozie-client-*/examples/apps/spark2x/
```

[Table 19-2](#) lists the files that you need to pay attention to in the directory.

**Table 19-2** File description

File	Description
job.properties	Parameter definition file of a workflow
workflow.xml	Rule definition file of a workflow
lib	Directory of the JAR file on which a workflow depends

**Step 5** Run the following command to edit the **job.properties** file:

```
vi job.properties
```

Perform the following modifications:

Change the value of **userName** to the name of the human-machine user who submits the job, for example, **userName=oozieuser**.

**Step 6** Run the **oozie job** command to run the workflow file:

```
oozie job -oozie https://Host name of the Oozie role:21003/oozie/ -config job.properties -run
```

 NOTE

- The command parameters are described as follows:
  - oozie** URL of the Oozie server that executes a job
  - config** Workflow property file
  - run** Executing a workflow
- If a job ID, for example, **job: 0000021-140222101051722-oozie-omm-W**, is displayed after the workflow file is executed, the job is successfully submitted. You can view the execution results on the Oozie management page.

Log in to the Oozie web UI at **https://IP address of the Oozie role:21003/oozie** as user **oozieuser**.

On the Oozie web UI, you can view the submitted workflow information based on the job ID in the table on the page.

----End

## 19.3.3 Submitting a Loader Job

### Scenario

This section describes how to submit a Loader job using the Oozie client.

 NOTE

You are advised to download the latest client.

### Prerequisites

- The Hive and Oozie components and clients have been installed and are running properly.
- You have created or obtained the human-machine account and password for accessing the Oozie service.

 NOTE

- This user must belong to the **hadoop**, **supergroup**, and **hive** groups and be assigned with the Oozie role operation permission. If the multi-instance function is enabled for Hive, the user must belong to a specific Hive instance group, for example, **hive3**.
- This user must also be assigned the **manager\_viewer** role at least.
- You have obtained the URL of the Oozie server (any instance) in the running state, for example, **https://10.1.130.10:21003/oozie**.
- You have obtained the name of the Oozie server, for example, **10-1-130-10**.
- You have obtained the IP address of the active Yarn ResourceManager, for example, **10.1.130.11**.
- You have created a Loader job to be scheduled and obtained the job ID.

### Procedure

**Step 1** Log in to the node where the Oozie client is installed as the client installation user.

**Step 2** Run the following command to obtain the installation environment. In the command, **/opt/client/** indicates the client installation path.



```
source /opt/client/bigdata_env
```

**Step 3** Check the cluster authentication mode.

- If the cluster is in security mode, run the **kinit** command to authenticate users.

For example, the **oozieuser** user is authenticated using the following command:

```
kinit oozieuser
```

- If the cluster is in normal mode, go to [Step 4](#).

**Step 4** Run the following command to go to the example directory:

```
cd /opt/client/Oozie/oozie-client-*/examples/apps/sqoop/
```

[Table 19-3](#) lists the files that you need to pay attention to in the directory.

**Table 19-3** File description

File	Description
job.properties	Parameter definition file of a workflow
workflow.xml	Rule definition file of a workflow

**Step 5** Run the following command to edit the **job.properties** file:

```
vi job.properties
```

Perform the following modifications:

Change the value of **userName** to the name of the human-machine user who submits the job, for example, **userName=oozieuser**.

**Step 6** Run the following command to edit the **workflow.xml** file:

```
vi workflow.xml
```

Perform the following modifications:

Change the value of **command** to the ID of the Loader job to be scheduled, for example, **1**.

Upload the **workflow.xml** file to the HDFS path in the **job.properties** file.

```
hdfs dfs -put -f workflow.xml /user/userName/examples/apps/sqoop
```

**Step 7** Run the **oozie job** command to run the workflow file:

```
oozie job -oozie https://Host name of the Oozie role:21003/oozie/ -config job.properties -run
```

 NOTE

- The command parameters are described as follows:
  - oozie** URL of the Oozie server that executes a job
  - config** Workflow property file
  - run** Executing a workflow
- If a job ID, for example, **job: 0000021-140222101051722-oozie-omm-W**, is displayed after the workflow file is executed, the job is successfully submitted. You can view the execution results on the Oozie management page.

Log in to the Oozie web UI at **https://IP address of the Oozie role:21003/oozie** as user **oozieuser**.

On the Oozie web UI, you can view the submitted workflow information based on the job ID in the table on the page.

----End

## 19.3.4 Submitting Other Jobs

### Scenario

In addition to Hive, Spark2x, and Loader jobs, MapReduce, Java, Shell, HDFS, SSH, SubWorkflow, Streaming, and scheduled jobs can be submitted using the Oozie client.

 NOTE

You are advised to download the latest client.

### Prerequisites

- The Oozie component and its client have been installed and are running properly.
- You have created or obtained the human-machine account and password for accessing the Oozie service.

 NOTE

- Shell job:

This user must belong to the **hadoop** and **supergroup** groups and be assigned the Oozie role operation permission. The Shell script must have the execution permission on each NodeManager.
- SSH job:

This user must belong to the **hadoop** and **supergroup** groups and be assigned the Oozie role operation permission. The mutual trust configuration is complete.
- Other jobs:

This user must belong to the **hadoop** and **supergroup** groups and be assigned the Oozie role operation permission and other required permissions.

  - This user must also be assigned the **manager\_viewer** role at least.
- You have obtained the URL of the Oozie server (any instance) in the running state, for example, **https://10.1.130.10:21003/oozie**.
- You have obtained the name of the Oozie server, for example, **10-1-130-10**.
- You have obtained the IP address of the active Yarn ResourceManager, for example, **10.1.130.11**.

## Procedure

**Step 1** Log in to the node where the Oozie client is installed as the client installation user.

**Step 2** Run the following command to obtain the installation environment. In the command, **/opt/client/** indicates the client installation path.

```
source /opt/client/bigdata_env
```

**Step 3** Check the cluster authentication mode.

- If the cluster is in security mode, run the **kinit** command to authenticate users.

For example, the **oozieuser** user is authenticated using the following command:

```
kinit oozieuser
```

- If the cluster is in normal mode, go to [Step 4](#).

**Step 4** Go to the example directory based on the type of the task you submit.

**Table 19-4** List of example directories

Job Type	Example Directory
MapReduce job	<i>Client installation directory/Oozie/oozie-client-xxx/examples/apps/map-reduce</i>
Java job	<i>Client installation directory/Oozie/oozie-client-xxx/examples/apps/java-main</i>
Shell job	<i>Client installation directory/Oozie/oozie-client-xxx/examples/apps/shell</i>
Streaming job	<i>Client installation directory/Oozie/oozie-client-xxx/examples/apps/streaming</i>
SubWorkflow job	<i>Client installation directory/Oozie/oozie-client-xxx/examples/apps/subwf</i>
SSH job	<i>Client installation directory/Oozie/oozie-client-xxx/examples/apps/ssh</i>
Scheduled job	<i>Client installation directory/Oozie/oozie-client-xxx/examples/apps/cron</i>

### NOTE

The examples of other jobs contain HDFS job examples.

[Table 19-5](#) lists the files that you need to pay attention to in the example directory.

**Table 19-5** File description

File	Description
job.properties	Parameter definition file of a workflow
workflow.xml	Rule definition file of a workflow
lib	Directory of the JAR file on which a workflow depends
coordinator.xml	Scheduled job configuration file which can be used to set a scheduled policy. The file is in the <b>cron</b> directory.
oozie_shell.sh	Shell script file required for submitting shell jobs. The file is in the <b>shell</b> directory.

**Step 5** Run the following command to edit the **job.properties** file:

```
vi job.properties
```

Perform the following modifications:

Change the value of **userName** to the name of the human-machine user who submits the job, for example, **userName=oozieuser**.

**Step 6** Run the **oozie job** command to run the workflow file:

```
oozie job -oozie https://Host name of the oozie role:21003/oozie -config File path of job.properties -run
```

Example:

```
oozie job -oozie https://10-1-130-10:21003/oozie -config /opt/client/Oozie/oozie-client-*/examples/apps/map-reduce/job.properties -run
```

 **NOTE**

- The command parameters are described as follows:  
-**oozie** URL of the Oozie server that executes a job  
-**config** Workflow property file  
-**run** Executing a workflow
- If a job ID, for example, **job: 0000021-140222101051722-oozie-omm-W**, is displayed after the workflow file is executed, the job is successfully submitted. You can view the execution results on the Oozie management page.

Log in to the Oozie web UI at **https://IP address of the Oozie role:21003/oozie** as user **oozieuser**.

On the Oozie web UI, you can view the submitted workflow information based on the job ID in the table on the page.

----End

## 19.4 Using Hue to Submit an Oozie Job

### 19.4.1 Creating a Workflow

#### Scenario


You can submit an Oozie job on the Hue management page, but a workflow must be created before the job is submitted.

#### Prerequisites

Before using Hue to submit an Oozie job, configure the Oozie client and upload the sample configuration file and JAR file to the specified HDFS directory. For details, see [Using the Oozie Client](#).

#### Procedure

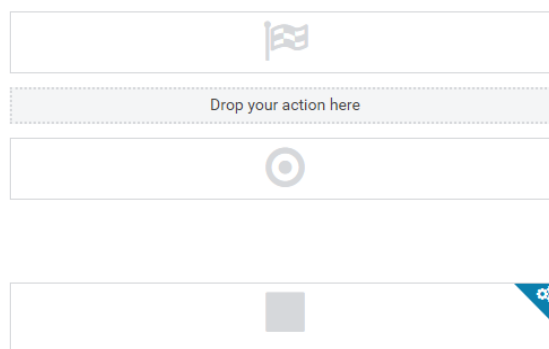
**Step 1** Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

**Step 2** In the navigation tree on the left, click  and choose **Workflow** to open the Workflow editor.

**Step 3** Select **Actions** from the **DOCUMENTS** drop-down list, select the job type to be created and drag it to the operation area.



My Workflow  
[Add a description...](#)



For submitting different job types, follow instructions in the following sections:

- [Submitting a Hive2 Job](#)
- [Submitting a Spark2x Job](#)
- [Submitting a Java Job](#)

- [Submitting a Loader Job](#)
- [Submitting a MapReduce Job](#)
- [Submitting a Sub-workflow Job](#)
- [Submitting a Shell Job](#)
- [Submitting an HDFS Job](#)
- [Submitting a Streaming Job](#)
- [Submitting a DistCp Job](#)

----End

## 19.4.2 Submitting a Workflow Job


### 19.4.2.1 Submitting a Hive2 Job

#### Scenario

This section describes how to submit an Oozie job of the Hive2 type on the Hue web UI.

#### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).

**Step 2** On the workflow editing page, select  next to **HiveServer2 Script** and drag it to the operation area.

**Step 3** In the **HiveServer2 Script** dialog box that is displayed, configure the script path in the HDFS, for example, `/user/admin/examples/apps/hive2/script.q`, and click **Add**.

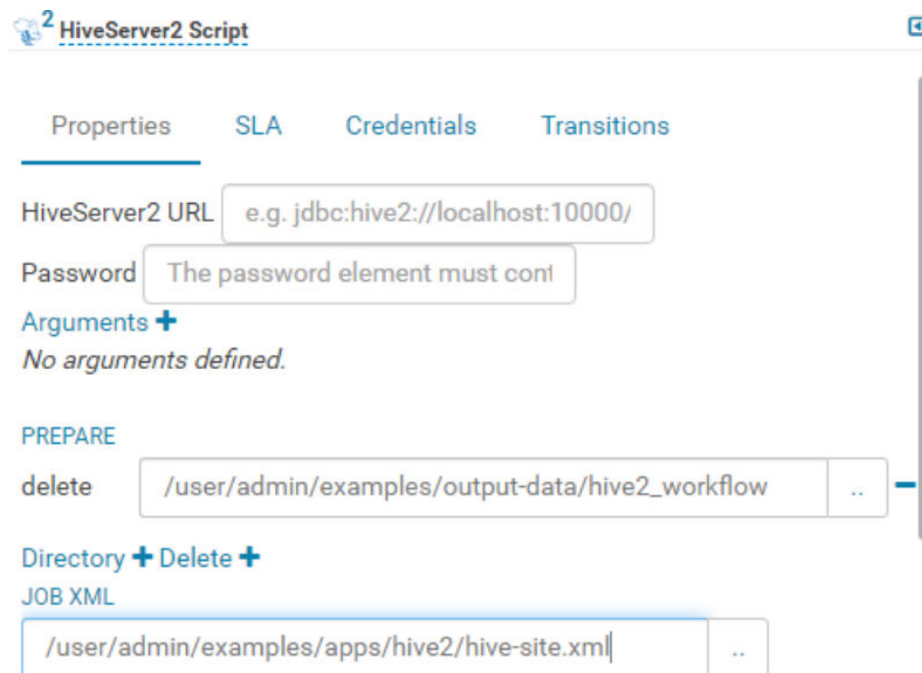
**Step 4** Click **PARAMETER+** to add input and output parameters.

For example, if the input parameter is `INPUT=/user/admin/examples/input-data/table`, the output parameter is `OUTPUT=/user/admin/examples/output-data/hive2_workflow`.




**Step 5** Click the configuration button  in the upper right corner. On the configuration page that is displayed, click **Delete +** to delete a directory, for example, `/user/admin/examples/output-data/hive2_workflow`.

**Step 6** Configure the job XML, for example, to the HDFS path `/user/admin/examples/apps/hive2/hive-site.xml`.



**NOTE**

If the preceding parameters and values are modified, you can query them in **Oozie client installation directory/oozie-client-xxx/conf/hive-site.xml**.

**Step 7** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Hive2-Workflow**.

**Step 8** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End


## 19.4.2.2 Submitting a Spark2x Job

### Scenario

This section describes how to submit an Oozie job of the Spark2x type on Hue.

### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).

- Step 2** On the workflow editing page, select  next to **Spark program** and drag it to the operation area.
- Step 3** In the Spark window that is displayed, set the value of **Files**, for example, to **hdfs://hacluster/user/admin/examples/apps/spark2x/lib/oozie-examples.jar**. Set the value of **jar/py name**, for example, to **org.apache.oozie.example.SparkFileCopy**, and click **Add**.
- Step 4** Set the value of **Main class**, for example, **org.apache.oozie.example.SparkFileCopy**.
- Step 5** Click **PARAMETER+** to add related input and output parameters.


For example, add the following parameters:


- **hdfs://hacluster/user/admin/examples/input-data/text/data.txt**
- **hdfs://hacluster/user/admin/examples/output-data/spark\_workflow**

- Step 6** In the **Options list** text box, specify Spark parameters, for example, **--conf spark.yarn.archive=hdfs://hacluster/user/spark2x/jars/8.0.2.1/spark-archive-2x.zip --conf spark.eventLog.enabled=true --conf spark.eventLog.dir=hdfs://hacluster/spark2xJobHistory2x**.

 **NOTE**

The version 8.0.2.1 is used as an example. Replace it with the actual version number.

- Step 7** Click the configuration button  in the upper right corner. Set the value of **Spark Master**, for example, to **yarn-cluster**. Set the value of **Mode**, for example, **cluster**.
- Step 8** On the configuration page that is displayed, click **Delete +** to delete a directory, for example, **hdfs://hacluster/user/admin/examples/output-data/spark\_workflow**.
- Step 9** Click **PROPERTIES+** and add **sharelib** used by Oozie. Enter the attribute name **oozie.action.sharelib.for.spark** in the left text box and the attribute value **spark2x** in the right text box.

- Step 10** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Spark-Workflow**.

- Step 11** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End



### 19.4.2.3 Submitting a Java Job

#### Scenario


This section describes how to submit an Oozie job of the Java type on the Hue web UI.

#### Procedure


**Step 1** Create a workflow. For details, see [Creating a Workflow](#).

**Step 2** On the workflow editing page, select  next to **Java program** and drag it to the operation area.

**Step 3** In the **Jar program** window that is displayed, set the value of **Jar name**, for example, `/user/admin/examples/apps/java-main/lib/oozie-examples-5.1.0.jar`. Set the value of **Main class**, for example, `org.apache.oozie.example.DemoJavaMain`. Click **Add**.

**Step 4** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Java-Workflow**.

**Step 5** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End

### 19.4.2.4 Submitting a Loader Job

#### Scenario

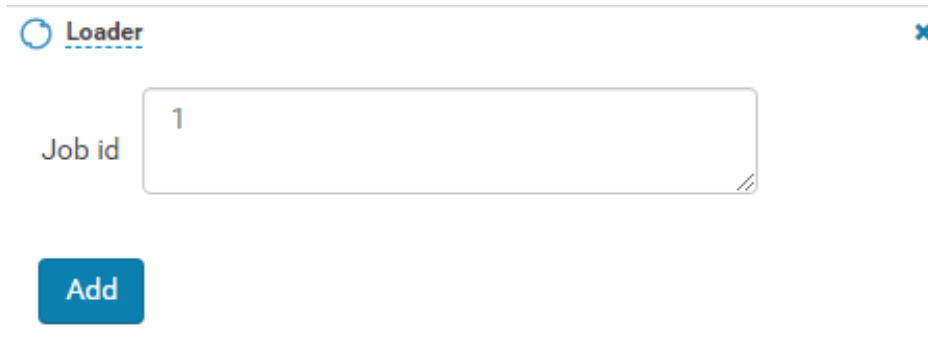
This section describes how to submit an Oozie job of the Loader type on the Hue web UI.

#### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).

**Step 2** On the workflow editing page, select  next to **Loader** and drag it to the operation area.


**Step 3** In the **Loader** window that is displayed, set **Job id**, for example, to **1**. Click **Add**.




 **NOTE**

**Job id** is the ID of the Loader job to be orchestrated and can be obtained from the Loader page.

You can create a Loader job to be scheduled and obtain its job ID. For details, see [Using Loader](#).

**Step 4** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Loader-Workflow**.

**Step 5** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End

## 19.4.2.5 Submitting a MapReduce Job

### Scenario

This section describes how to submit an Oozie job of the MapReduce type on the Hue web UI.

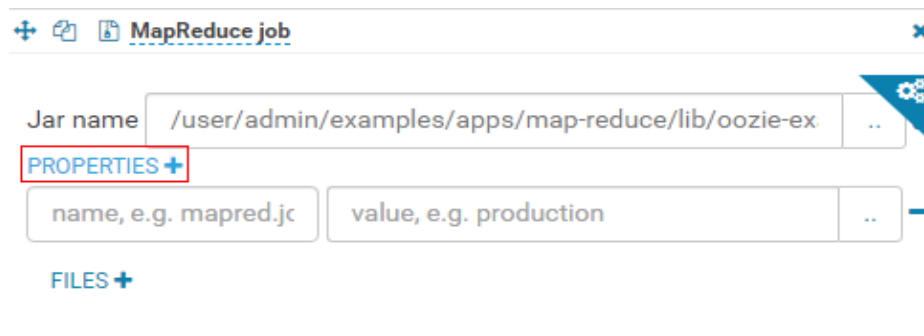
### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).

**Step 2** On the workflow editing page, select  next to **MapReduce job** and drag it to the operation area.


**Step 3** In the displayed **MapReduce job** dialog box, set **Jar name**, for example, to `/user/admin/examples/apps/map-reduce/lib/oozie-examples-5.1.0.jar`. Click **Add**.

**Step 4** Click **PROPERTIES+** to add input and output properties.



For example, set the value of **mapred.input.dir** to **/user/admin/examples/input-data/text** and set the value of **mapred.output.dir** to **/user/admin/examples/output-data/map-reduce\_workflow**.

**Step 5** Click the configuration button  in the upper right corner. On the configuration page that is displayed, click **Delete +** to delete a directory, for example, **/user/admin/examples/output-data/map-reduce\_workflow**.

**Step 6** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **MapReduce-Workflow**.

**Step 7** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End

### 19.4.2.6 Submitting a Sub-workflow Job

#### Scenario

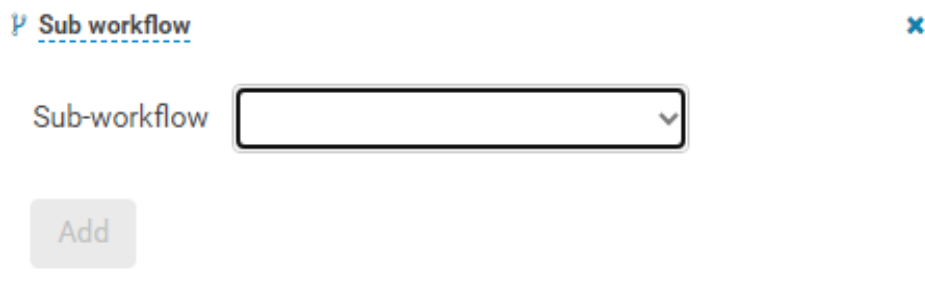
This section describes how to submit an Oozie job of the Sub-workflow type on the Hue web UI.


#### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).


**Step 2** On the workflow editing page, select  next to **Sub workflow** and drag it to the operation area.

**Step 3** In the **Sub workflow** dialog box that is displayed, set **Sub-workflow**, for example, to **Java-Workflow** (one of the created workflows) from the drop-down list box, and click **Add**.



**Step 4** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Subworkflow-Workflow**.

**Step 5** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End

### 19.4.2.7 Submitting a Shell Job

#### Scenario

This section describes how to submit an Oozie job of the Shell type on the Hue web UI.

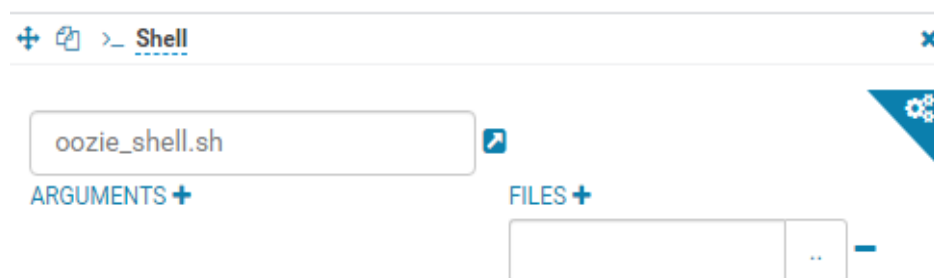
#### Procedure


**Step 1** Create a workflow. For details, see [Creating a Workflow](#).

**Step 2** On the workflow editing page, select  next to **Shell** and drag it to the operation area.


**Step 3** In the **Shell** window that is displayed, set **Shell command**, for example, to **oozie\_shell.sh**, and click **Add**.

**Step 4** Click **FILE+** to add the Shell command execution file and Oozie example execution file, for example, **/user/admin/examples/apps/shell/oozie\_shell.sh**.



**Step 5** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Shell-Workflow**.

**Step 6** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

 **NOTE**

- When configuring a shell command as a Linux command, specify it as the original command instead of the shortcut key command. For example, do not set **ls -l** to **ll**. You can configure it as the shell command **ls**, and add a parameter **-l**.
- When uploading the shell script to HDFS on Windows, make sure that the shell script format is Unix. If the format is incorrect, the shell job fails to be submitted.

----End


## 19.4.2.8 Submitting an HDFS Job

### Scenario

This section describes how to submit an Oozie job of the HDFS type on the Hue web UI.


### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).


**Step 2** On the workflow editing page, select  next to **Fs** and drag it to the operation area.

**Step 3** In the **Fs** window that is displayed, click **Add**.

**Step 4** Click **CREATE DIRECTORY+** to add the HDFS directories to be created, for example, **/user/admin/examples/output-data/mkdir\_workflow** and **/user/admin/examples/output-data/mkdir\_workflow1**.

**Step 5** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **HDFS-Workflow**.

**Step 6** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End

## 19.4.2.9 Submitting a Streaming Job

### Scenario

This section describes how to submit an Oozie job of the Streaming type on the Hue web UI.

### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).

**Step 2** On the workflow editing page, select  next to **Streaming** and drag it to the operation area.

**Step 3** In the **Streaming** window that is displayed, set **Mapper**, for example, to `/bin/cat`. Set **Reducer**, for example, to `/usr/bin/wc`. Click **Add**.


**Step 4** Click **FILE+** to add the files required for running.

for example, `/user/oozie/share/lib/mapreduce-streaming/hadoop-streaming-3.1.1.jar` and `/user/oozie/share/lib/mapreduce-streaming/oozie-sharelib-streaming-5.1.0.jar`.

**Step 5** Click the configuration button  in the upper right corner. On the configuration page that is displayed, click **Delete+** to delete a directory, for example, `/user/admin/examples/output-data/streaming_workflow`.

**Step 6** Click **PROPERTIES+** to add the following properties:

- Enter the property name `mapred.input.dir` in the left box and enter the property value `/user/admin/examples/input-data/text` in the right box.
- Enter the property name `mapred.output.dir` in the left box and enter the attribute value `/user/admin/examples/output-data/streaming_workflow` in the right box.

**Step 7** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Streaming-Workflow**.

**Step 8** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End

### 19.4.2.10 Submitting a DistCp Job

#### Scenario


This section describes how to submit an Oozie job of the DistCp type on the Hue web UI.


#### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).


**Step 2** On the workflow editing page, select  next to **Distcp** and drag it to the operation area.

**Step 3** In the **Distcp** window that is displayed, set the value of **Source**, for example, to `/user/admin/examples/input-data/text/data.txt`. Set **Destination**, for example, to `/user/admin/examples/output-data/distcp-workflow/data.txt`. Click **Add**.

**Step 4** Click  in the upper right corner. On the configuration page that is displayed, click **Delete+** and add the directory to be deleted, for example, `/user/admin/examples/output-data/distcp-workflow`.

**Step 5** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Distcp-Workflow**.

**Step 6** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End

### 19.4.2.11 Example of Mutual Trust Operations

#### Scenario

This section guides you to enable unidirectional password-free mutual trust when Oozie nodes are used to execute shell scripts of external nodes through SSH jobs.

#### Prerequisites

You have installed Oozie, and it can communicate with external nodes (nodes connected using SSH).

#### Procedure

**Step 1** Ensure that the user used for SSH connection exists on the external node, and the user directory `~/.ssh` exists.

**Step 2** Log in to the Oozie node as user **omm** and run the **ssh-keygen -t rsa** command to generate public and private keys.

**Step 3** Run the **cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys** statement to add the public key to the **authorized\_keys** file.

**Step 4** Upload the **id\_rsa.pub** file to an existing directory, for example, **/opt/**, on the external node as user **root**.

```
scp ~/.ssh/id_rsa.pub root@IP address of the external node:/opt/id_rsa.pub
```

**Step 5** Log in to the external node where the shell is located and go to the directory described in **Step 4**. The **id\_rsa.pub** file can be found.

Run the **cat id\_rsa.pub >> ~/.ssh/authorized\_keys** statement to add the public key to the **authorized\_keys** file of the shell user.

**Step 6** Change the permission on the directory.

```
chmod 700 ~/.ssh
```

```
chmod 600 /opt/id_rsa.pub
```

```
chmod 600 ~/.ssh/authorized_keys
```

 **NOTE**

- The user of the node where shell resides (external node) has the permission to execute shell scripts and access all directories and files involved in the Shell scripts.
- If Oozie has multiple nodes, perform **Step 2** to **Step 6** on all Oozie nodes.

----End

## 19.4.2.12 Submitting an SSH Job


### Scenario

This section guides you to submit an Oozie job of the SSH type on the Hue web UI.


### Procedure

**Step 1** Create a workflow. For details, see [Creating a Workflow](#).

**Step 2** For details about how to add the trust relationship, see [Example of Mutual Trust Operations](#).


**Step 3** On the workflow editing page, select the **Ssh** button  and drag it to the operation area.

**Step 4** In the **Ssh** window that is displayed, set **User and Host** and **Ssh command** commands and click **Add**.

**Step 5** Click  in the upper right corner of the Oozie editor.

If you need to modify the job name before saving the job (default value: **My Workflow**), click the name directly for modification, for example, **Ssh-Workflow**.



**Step 6** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End


### 19.4.2.13 Submitting a Hive Script


#### Scenario

This section describes how to submit a Hive job on the Hue web UI.

#### Procedure

**Step 1** Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).


**Step 2** In the navigation tree on the left, click  and choose **Workflow** to open the Workflow editor.


**Step 3** Click **Documents**, click  to select a Hive script from the operation list, and drag it to the operation page.

**Step 4** In the **HiveServer2 Script** dialog box that is displayed, select the saved Hive script. For details about how to save the Hive script, see [Using HiveQL Editor on the Hue Web UI](#). Select a script and click **Add**.



**Step 5** Configure the Job XML, for example, to the HDFS path `/user/admin/examples/apps/hive2/hive-site.xml`. For details, see [Submitting a Hive2 Job](#).

**Step 6** Click  in the upper right corner of the Oozie editor.

**Step 7** After the configuration is saved, click , and submit the job.

After the job is submitted, you can view the related contents of the job, such as the detailed information, logs, and processes, on Hue.

----End

## 19.4.3 Submitting a Coordinator Periodic Scheduling Job

### Scenario


This section describes how to submit a job of the periodic scheduling type on the Hue web UI.

### Prerequisites

Required workflow jobs have been configured before the coordinator task is submitted.

### Procedure

**Step 1** Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

**Step 2** In the navigation tree on the left, click  and choose **Schedule** to open the Coordinator editor.

**Step 3** On the job editing page, click **My Schedule** to change the job name.


**Step 4** Click **Choose a Workflow...** to select the workflow to be orchestrated.

**My Schedule**

[Add a description...](#)


Which workflow to schedule?

[Choose a workflow...](#)

**Step 5** Select a workflow, set the job execution frequency as prompted, and click  in the upper right corner to save the workflow job.

#### NOTE

Because the time zone is changed, the difference between the time and the local time may be several hours.

**Step 6** Click  in the upper right corner of the editor, set the start value and end value of the time range for executing the scheduled job, and click **Submit** to submit the job.

 **NOTE**

Because the time zone is changed, the difference between the time and the local time may be several hours.

----End

## 19.4.4 Submitting a Bundle Batch Processing Job





### Scenario

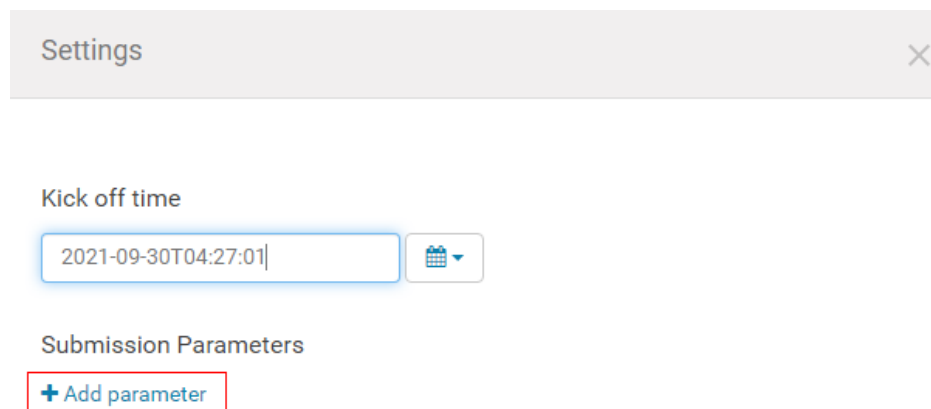
In the case that multiple scheduled jobs exist at the same time, you can manage the jobs in batches over the Bundle task. This section describes how to submit a job of the batch type on the Hue web UI.

### Prerequisites

Required related workflow and Coordinator jobs have been configured before the Bundle batch processing job is submitted.


### Procedure

- Step 1** Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).
- Step 2** In the navigation tree on the left, click  and choose **Bundle** to open the Bundle editor.
- Step 3** On the job editing page, click **My Bundle** to change the job name.
- Step 4** Click **+Add a coordinator** to select the Coordinator job to be orchestrated.
- Step 5** Set the start time and the end time for the scheduled coordinator jobs as prompted and click  in the upper right corner to save the job.
- Step 6** Click  in the upper right corner of the editor, select  from the displayed menu, set the start time of the bundle task, click **+Add parameter** to add parameters, and close the dialog box to save the settings.



 NOTE

Because the time zone is changed, the difference between the time and the local time may be several hours.

**Step 7** Click  in the upper right corner of the editor. In the dialog box that is displayed, click **Submit** to submit the job.

----End


## 19.4.5 Querying the Operation Results

### Scenario

After the jobs are submitted, you can view the execution status of a specific job on Hue.

### Procedure

**Step 1** Access the Hue web UI. For details, see [Accessing the Hue Web UI](#).

**Step 2** Click . On the displayed page, you can view information about the Workflow, Schedule, and Bundle tasks.

----End

## 19.5 Oozie Log Overview

### Log Description

**Log path:** The default storage paths of Oozie log files are as follows:

- Run log: `/var/log/Bigdata/oozie`
- Audit log: `/var/log/Bigdata/audit/oozie`

**Log archiving rule:** Oozie logs are classified into run logs, script logs, and audit logs. The maximum size of a run log file is 20 MB, and a maximum of 20 run log files can be reserved. The maximum size of an audit log file is 20 MB, and a maximum of 20 audit log files can be reserved.

 NOTE

A compressed log file is generated for **oozie.log** every hour. 720 compressed files (log files of one month) are retained by default.

**Table 19-6** Oozie log list

Log Type	Log File Name	Description
Run log	jetty.log	Oozie built-in jetty server log file, which is used to process the request and response information of OozieServlet
	jetty.out	Oozie process startup log file
	oozie_db_temp.log	Oozie database connection log
	oozie-instrumentation.log	Oozie dashboard log file, which records the Oozie running status and configuration information of each component
	oozie-jpa.log	openJPa run log file
	oozie.log	Oozie run log file
	oozie-<SSH_USER>-<DATE>-<PID>-gc.log	Log file that records the garbage collection of the Oozie service
	oozie-ops.log	Oozie operation log file
	threadDump-<DATE>.log	Log file that records stack information when the service process exits normally
Script logs	postinstallDetail.log	Work log file generated after the installation and before the startup
	prestartDetail.log	Pre-startup log file
	startDetail.log	Service startup log file
	stopDetail.log	Service stop log file
	upload-sharelib.log	Operation logs uploaded by <b>sharelib</b>
Audit log	oozie-audit.log	Audit log

## Log Level

**Table 19-7** describes the log levels provided by Oozie.

The priorities of log levels are ERROR, WARN, INFO, and DEBUG in descending order. Logs whose levels are higher than or equal to the set level are printed. The number of printed logs decreases as the configured log level increases.

**Table 19-7** Log levels

Level	Description
ERROR	Logs of this level record abnormal information about events that cause process exceptions.
WARN	Logs of this level record exception information about the current event processing.
INFO	Logs of this level record normal running status information about the system and events.
DEBUG	Logs of this level record system information and information about database underlying data transmission.

To modify log levels, perform the following operations:

- Step 1** Log in to FusionInsight Manager.
- Step 2** Choose **Cluster** > *Name of the desired cluster* > **Services** > **Oozie** > **Configurations**.
- Step 3** Select **All Configurations**.
- Step 4** On the menu bar on the left, select the log menu of the target role.
- Step 5** Select a desired log level.
- Step 6** Click **Save**, and then click **OK**. The settings take effect after the processing is complete.

----End

## Log Formats

The following table lists the Oozie log formats.

**Table 19-8** Log formats

Log Type	Format	Example
Run log	<yyyy-MM-dd HH:mm:ss,SSS><Log level><Location where the log event occurs><Log level><Message in the log>	2015-05-29 21:01:45,268 INFO StatusTransitService\$StatusTransitRun- nable:539 - USER[-] GROUP[-] Released lock for [org.apache.oozie.service.StatusTransitSe rvice]
Script logs	<yyyy-MM-dd HH:mm:ss,SSS><Host name > <Log level > <Message in the log>	2015-06-01 17:18:03 001 suse11-192-168-0-111 oozie INFO Running oozie service check script

Log Type	Format	Example
Audit log	<i>&lt;yyyy-MM-dd HH:mm:ss,SSS&gt; &lt;Log Level&gt; &lt; Thread name    Message in the log   Location where the log event occurs</i>	2015-06-01 22:38:41,323   INFO   http- bio-21003-exec-8   IP [192.168.0.111] USER [null], GROUP [null], APP [null], JOBID [null], OPERATION [null], PARAMETER [null], RESULT [SUCCESS], HTTPCODE [200], ERRORCODE [null], ERRORMESSAGE [null]   org.apache.oozie.util.XLog.log(XLog.java: 539)

## 19.6 Common Issues About Oozie

### 19.6.1 Oozie Scheduled Tasks Are Not Executed on Time

#### Question

Why are not Coordinator scheduled jobs executed on time on the Hue or Oozie client?

#### Answer

Use UTC time. For example, set **start=2016-12-20T09:00Z** in **job.properties** file.

### 19.6.2 Why Update of the share lib Directory of Oozie on HDFS Does Not Take Effect?

#### Symptom

A new JAR package is uploaded to the **/user/oozie/share/lib** directory on HDFS. However, an error indicating that the class cannot be found is reported during task execution.

#### Solution

Run the following command on the client to refresh the directory:

```
oozie admin -oozie https://xxx.xxx.xxx.xxx:21003/oozie -sharelibupdate
```

# 20 Using Presto

## 20.1 Accessing the Presto Web UI

You can view the Presto statistics on the graphical Presto web UI. You are advised to use Google Chrome to access the Presto web UI because it cannot be accessed using Internet Explorer.

### Prerequisites

- Presto has been installed in a cluster.
- The cluster client has been installed, for example, in the `/opt/client` directory. The client directory in the following operations is only an example. Change it based on the actual installation directory onsite.

### Accessing the Presto Web UI

- Method 1 (for MRS 3.x or later)
  - a. Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the target cluster* > **Services**.
  - b. Select **Presto**. In the **Basic Information** area, click **Coordinator(Coordinator)** next to **Coordinator WebUI**. The Coordinator web UI is displayed.

Figure 20-1 Coordinator WebUI

#### Basic Information

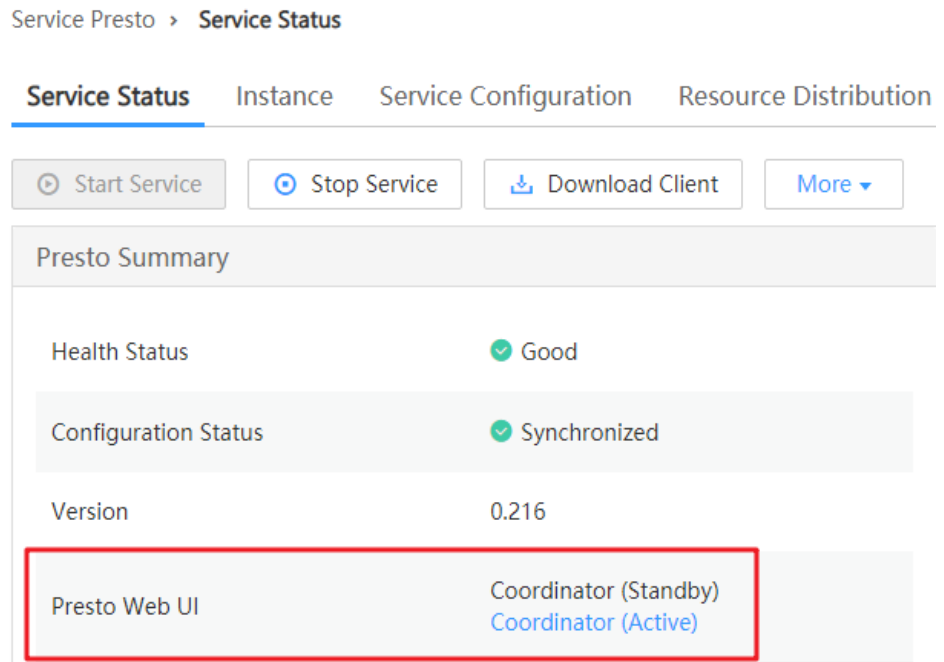
Running Status: ● Normal  
Configuration Status: ⊕ Synchronized  
Version: 333

Coordinator WebUI: [Coordinator\(Coordinator\)](#)  
[Coordinator\(Coordinator\)](#)



- Method 2 (for versions earlier than MRS 3.x)
  - a. Log in to MRS Manager and choose **Services**.
  - b. Select **Presto**. In the **Presto Summary** area, click **Coordinator (Active)** next to **Presto Web UI**. The Presto web UI is displayed.

Figure 20-2 Presto WebUI



**NOTE**

When accessing the Presto web UI for the first time, you must add the address to the trusted site list.

## 20.2 Using a Client to Execute Query Statements

You can perform an interactive query on an MRS cluster client. For clusters with Kerberos authentication enabled, users who submit topologies must belong to the **presto** group.

The Presto component of MRS 3.x does not support Kerberos authentication.

### Prerequisites

- The password of user **admin** has been obtained. The password of user **admin** is specified by the user during MRS cluster creation.
- The client has been updated.

## Procedure

- Step 1** For clusters with Kerberos authentication enabled, log in to MRS Manager and create a role with the **Hive Admin Privilege** permission. For details about how to create a role, see [Creating a Role](#).
- Step 2** Create a user that belongs to the **Presto** and **Hive** groups, bind the role created in [Step 1](#) to the user, and download the user authentication file. For details, see [Creating a User](#) and [Downloading a User Authentication File](#).
- Step 3** Upload the downloaded **user.keytab** and **krb5.conf** files to the node where the MRS client resides.

### NOTE

For clusters with Kerberos authentication enabled, [Step 2](#) to [Step 3](#) must be performed. For normal clusters, start from [Step 4](#).

- Step 4** Prepare a client based on service conditions and log in to the node where the client is installed.

For example, if you have updated the client on the Master2 node, log in to the Master2 node to use the client. For details, see [Updating a Client](#).

- Step 5** Run the following command to switch the user:

```
sudo su - omm
```

- Step 6** Run the following command to switch to the client directory, for example, **/opt/client**.

```
cd /opt/client
```

- Step 7** Run the following command to configure environment variables:

```
source bigdata_env
```

- Step 8** Connect to the Presto Server. The following provides two client connection methods based on the client type.

- Using the client provided by MRS
  - For clusters with Kerberos authentication disabled, run the following command to connect to the Presto Server of the cluster:  
**presto\_cli.sh**
  - For clusters with Kerberos authentication disabled, run the following command to connect to the Presto Server of other clusters. In the command, **ip** indicates the Presto Server IP address of the cluster, and **port** indicates the Presto Server port number. The default port is **7520**.  
**presto\_cli.sh --server http://ip:port**
  - For clusters with Kerberos authentication enabled, run the following command to connect to the Presto Server of the cluster:  
**presto\_cli.sh --krb5-config-path *krb5.conf file path* --krb5-principal *User's principal* --krb5-keytab-path *user.keytab file path* --user *presto username***
  - For clusters with Kerberos authentication enabled, run the following command to connect to the Presto Server of other clusters. In the

command, **ip** indicates the Presto Server IP address of the cluster, and **port** indicates the Presto Server port number. The default port is **7521**.

```
presto_cli.sh --krb5-config-path krb5.conf file path --krb5-principal
User's principal --krb5-keytab-path user.keytab file path --server
https://ip:port --krb5-remote-service-name Presto Server name
```

- Using the native client

The native client of Presto is **Presto/presto/bin/presto** in the client directory.

**Step 9** Run a query statement, for example, **show catalogs**.

 **NOTE**

For clusters with Kerberos authentication enabled, when querying **Hive Catalog** data, the user who runs the Presto client must have the permission to access Hive tables and run the **grant all on table [table\_name] to group hive** command in Hive beeline to grant permissions to the Hive group.

**Step 10** After the query is complete, run the following command to exit the client:

```
quit
```

```
----End
```

## 20.3 Using Presto to Dump Data in DLF

### Prerequisites

- The Presto component has been installed in an MRS cluster.
- You have synchronized IAM users. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)
- You have the permission to operate the OBS file system. For details, see and .
- The Presto permission has been configured. For details, see [Configuring Presto Permissions](#).

### Creating a Data Connection of the MRS PrestoSQL Type in DLF

**Step 1** In the left navigation pane of the DLF console, choose **Connection > Manage Connection**.

**Step 2** In the upper right corner of the page, click **Create Data Connection**.

**Step 3** Set parameters according to [Table 20-1](#).

**Table 20-1** Parameters for creating a data connection

Parameter	Description
Data Connection Type	Select <b>MRS PrestoSQL</b> .

Parameter	Description
Data Connection Name	Name of the data connection to be created, which contains 1 to 100 characters and consists of only letters, digits, underscores (_), and hyphens (-).
Cluster Name	Name of the MRS cluster to which Presto belongs.

**Step 4** Click **Test** to test connectivity of the data connection to be created. If the test passes, the data connection is created.

**Step 5** Click **OK**.

----End

## Creating and Executing SQL Scripts on the DLF Script Development Page

### CAUTION

In this scenario, the query results dumped to OBS can be retained for a maximum of 10,000 times. If the number of query times exceeds 10,000, the historical query results are automatically aged based on the query time sequence. To prevent data loss, exercise caution when performing this operation.

**Step 1** In the left navigation pane of the DLF console, choose **Development > Develop Script**.

**Step 2** In the right pane, click **Create SQL Script** and select **Presto**.

**Step 3** In the upper right part of the editor, select the connection created in [Creating a Data Connection of the MRS PrestoSQL Type in DLF](#) from the **Connection** drop-down list.

**Step 4** In the upper right part of the editor, select the schema from the **Schema** drop-down list box.

**Step 5** Enter one or more SQL statements in the editor. If you need to run a specified SQL statement separately, select the SQL statement before running it.

**Step 6** In the upper part of the editor, click **Execute**. After executing the SQL statement, view the execution history and result of the script in the lower part of the editor.

 **NOTE**

- Administrator operations are not supported. That is, all commands that can be executed only after the **set role admin** command is executed are not supported.
- Each statement is executed independently. Therefore, the statement with context settings (for example, **use**) does not take effect after being executed.
- When Presto authorization is enabled, the default permissions of various users are as follows:
  - All users have the read/write permissions on the **mrs\_reserved** database in Hive by default.
  - All IAM users with the MRS CommonOperations, MRS FullAccess, MRS Administrator, and Tenant Administrator policies have read/write permission on the **default** database in Hive, and users with the MRS ReadOnlyAccess policy have read-only permission on the database.
  - User **admin** of the cluster and IAM users with the MRS FullAccess, MRS Administrator, and Tenant Administrator policies have the **admin** role permission on the Hive database.

----End

## 20.4 Configuring Presto Permissions

### Configuring Presto Permissions in a Security Cluster

By default, the Hive Catalog authorization of the Presto component is enabled in a security cluster. The Presto permission configuration procedure is as follows:

- Step 1** Log in to Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#).
- Step 2** Choose **System > Manage Role**, configure a role that has the Hive database/table permissions, and bind the role to the user.

----End

### Configuring Presto Permissions in a Normal Cluster

By default, Presto authorization is not enabled in a normal cluster. You need to manually configure Presto permissions as follows:

- Step 1** Go to the MRS cluster details page.
- Step 2** Choose **Components > Hive**. Set **Type** to **All**. On the displayed Hive configuration page, modify parameter settings.
- Step 3** Search for and modify the following parameters:
  - Set **hive.security.authorization.enabled** to **true**.
  - Set **hive.security.authorization.manager** to **org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQLStdHiveAuthorizerFactory**.
- Step 4** Click **Save Configuration** and select **Restart the affected services or instances** to restart the Hive service.

- Step 5** Choose **Components > Presto**. Set **Type** to **All**. On the displayed Presto configuration page, modify parameter settings.
  - Step 6** Search for and modify the value of **hive.security** to **sql-standard-with-group**.
  - Step 7** Click **Save Configuration** and select **Restart the affected services or instances** to restart the Presto service.
  - Step 8** Logging in to MRS Manager
  - Step 9** Choose **System > Change OMS Database Password > Restart the OMS service**.
  - Step 10** Choose **System > Manage Role**, configure a role that has the Hive database/table permissions, and bind the role to the user.
- End

# 21 Using Ranger (MRS 1.9.2)

## 21.1 Creating a Ranger Cluster

**Step 1** Create a cluster by referring to **Configuring a Cluster > Custom Creation of a Cluster** in the *User Guide*. Select the Ranger component during cluster creation.

Currently, only normal MRS 1.9.2 clusters support Ranger. Security clusters with Kerberos authentication enabled do not support Ranger.

**Figure 21-1** Selecting the Ranger component

Cluster Type: Analysis cluster | Streaming cluster | Hybrid cluster

Mandatory components are selected by default. If you select other components, any dependent components will be automatically selected.

**Analysis Components**

<input checked="" type="checkbox"/>	Name	Version	Description
<input type="checkbox"/>	Presto	0.216	An open source distributed SQL query engine.
<input checked="" type="checkbox"/>	Hadoop	2.8.3	A distributed data storage and processing framework for large data sets, including core compon...
<input type="checkbox"/>	Spark	2.2.2	A fast and general engine for large-scale data processing.
<input type="checkbox"/>	HBase	1.3.1	A scalable, distributed database that supports structured data storage for large tables.
<input type="checkbox"/>	Opentsdb	2.3.0	One scalable, distributed time series database which can store and serve massive amounts of ti...
<input type="checkbox"/>	Hive	2.3.3	A data warehouse infrastructure that provides data summarization and ad hoc querying.
<input type="checkbox"/>	Hue	3.11.0	A component that provides the Hadoop UI capability, which enables users to analyze and proces...
<input type="checkbox"/>	Loader	2.0.0	A tool developed based on open source Sqoop 1.99.7, designed for efficiently transferring bulk d...
<input type="checkbox"/>	Tez	0.9.1	An application framework which allows for a complex directed-acyclic-graph of tasks for processi...
<input type="checkbox"/>	Flink	1.7.0	A framework and distributed processing engine for stateful computations over unbounded and b...
<input type="checkbox"/>	Alluxio	2.0.1	Alluxio is a memory speed virtual distributed storage system.
<input checked="" type="checkbox"/>	Ranger	1.0.1	Apache Ranger is a framework to enable, monitor and manage comprehensive data security acr...

Use External Data Sources to Store Metadata

**Step 2** Enable or disable **Use External Data Sources to Store Metadata**.

- Enabled: An external MySQL database is used to store the user, group, and policy data of Ranger.
- Disabled: The user, group, and policy data of Ranger is stored in the local database of the current cluster by default.

**Step 3** If **Use External Data Sources to Store Metadata** is enabled, set **Data Connection Type** to **RDS MySQL database**. Select an existing data connection instance or click **Create Data Connection** to create a data connection.

**Figure 21-2** Using the RDS MySQL database

Use External Data Sources to Store Metadata

Name	Data Connection Type	Data Connection Instance
Ranger	RDS MySQL database	<input type="text"/>

[Create Data Connection](#)

The user configured in the data connection used by Ranger must be the root user or authorized user.

**NOTE**

If the selected data connection is an **RDS MySQL database**, ensure that the database user is a **root** user. If the database user is not a **root** user, log in to the database as user **root** and run the following SQL statement to grant permissions to the database user. In the command, **`\${db\_name}`** and **`\${db\_user}`** indicate the database name and username entered during data connection creation.

```
grant select on mysql.user to `${db_user}`;
grant all privileges on `${db_name}`.* to `${db_user}`@'%' with grant option;
grant reload on *.* to `${db_user}`@'%' with grant option;
flush privileges;
```

**Step 4** Configure other parameters by referring to **Configuring a Cluster > Custom Creation of a Cluster** in the *User Guide*.

**NOTE**

- After the cluster is created, Ranger does not control users' permissions to access Hive and HBase.
- When Ranger is used to manage component permissions, for example, manage Hive table permissions, if a user submits a Hive job (operation on Hive data tables) on the interface or client, a message may be displayed indicating that the user does not have the permissions. In this case, you need to configure the database or table permissions for the user who submits the job in Ranger. For details, see the step for adding a policy in [Configuring Hive Access Permissions in Ranger](#) or [Configuring HBase Access Permissions in Ranger](#).

----End

## 21.2 Accessing the Ranger Web UI and Synchronizing Unix Users to the Ranger Web UI

You can manage Ranger on the Ranger web UI.



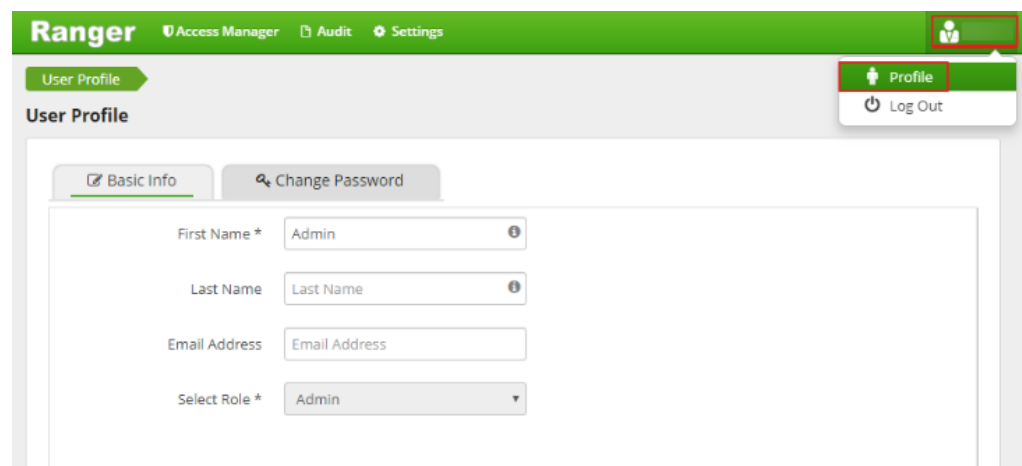
## Accessing the Ranger Admin Web UI

- Step 1** On the MRS management console, click the cluster name to go to the cluster details page.
- Step 2** Click the **Components** tab.
- Step 3** Select **Ranger**. In **Ranger Summary**, click **RangerAdmin** corresponding to **Ranger Web UI**.
- Step 4** On the Ranger web UI login page, the default username for MRS 1.9.2 is **admin** and the password is **admin@12345**. The default username for MRS 1.9.3 is **admin** and the password is **ranger@A1!**.

After logging in to the Ranger Web UI for the first time, change the password and keep it secure.

- Step 5** Click the username in the upper right corner, choose **Profile** from the drop-down list, and click **Change Password** to change the password.

**Figure 21-3** Changing the Ranger web UI login password



- Step 6** After changing the password, click the username in the upper right corner, choose **Log Out** from the drop-down list, and use the new password to log in to the web UI again.

----End

## Using Ranger UserSync to Synchronize Unix OS Users on Cluster Nodes

Ranger UserSync is an important component of Ranger. It can synchronize Unix system users or LDAP users to the Ranger web UI. Currently, MRS can synchronize only Unix users on the node where the Ranger UserSync process resides.

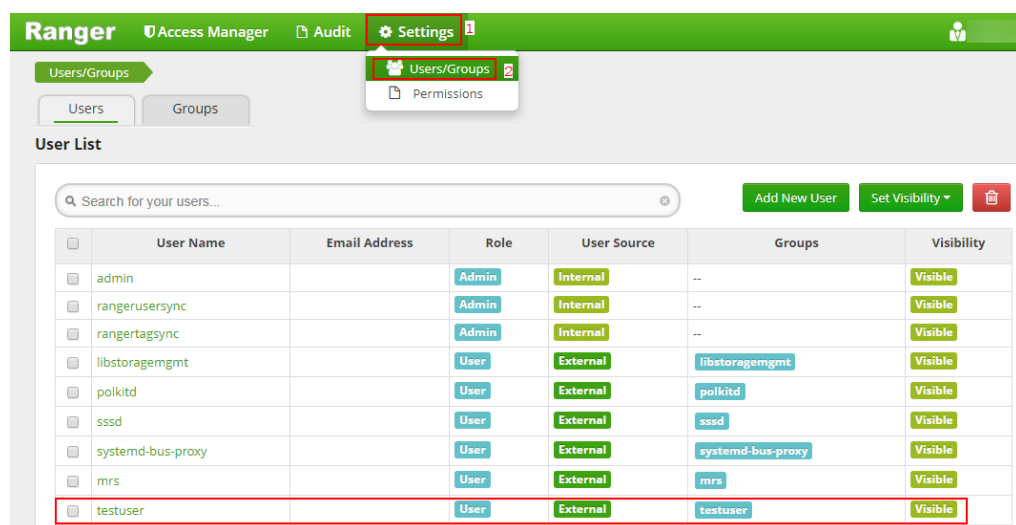
- Step 1** Log in to the node where the Ranger UserSync process is located.
- Step 2** Run the **useradd** command to add a system user, for example, **testuser**.

**Figure 21-4** Adding the `testuser` system user

```
[root@node-master1aHRf ~]# useradd testuser
[root@node-master1aHRf ~]# passwd testuser
Changing password for user testuser.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

**Step 3** After the user is added, wait for about 1 minute and log in to the Ranger web UI. Then, you can see that the user is synchronized.

**Figure 21-5** User synchronization completed



----End

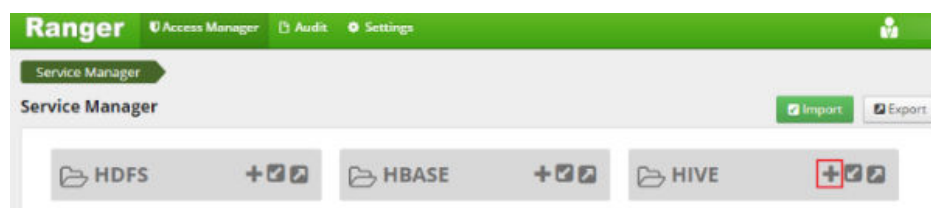
## 21.3 Configuring Hive Access Permissions in Ranger

After an MRS cluster with Ranger installed is created, Hive access control is not integrated into Ranger. This section describes how to integrate Hive into Ranger.

**Step 1** Log in to the Ranger web UI.

**Step 2** In the **Service Manager** area, click **+** next to **HIVE** to add a Hive service.

**Figure 21-6** Adding a Hive service



**Step 3** Set the parameters for adding a Hive service according to [Table 21-1](#). Use the default values for the parameters that are not listed in the table.

**Table 21-1 Parameter description**

Parameter	Description	Example Value
Service Name	Name of the service to be created. The value is fixed to <b>hivedev</b> .	hivedev
Username	You can set this parameter to any value.	admin
Password	You can set this parameter to any value.	-
jdbc.driverClassName	Driver class for connecting to Hive. The value is fixed to <b>org.apache.hive.jdbc.HiveDriver</b> .	org.apache.hive.jdbc.HiveDriver
jdbc.url	<p>URL for connecting to Hive. The format is ZooKeeper mode:  jdbcTemplate://&lt;host&gt;:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2</p> <p>&lt;host&gt; indicates a ZooKeeper address. To obtain the ZooKeeper address, log in to MRS Manager, choose <b>Services &gt; ZooKeeper &gt; Instance</b>, and view the management IP address of the ZooKeeper instance.</p>	jdbcTemplate://xx.xx.xx.xx:2181,xx.xx.xx.xx:2181,xx.xx.xx.xx:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2

**Figure 21-7** Creating hivedev

The screenshot shows the 'Create Service' page in the Service Manager. It is divided into two main sections: 'Service Details' and 'Config Properties'.

**Service Details:**

- Service Name \*:
- Description:
- Active Status:  Enabled  Disabled
- Select Tag Service:

**Config Properties:**

- Username \*:
- Password \*:
- jdbc.driverClassName \*:
- jdbc.url \*:
- Common Name for Certificate:

**Add New Configurations:**

Name	Value
<input type="text"/>	<input type="text"/>

Buttons: Test Connection, Add, Cancel

**Step 4** Click **Add** to add the service.

**Step 5** Start the Ranger Hive plugin to authorize Ranger to manage Hive.

1. On the MRS management console, click the cluster name to go to the cluster details page.
2. Click the **Components** tab.
3. Choose **Hive > Service Configuration** and switch **Basic** to **All**.
4. Search for **hive.security.authorization** and modify the following configurations:
  - hive.security.authorization.enabled = true
  - hive.security.authorization.manager = org.apache.ranger.authorization.hive.authorizer.RangerHiveAuthorizerFactory
5. Click **Save Configuration** and select **Restart the affected services or instances** to restart the Hive service.

**Step 6** Add an access control policy.

1. Log in to the Ranger web UI.

2. In the **HIVE** area, click the added service **hivedev**.
3. Click **Add New Policy** to add an access control policy.
4. Set the parameters according to **Table 21-2**. Use the default values for the parameters that are not listed in the table.

**Table 21-2** Parameter description

Parameter	Description	Example Value
Policy Name	Policy name	Policy001
database	Name of the database that the policy allows to access	test
table	Name of the table corresponding to the database that the policy allows to access	table1
Hive Column	Column name of the table corresponding to the database that the policy allows to access	name
Allow Conditions	<ul style="list-style-type: none"> <li>- <b>Select Group:</b> user group that the policy allows to access</li> <li>- <b>Select User:</b> user in the user group that the policy allows to access</li> <li>- <b>Permissions:</b> permissions that the policy allows the user to have</li> </ul>	<ul style="list-style-type: none"> <li>- Select Group: <b>testuser</b></li> <li>- Select User: <b>testuser</b></li> <li>- Permissions: <b>Create and Select</b></li> </ul>

**Figure 21-8** Adding an access control policy for **hivedev**

The screenshot shows the 'Create Policy' interface in Ranger. The breadcrumb navigation is 'Service Manager > hivedev Policies > Create Policy'. The 'Policy Details' section includes:

- Policy Type: Access
- Policy Name: Policy001 (enabled)
- database: test (include)
- table: table1 (include)
- Hive Column: name (include)
- Audit Logging: YES
- Description: (empty)
- Policy Label: Policy Label

The 'Allow Conditions' section contains a table with the following data:

Select Group	Select User	Permissions	Delegate Admin
testuser	testuser	Create select	<input type="checkbox"/>

5. Click **Add** to add the policy. According to the preceding policy, user **testuser** in the **testuser** user group has the **Create** and **Select** permissions on the **name** column of **table1** in the **test** database of Hive, but no permissions to access other columns.

**Step 7** Log in to the Hive client by referring to [Using Hive from Scratch](#), and check whether Hive has been integrated into Ranger.

1. Run the following command to access the Hive beeline:  
**source /opt/client/bigdata\_env**  
**beeline**
2. Run the following command to set up a connection and log in as user **testuser**:  
**!connect jdbc:hive2://xx.xx.xx.xx:2181,xx.xx.3.81:2181,192.168.3.153:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2**

Figure 21-9 Logging in to Hive

```
[root@node-master1aHRF ~]# source /opt/client/bigdata_env
[root@node-master1aHRF ~]# beeline
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/share/log4j-slf4j-impl-2.6.2/log4j-slf4j-impl-2.6.2.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/share/slf4j-log4j12-1.7.5/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Connecting to jdbc:hive2://192.168.3.88:2181,192.168.3.83:2181,192.168.3.244:2181/?serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2
Connected to: Apache Hive (version)
Driver: Hive JDBC (version)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version by Apache Hive
0: jdbc:hive2://192.168.3.88:2181,192.168.3.83:2181,192.168.3.244:2181/?serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2
Connecting to jdbc:hive2://192.168.3.88:2181,192.168.3.83:2181,192.168.3.244:2181/?serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2
Enter username for jdbc:hive2://192.168.3.88:2181,192.168.3.83:2181,192.168.3.244:2181/: testuser
Enter password for jdbc:hive2://192.168.3.88:2181,192.168.3.83:2181,192.168.3.244:2181/:
Connected to: Apache Hive (version)
Driver: Hive JDBC (version)
Transaction isolation: TRANSACTION_REPEATABLE_READ
1: jdbc:hive2://192.168.3.88:2181,192.168.3.83:2181,192.168.3.244:2181/?
```

3. Query data and check whether Ranger is integrated.

Figure 21-10 Verifying the integration of Ranger with Hive

```
1: jdbc:hive2://192.168.3.88:2181,192.168.3.83:2181,192.168.3.244:2181/?select * from table1;
Error: Error while compiling statement: FAILED: HiveAccessControlException Permission denied: user [testuser] does not have [SELECT] privilege on [test/table1/*] (state=42000,code=40000)
1: jdbc:hive2://192.168.3.88:2181,192.168.3.83:2181,192.168.3.244:2181/?select name from table1;
INFO : State: Compiling.
INFO : Compiling command(queryId=ommm_20191204095459_5713dad6-1fff-4a98-96f8-0da351c63df9): select name from table1
INFO : Semantic Analysis Completed
INFO : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name=name, type:string, comment:null)], properties:null)
INFO : EXPLAIN output for queryId ommm_20191204095459_5713dad6-1fff-4a98-96f8-0da351c63df9 : STAGE DEPENDENCIES:
Stage-0 is a root stage [FETCH]

STAGE PLANS:
Stage: Stage-0
Fetch Operator
limit: -1
Processor Tree:
TableScan
alias: table1
GatherStats: false
Select Operator
expressions: name (type: string)
outputColumnNames: _col0
ListSink

INFO : Completed compiling command(queryId=ommm_20191204095459_5713dad6-1fff-4a98-96f8-0da351c63df9); Time taken: 0.12 seconds
INFO : Concurrency mode is disabled, not creating a lock manager
INFO : State: Executing.
INFO : Executing command(queryId=ommm_20191204095459_5713dad6-1fff-4a98-96f8-0da351c63df9): select name from table1
INFO : Completed executing command(queryId=ommm_20191204095459_5713dad6-1fff-4a98-96f8-0da351c63df9); Time taken: 0.001 seconds
INFO : OK
+-----+
| name |
+-----+
+-----+
No rows selected (0.167 seconds)
```

----End

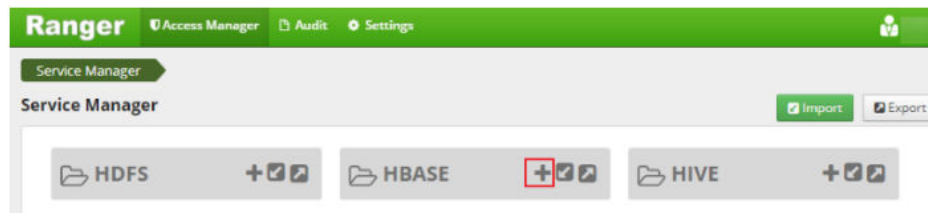
## 21.4 Configuring HBase Access Permissions in Ranger

After an MRS cluster with Ranger installed is created, HBase access control is not integrated into Ranger. This section describes how to integrate HBase into Ranger.

**Step 1** Log in to the Ranger web UI.

**Step 2** In the **Service Manager** area, click  next to **HBASE** to add an HBase service.

**Figure 21-11** Adding an HBase service



**Step 3** Set the parameters for adding an HBase service according to [Table 21-3](#). Use the default values for the parameters that are not listed in the table.

**Table 21-3** Parameter description

Parameter	Description	Example Value
Service Name	Name of the service to be created. The value is fixed to <b>hbasedev</b> .	hbasedev
Username	You can set this parameter to any value.	admin
Password	You can set this parameter to any value.	-
hadoop.security.authentication	Hadoop authentication mode. The value is fixed to <b>Simple</b> .	Simple
hbase.security.authentication	HBase authentication mode. The value is fixed to <b>Simple</b> .	Simple
hbase.zookeeper.property.clientPort	Port number of ZooKeeper in the HBase cluster.	2181
hbase.zookeeper.quorum	ZooKeeper address in the HBase cluster.	192.168.0.7,192.168.0.8,192.168.0.9
zookeeper.znode.parent	Path of the root node of HBase in ZooKeeper. The value is fixed to <b>/hbase</b> .	/hbase



Figure 21-12 Creating hbasedev

Service Manager > Create Service

### Create Service

**Service Details :**

Service Name \*

Description

Active Status  Enabled  Disabled

Select Tag Service

**Config Properties :**

Username \*

Password \*

hadoop.security.authentication \*

hbase.master.kerberos.principal

hbase.security.authentication \*

hbase.zookeeper.property.clientPort \*

hbase.zookeeper.quorum \*

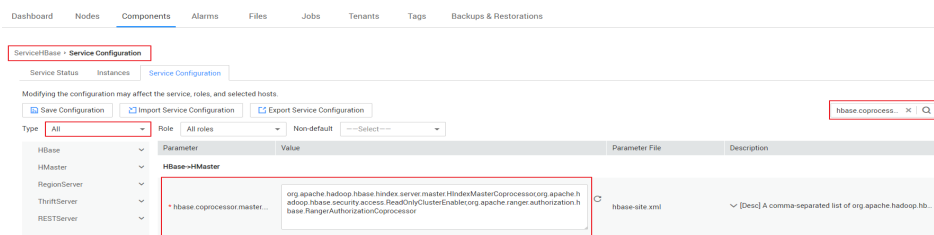
zookeeper.znode.parent \*

**Step 4** Click **Add** to add the service.

**Step 5** Start the Ranger HBase plugin to authorize Ranger to manage HBase.

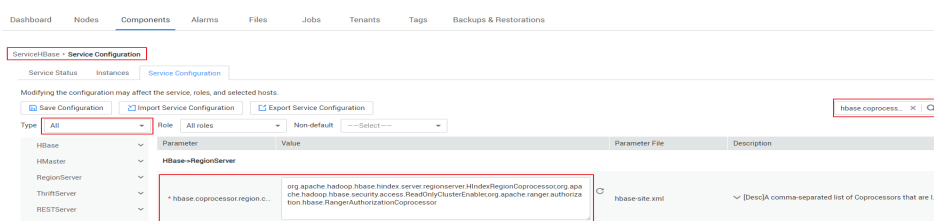
1. On the MRS management console, click the cluster name to go to the cluster details page.
2. Click the **Components** tab.
3. Choose **HBase > Service Configuration** and switch **Basic** to **All**.
4. Search for **hbase.security.authorization** and change its value to **true** (select the first HBase parameter).

**Figure 21-13** Modifying `hbase.security.authorization`



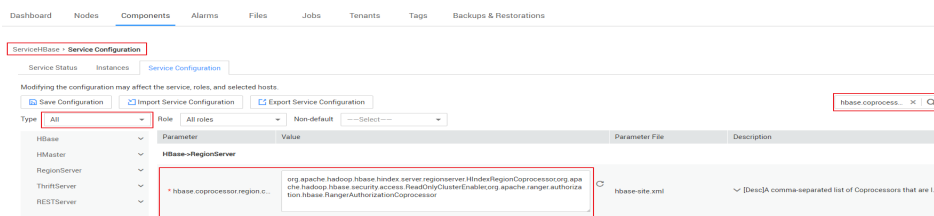
5. Search for `hbase.coprocessor.master.classes` and append `,org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor` to its original value.

**Figure 21-14** `hbase.coprocessor.master.classes`



6. Search for `hbase.coprocessor.region.classes` and append `,org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor` to its original value.

**Figure 21-15** `hbase.coprocessor.region.classes`



7. Click **Save Configuration** and select **Restart the affected services or instances** to restart the HMaster and RegionServer instances.

**Step 6** Create a policy under **HBase Service hbasedev**.

1. Log in to the Ranger web UI.
2. In the **HBASE** area, click the added service **hbasedev**.
3. Click **Add New Policy** to add an access control policy.
4. Set the parameters according to [Table 21-4](#). Use the default values for the parameters that are not listed in the table.

**Table 21-4** Parameter description

Parameter	Description	Example Value
Policy Name	Policy name	Policy002

Parameter	Description	Example Value
HBase Table	Name of the HBase table that the policy allows to access	test1
HBase Column-family	Column family of the HBase table that the policy allows to access	cf1
HBase Column	Column name of the table corresponding to the HBase table that the policy allows to access	name
Allow Conditions	<ul style="list-style-type: none"> <li>- <b>Select Group:</b> user group that the policy allows to access</li> <li>- <b>Select User:</b> user in the user group that the policy allows to access</li> <li>- <b>Permissions:</b> permissions that the policy allows the user to have</li> </ul>	<ul style="list-style-type: none"> <li>- Select Group: <b>testuser</b></li> <li>- Select User: <b>testuser</b></li> <li>- Permissions: <b>Create</b> and <b>Select</b></li> </ul>

**Figure 21-16** Adding an access control policy for **hbasedev**

The screenshot shows the 'Create Policy' page in the Service Manager. The breadcrumb trail is 'Service Manager > hbasedev Policies > Create Policy'. The page title is 'Create Policy'. Under 'Policy Details:', there are several fields: 'Policy Type' is 'Access'; 'Policy Name \*' is 'Policy002' with an 'enabled' toggle; 'HBase Table \*' is 'test1' with an 'include' toggle; 'HBase Column-family \*' is 'cf1' with an 'include' toggle; 'HBase Column \*' is 'name' with an 'include' toggle; 'Audit Logging' is 'YES' with a toggle; 'Description' is an empty text area; and 'Policy Label' is 'Policy Label'. Below this is the 'Allow Conditions:' section, which contains a table with columns: 'Select Group', 'Select User', 'Permissions', and 'Delegate Admin'. The 'testuser' group and user are selected in the first two columns. The 'Permissions' column shows 'Read' and 'Write' buttons. The 'Delegate Admin' column has a checkbox and a red 'X' button.

5. Click **Add** to add the policy. According to the preceding policy, user **testuser** in the **testuser** user group has the **Create** and **Select** permissions on the **cf1:name** column in the **test1** table of the **default** namespace in HBase, but no permissions to access other columns.

**Step 7** Update and log in to the HBase client by referring to [Using HBase from Scratch](#), and check whether HBase has been integrated into Ranger.

1. Run the following command to access the HBase shell:

```
source /opt/client/bigdata_env
hbase shell
```

**Figure 21-17** Accessing the HBase shell

```
[root@node-master1aHRf conf]# source /opt/client/bigdata_env
[root@node-master1aHRf conf]# hbase shell
INFO: Watching file:/opt/client/HBase/hbase/conf/log4j.properties for changes with interval : 60000
```

2. Add data and check whether Ranger is integrated.
  - a. Add data to the **cf1:name** column in the **test1** table.  
**put 'test1','001','cf1:name','tom'**
  - b. Add data to the **cf1:age** column in the **test1** table. If the user has no permission to access this column, the data fails to be added.  
**put 'test1','001','cf1:age',10**

**Figure 21-18** Verifying the integration of Ranger with HBase

```
hbase(main):037:0> put 'test1','001','cf1:name','Tom'
0 row(s) in 0.1120 seconds

hbase(main):038:0> scan 'test1'
ROW COLUMN+CELL
001 column=cf1:name, timestamp=1575426581007, value=Tom
1 row(s) in 0.0170 seconds

hbase(main):039:0> put 'test1','001','cf1:age','10'
2019-12-04 10:30:15,637 WARN [hconnection-0x52ff99cd-shared--pool1-t26] client.AsyncProcess: #3, table=test1, attempt=1/7 failed-ops, last exception: org.apache.hadoop.hbase.security.AccessDeniedException: org.apache.hadoop.hbase.security.AccessDeniedException: Insufficient permissions for user 'testuser',action: put, tableName:test1, family:cf1, column: age
 at org.apache.ranger.authorization.hbase.RangerAuthorizationCoprocessor.requirePermission(RangerAuthorizationCoprocessor.java:582)
```

----End

# 22 Using Ranger (MRS 3.x)

---

## 22.1 Logging In to the Ranger Web UI

Ranger provides a centralized permission management framework to implement fine-grained permission control on components such as HDFS, HBase, Hive, and Yarn. In addition, Ranger also provides a web UI to perform operations.

### Ranger User Type

Ranger users are classified into **admin**, **user**, and **auditor**. Different users have different permissions to view and operate the Ranger management interface.

- **Admin:** A security administrator can view all page content, manage permission management plug-ins and access control policies, view audit information, and set user types.
- **Auditor:** An audit administrator can view the permission management plug-ins and access control policies.
- **User:** A common user who can be assigned with specific permissions by the administrator.

### Logging In to the Ranger Web UI

Security mode (Kerberos authentication is enabled for clusters)

**Step 1** Log in to FusionInsight Manager as user **admin**. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster > Services > Ranger**. The Ranger service overview page is displayed.

**Step 2** Click **RangerAdmin** in the **Basic Information** area. The Ranger web UI is displayed.

- The **admin** user in Ranger belongs to the **User** type and can only view the **Access Manager** as well as **Security Zone** pages.
- To view all management pages, switch to user **rangeradmin** or other users who have the Ranger administrator permissions.
  - a. On the Ranger WebUI, click the user name in the upper right corner and choose **Log Out** to log out of the Ranger WebUI.



- b. Log in to the system as user **rangeradmin** (default password: **Rangeradmin@123**) or another user who has the Ranger administrator permissions.

----End

Normal mode (Kerberos authentication is disabled for clusters)

**Step 1** Log in to FusionInsight Manager as user **admin**. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster > Services > Ranger**. The Ranger service overview page is displayed.

**Step 2** Click **RangerAdmin** in the **Basic Information** area. The Ranger web UI is displayed.

The **admin** user in Ranger belongs to the **Admin** type and can view all management pages of Ranger without switching to user **rangeradmin**.

**NOTE**

When a user logs in to the Ranger WebUI as user **rangeradmin** in normal mode, error 401 is reported.

----End

On the homepage of Ranger web UI, you can view the permission management plug-ins of the services integrated in Ranger. The plug-ins can be used to set more fine-grained permissions. For details about functions of main operations you can perform on the page, see [Table 22-1](#).

**Table 22-1** Functions of each operation portal on the Ranger page

Portal	Function
Access Manager	You can view the permission management plug-ins of each service integrated in Ranger. The plug-ins can be used to set more fine-grained permissions. For details, see <a href="#">Configuring Component Permission Policies</a> .
Audit	You can view the audit logs related to Ranger running and permission control. For details, see <a href="#">Viewing Ranger Audit Information</a> .
Security Zone	MRS cluster administrators can divide resources of each component into multiple security zones where different administrators set security policies for specified resources of services to facilitate management. For details, see <a href="#">Configuring a Security Zone</a> .
Settings	You can view Ranger permission settings, such as users, user groups, and roles. For details, see <a href="#">Viewing Ranger Permission Information</a> .

## 22.2 Enabling Ranger Authentication

### Scenario

This section guides you how to enable Ranger authentication. Ranger authentication is enabled by default in security mode and disabled by default in normal mode.

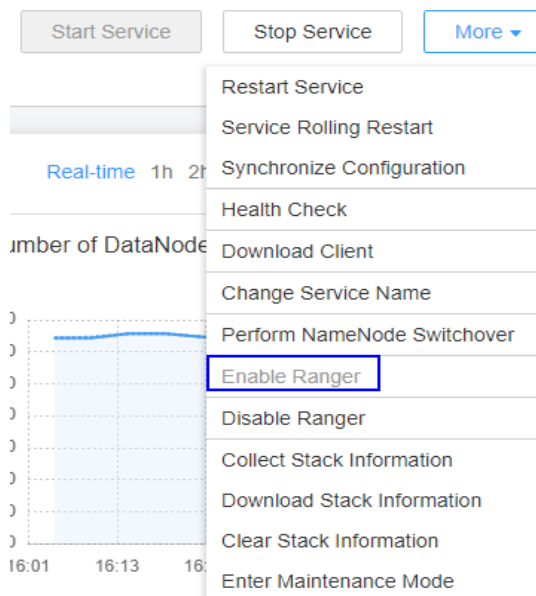
### Procedure

- Step 1** Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster > Services > Name of the service for which Ranger authentication is enabled**.
- Step 2** In the upper right corner of the **Dashboard** page, click **More** and select **Enable Ranger**. In the displayed dialog box, enter the password and click **OK**. After the operation is successful, click **Finish**.

 **NOTE**

If **Enable Ranger** is dimmed, Ranger authentication is enabled, as shown in [Figure 22-1](#).

**Figure 22-1** Enabling Ranger Authentication



- Step 3** Perform a rolling service restart or restart the service.

----End

## 22.3 Configuring Component Permission Policies

In the newly installed MRS cluster, Ranger is installed by default, with the Ranger authentication model enabled. You can set fine-grained security policies for accessing component resources through the component permission plug-ins.



Currently, the following components in a cluster in security mode support Ranger: HDFS, Yarn, HBase, Hive, Spark2x, Kafka, Storm..

## Configuring User Permission Policies Using Ranger

- Step 1** Log in to the Ranger management page.
- Step 2** In the **Service Manager** area on the Ranger homepage, click the permission plug-in name of a component. The page for security access policy list of the component is displayed.

### NOTE

In the policy list of each component, many items are generated by default to ensure the permissions of some default users or user groups (such as the **supergroup** user group). Do not delete these items. Otherwise, the permissions of the default users or user groups are affected.

- Step 3** Click **Add New Policy** and configure resource access policies for related users or user groups based on the service scenario plan.

The following policies are examples for different components:

- [Adding a Ranger Access Permission Policy for HDFS](#)
- [Adding a Ranger Access Permission Policy for HBase](#)
- [Adding a Ranger Access Permission Policy for Hive](#)
- [Adding a Ranger Access Permission Policy for Yarn](#)
- [Adding a Ranger Access Permission Policy for Spark2x](#)
- [Adding a Ranger Access Permission Policy for Kafka](#)
- [Adding a Ranger Access Permission Policy for Storm](#)

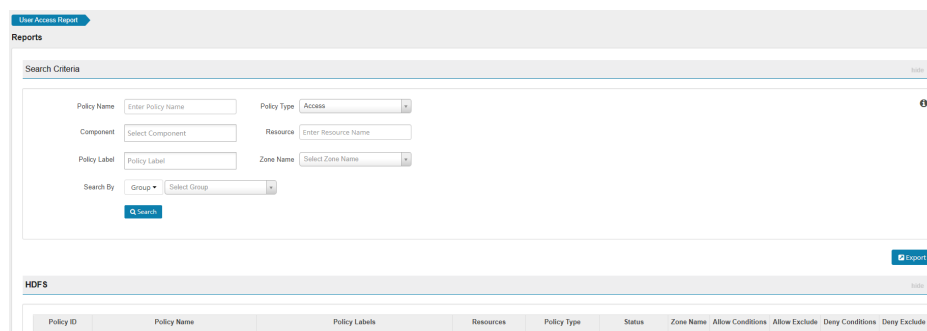
After the policies are added, wait for about 30 seconds for them to take effect.

### NOTE

Each time a component is started, the system checks whether the default Ranger service of the component exists. If the service does not exist, the system creates the Ranger service and adds a default policy for it. If a service is deleted by mistake, you can restart or restart the corresponding component service in rolling mode to restore the service. If the default policy is deleted by mistake, you can manually delete the service and then restart the component service.

- Step 4** Choose **Access Manager > Reports** to view all security access policies of each component.

If there are many system policies, filter and search for policies by the policy name, policy type, component, resource, policy label, security zone, user, or user group. Alternatively, click **Export** to export related policies.



**NOTE**

- Generally, only one policy can be configured for a fixed resource object. If multiple policies are configured for the same resource object, the policies cannot be saved.
- For details about the priorities of different policies, see [Condition Priorities of the Ranger Permission Policy](#).

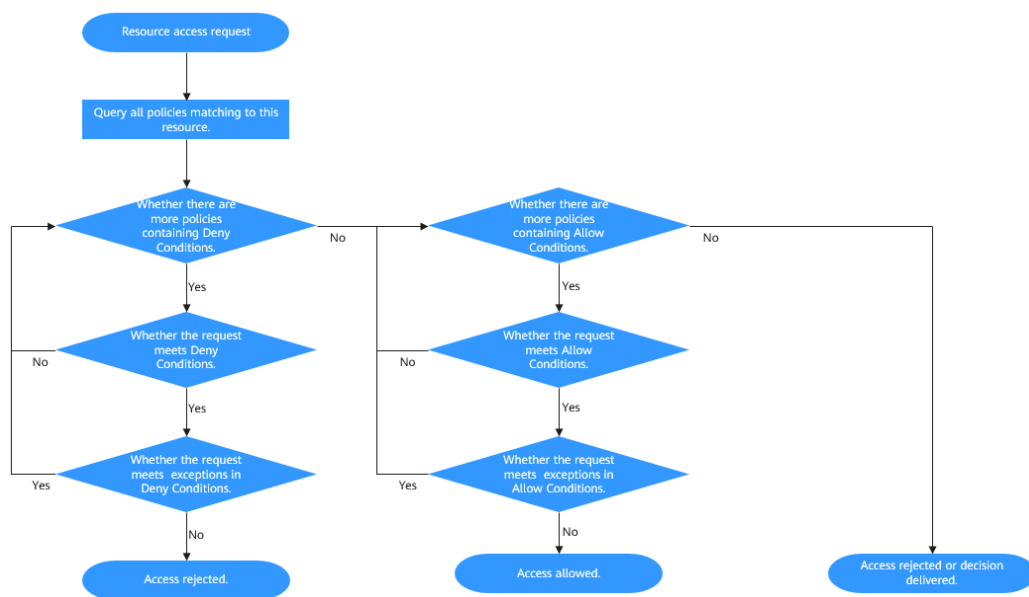
----End

## Condition Priorities of the Ranger Permission Policy

When configuring a permission policy for a resource, you can configure Allow Conditions, Exclude from Allow Conditions, Deny Conditions, and Exclude from Deny Conditions for the resource, to meet unexpected requirements in different scenarios.

The priorities of different conditions are listed in descending order: Exclude from Deny Conditions > Deny Conditions > Exclude from Allow Conditions > Allow Conditions

The following figure shows the process of determining condition priorities. If the component resource request does not match the permission policy in Ranger, the system rejects the access by default. However, for HDFS and Yarn, the system delivers the decision to the access control layer of the component for determination.



For example, if you want to grant the read and write permissions of the **FileA** folder to the **groupA** user group, but the user in the group is not **UserA**, you can add an allowed condition and an exception condition.

## 22.4 Viewing Ranger Audit Information

You can view audit logs of the Ranger running and the permission control after Ranger authentication is enabled on the Ranger web UI.

## Viewing Ranger Audit Information

- Step 1** Log in to the Ranger management page.
- Step 2** Click **Audit** to view the audit information. For details about the content on each tab page, see [Table 22-2](#). If there are a large number of items, click the search box and filter the items based on the keyword field.

**Table 22-2** Audit information


Tab	Description
Access	Records audit information about users' access to component resources through Ranger authentication.
Admin	Records operation audit information on Ranger, such as the creation, update, and deletion of security access policies, component permission policies, and roles.
Login Sessions	Records session audit information for users who have logged in to Ranger.
Plugins	Records permission policy information of components in Ranger.
Plugin Status	Records audit information about permission policies of each component node.
User Sync	Records synchronized audit information of LDAP and Ranger users.

----End

## 22.5 Configuring a Security Zone

Security zone can be configured using Ranger. You can divide resources of each component into multiple security zones where MRS cluster administrators set security policies for specified resources in the zones to facilitate management. Policies defined in a security zone apply only to resources in the zone. After service resources are allocated to the security zone, the access permission policies for the resources in the non-security zone do not take effect. The MRS cluster administrator of a security zone can set policies only in the security zone that the MRS cluster administrator belongs to.

### Adding a Security Zone

- Step 1** Log in to the Ranger management page as the Ranger administrator.
- Step 2** Click **Security Zone**. On the zone list page, click  to add a zone.

**Table 22-3** Parameters for configuring a security zone

Parameter	Description	Example Value
Zone Name	Security zone	test
Zone Description	Description of the security zone	-
Admin Users/ Admin Usergroups	Management users and user groups in a security zone. You can add and modify permission policies for related resources in the security zone. At least one user or user group must be configured.	zone_admin
Auditor Users/ Auditor Usergroups	Audit users or user groups to be added. You can view the resource permission policies in the security zone. At least one user or user group must be configured.	zone_user
Select Tag Services	Tag information of a service	-
Select Resource Services	Services and resources in a security zone. After selecting a service, you need to add specific resource objects in the <b>Resource</b> column, such as the file directories of the HDFS server, Yarn queues, Hive databases and tables, and HBase tables and columns.	/ testzone

For example, to create a security zone for the **/testzone** directory in HDFS, the configuration is as follows:

**Zone Details :**

---

Zone Name \*

Zone Description

**Zone Administration :**

---

Admin Users

Admin Usergroups

Auditor Users

Auditor Usergroups

**Services :**

---

Select Tag Services

Select Resource Services \*

Service Name	Service Type	Resource
hacluster	HDFS	<input type="text" value="path: /testzone"/> <input type="button" value="edit"/> <input type="button" value="delete"/> <input type="button" value="+"/>

**Step 3** Click **Save** and wait until the security zone is added successfully.

The Ranger administrator can view all security zones on the **Security Zone** page and click **Edit** to modify the attributes of a security zone. If resources do not need to be managed in a security zone, the Ranger administrator can click **Delete** to delete the security zone.

----End

## Configuring Permission Policies in a Security Zone

**Step 1** Log in to the Ranger management page as the administrator of a security zone.

**Step 2** Select a security zone from the **Security Zone** drop-down list in the upper right corner of the Ranger home page to switch to the permission view of the security zone.



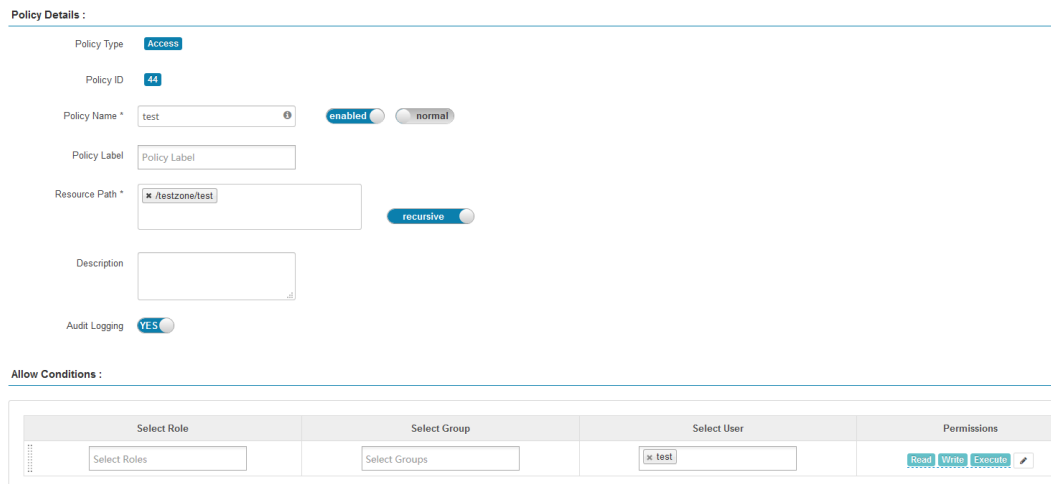
**Step 3** Click the permission plug-in name of a component. The page for security access policy list of the component is displayed.

 **NOTE**

In the policy list of each component, the default items generated by the system are automatically inherited to the security zone to ensure the permissions of some default users or user groups in the cluster.

**Step 4** Click **Add New Policy** and configure resource access policies for related users or user groups based on the service scenario plan.

In this example, a policy that allows user test to access the `/testzone/test` directory is configured in the security zone.



**Policy Details :**

Policy Type: **Access**

Policy ID: **44**

Policy Name \*  **enabled**  **normal**

Policy Label:

Resource Path \*  **recursive**

Description:

Audit Logging: **YES**

**Allow Conditions :**

Select Role	Select Group	Select User	Permissions
<input type="text" value="Select Roles"/>	<input type="text" value="Select Groups"/>	<input type="text" value="test"/>	<b>Read</b> <b>Write</b> <b>Execute</b>

The following access policies are examples for different components:

- [Adding a Ranger Access Permission Policy for HDFS](#)
- [Adding a Ranger Access Permission Policy for HBase](#)
- [Adding a Ranger Access Permission Policy for Hive](#)
- [Adding a Ranger Access Permission Policy for Yarn](#)
- [Adding a Ranger Access Permission Policy for Spark2x](#)
- [Adding a Ranger Access Permission Policy for Kafka](#)
- [Adding a Ranger Access Permission Policy for Storm](#)

After the policies are added, wait for about 30 seconds for them to take effect.

 **NOTE**

- Policies defined in a security zone apply only to resources in the zone. After service resources are allocated to the security zone, the access permission policies for the resources in the non-security zone do not take effect.
- To configure access policies for resources outside the current security zone, click **Security Zone** in the upper right corner of the Ranger homepage to exit the current security zone.

----End

## 22.6 Changing the Ranger Data Source to LDAP for a Normal Cluster

By default, the Ranger data source of the security cluster can be accessed by FusionInsight Manager LDAP users. By default, the Ranger data source of a common cluster can be accessed by Unix users.

### Prerequisites

- Check whether the cluster is in normal mode.
- The Ranger component has been installed.

### Procedure

- Step 1** Log in to the MRS management console.
- Step 2** Choose **Clusters > Active Clusters**, select a running cluster, and click its name to switch to the cluster details page.
- Step 3** Click the **Nodes** tab and select the node group whose **Node Type** is **Master**.
- Step 4** Go to the ECS page of the active Master node and click **Remote Login**.
- Step 5** Log in to the active Master node as user **root**, go to the **/opt/Bigdata/components/FusionInsight\_HD\_8.0.2.1/Ranger** directory, and change the value of **ranger.usersync.sync.source** in the **configurations.xml** file to **ldap**.

```
ranger.usersync.sync.source
<value model="NoSec">ldap</value>
```

- Step 6** Run the following commands on the active Master node to restart the controller process:

```
su - omm
```

```
sh /opt/Bigdata/om-server_8.0.2.1/om/sbin/restart-controller.sh
```

#### NOTE

When the controller process is restarted, the MRS Manager page cannot be accessed for a short period of time, which is normal. After the controller process is restarted, you can access the MRS Manager page properly.

- Step 7** Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster > Services > Ranger**. In the upper right corner of the **Dashboard** page, click **More** and choose **Synchronize Configuration**.
- Step 8** On the Ranger instance page, select the **UserSync** instance and choose **More > Restart Instance**.
- Step 9** On the **Dashboard** page of the Ranger service, click **RangerAdmin** and choose **Settings > Users/Groups/Roles** to check whether LDAP users exist.

----End

## 22.7 Viewing Ranger Permission Information

You can view Ranger permission settings, such as users, user groups, and roles.

### Viewing Ranger Permission Information

**Step 1** Log in to the Ranger management page.

**Step 2** Choose **Settings > Users/Groups/Roles** to view information about users, user groups, or roles in the system.

- **Users:** displays all user information synchronized from LDAP or OS to Ranger.
- **Groups:** displays information about all user groups and role information synchronized from LDAP or OS to Ranger.
- **Roles:** displays information about roles created in Ranger.

 **NOTE**

- The users, roles, user groups created on FusionInsight Manager are automatically synchronized to Ranger periodically. The default period is 300,000 milliseconds (5 minutes). After roles and user groups in FusionInsight Manager are synchronized to Ranger, they become user groups. Only roles and user groups that are associated with users can be automatically synchronized to Ranger.
- The role created on the Ranger page is a set of users or user groups, which is used to flexibly set the permission access policies of components. The role is different from that on FusionInsight Manager.

----End

### Adjusting Ranger User Types

**Step 1** Log in to the Ranger management page.

To change the Ranger user type, you must log in as an **admin** user. For details about the user types, see [Ranger User Type](#).

**Step 2** Choose **Settings > Users/Groups/Roles**. In the list of users, click the name of the user whose type you want to change.

**Step 3** Set **Select Role** to the type to be modified.

**Step 4** Click **Save**.

----End

### Creating a Ranger Role

You can flexibly configure permission access policies for components based on users, user groups, or roles. User and user group information is automatically synchronized from LDAP, and roles can be manually added.

**Step 1** Log in to the Ranger management page.

**Step 2** Choose **Settings > Users/Groups/Roles > Roles > Add New Role**.



**Step 3** Enter the role name and description as prompted.

**Step 4** Add users, user groups, and sub-roles to the role.

- In the **Users** area, select a created user in the system and click **Add Users**.
- In the **Groups** area, select a created user group and click **Add Group**.
- In the **Roles** area, select a created role in the system and click **Add Role**.

Users:

User Name	Is Role Admin	Action
test01	<input type="checkbox"/>	<input type="button" value="✖"/>

Select User

Groups:

Group Name	Is Role Admin	Action
hadoop	<input type="checkbox"/>	<input type="button" value="✖"/>

Select Group

Roles:

Role Name	Is Role Admin	Action
admin	<input type="checkbox"/>	<input type="button" value="✖"/>

Select Role

**Step 5** Click **Save**. The role is added successfully.

----End

## 22.8 Adding a Ranger Access Permission Policy for HDFS

### Scenario

You can use Ranger to configure the read, write, and execution permissions on HDFS directories or files for HDFS users.

### Prerequisites

- The Ranger service has been installed and is running properly.
- You have created users, user groups, or roles for which you want to configure permissions.

### Procedure

**Step 1** Log in to the Ranger management page.



**Step 2** On the homepage, click the component plug-in name in the **HDFS** area, for example, **hacluster**.

**Step 3** Click **Add New Policy** to add an HDFS permission control policy.

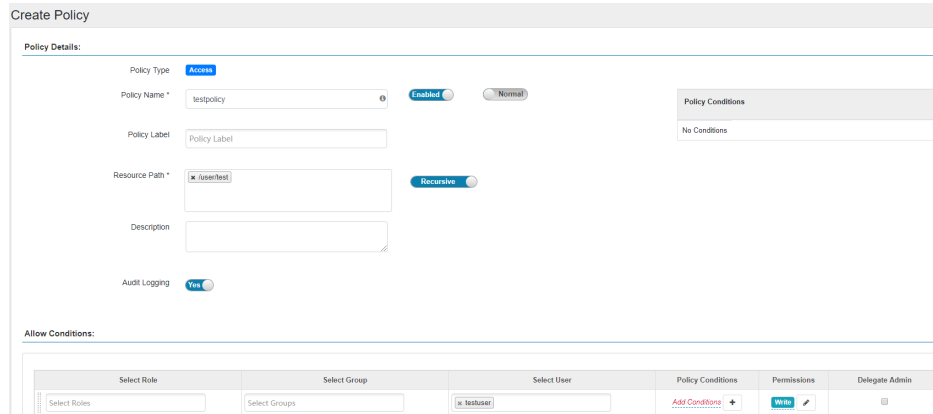
**Step 4** Configure the parameters listed in the table below based on the service demands.

**Table 22-4** HDFS permission parameters


Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
Resource Path	<p>Resource path, which is the HDFS path folder or file to which the current policy applies. You can enter multiple values and use the wildcard (*), for example, <b>/test/*</b>.</p> <p>To enable a subdirectory to inherit the permission of its upper-level directory, enable the recursion function.</p> <p>If recursion is enabled for the parent directory and a policy is configured for the subdirectory, the policy configured for the subdirectory is used.</p> <ul style="list-style-type: none"> <li>● <b>non-recursive</b>: recursion disabled</li> <li>● <b>recursive</b>: recursion enabled</li> </ul>
Description	Policy description.
Audit Logging	Whether to audit the policy.

Parameter	Description
Allow Conditions	<p>Permission and exception conditions allowed by a policy. The priority of an exception condition is higher than that of a normal condition.</p> <p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the role, user group, or user to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, and click <b>Add Permissions</b> to add the corresponding permission.</p> <ul style="list-style-type: none"> <li>• <b>Read</b>: permission to read data</li> <li>• <b>Write</b>: permission to write data</li> <li>• <b>Execute</b>: execution permission</li> <li>• <b>Select/Deselect All</b>: Select or deselect all.</li> </ul> <p>If users or user groups in the current condition need to manage this policy, select <b>Delegate Admin</b>. These users or user groups will become the agent administrators. The agent administrators can update and delete this policy and create sub-policies based on the original policy.</p> <p>To add multiple permission control rules, click . To delete a permission control rule, click .</p> <p><b>Exclude from Allow Conditions</b>: exception rules excluded from the allowed conditions</p>
Deny All Other Accesses	<p>Whether to reject all other access requests.</p> <ul style="list-style-type: none"> <li>• <b>True</b>: All other access requests are rejected.</li> <li>• <b>False</b>: <b>Deny Conditions</b> can be configured.</li> </ul>
Deny Conditions	<p>Policy rejection condition, which is used to configure the permissions and exceptions to be denied in the policy. The configuration method is the same as that of <b>Allow Conditions</b>. The priority of the rejection condition is higher than that of the allowed conditions configured in <b>Allow Conditions</b>.</p> <p><b>Exclude from Deny Conditions</b>: exception rules excluded from the denied conditions</p>

For example, to add the write permission for the **/user/test** directory of user **testuser**, the configuration is as follows:





**Table 22-5** Setting permissions


Task	Role Authorization
Setting the HDFS administrator permission	<ol style="list-style-type: none"> <li>1. On the homepage, click the component plug-in name in the <b>HDFS</b> area, for example, <b>hacluster</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - path</b> and click  to edit the policy.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> </ol>
Setting the permission for users to check and recover HDFS	<ol style="list-style-type: none"> <li>1. Add a folder or a file path in <b>Resource Path</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Read</b> and <b>Execute</b>.</li> </ol>
Setting the permission for users to read directories or files of other users	<ol style="list-style-type: none"> <li>1. Add a folder or a file path in <b>Resource Path</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Read</b> and <b>Execute</b>.</li> </ol>
Setting the permission for users to write data to files of other users	<ol style="list-style-type: none"> <li>1. Add a folder or a file path in <b>Resource Path</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Write</b> and <b>Execute</b>.</li> </ol>
Setting the permission for users to create or delete sub-files or sub-directories in the directory of other users	<ol style="list-style-type: none"> <li>1. Add a folder or a file path in <b>Resource Path</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Write</b> and <b>Execute</b>.</li> </ol>

Task	Role Authorization
Setting the permission for users to execute directories or files of other users	<ol style="list-style-type: none"> <li>1. Add a folder or a file path in <b>Resource Path</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Execute</b>.</li> </ol>
Setting the permission for allowing subdirectories to inherit all permissions of their parent directories	<ol style="list-style-type: none"> <li>1. Add a folder or a file path in <b>Resource Path</b>.</li> <li>2. Enable the recursion function. <b>Recursive</b> indicates that recursion is enabled.</li> </ol>

**Step 5** (Optional) Add the validity period of the policy. Click **Add Validity period** in the upper right corner of the page, set **Start Time** and **End Time**, and select **Time**

**Zone**. Click **Save**. To add multiple policy validity periods, click . To delete a policy validity period, click .

**Step 6** Click **Add** to view the basic information about the policy in the policy list. After the policy takes effect, check whether the related permissions are normal.

To disable a policy, click  to edit the policy and set the policy to **Disabled**.

If a policy is no longer used, click  to delete it.

----End

## 22.9 Adding a Ranger Access Permission Policy for HBase

### Scenario

You can use Ranger to configure permissions on HBase tables, column families, and columns for HBase users.

### Prerequisites

- The Ranger service has been installed and is running properly.
- You have created users, user groups, or roles for which you want to configure permissions.

### Procedure

**Step 1** Log in to the Ranger management page.



**Step 2** On the home page, click the component plug-in name in the **HBASE** area, for example, **HBase**.

**Step 3** Click **Add New Policy** to add an HBase permission control policy.


**Step 4** Configure the parameters listed in the table below based on the service demands.

**Table 22-6** HBase permission parameters

Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
HBase Table	Name of a table to which the policy applies. The value can contain wildcard (*). For example, <b>table1:*</b> indicates all tables in <b>table1</b> . The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object. <b>NOTE</b> The value of <b>hbase.rpc.protection</b> of the HBase service plug-in on Ranger must be the same as that of <b>hbase.rpc.protection</b> on the HBase server. For details, see <a href="#">When an HBase Policy Is Added or Modified on Ranger, Wildcard Characters Cannot Be Used to Search for Existing HBase Tables</a> .
HBase Column-family	Name of the column families to which the policy applies. The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
HBase Column	Name of the column to which the policy applies. The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
Description	Policy description.
Audit Logging	Whether to audit the policy.

Parameter	Description
Allow Conditions	<p>Policy allowed condition. You can configure permissions and exceptions allowed by the policy.</p> <p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the role, user group, or user to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, and click <b>Add Permissions</b> to add the corresponding permission.</p> <ul style="list-style-type: none"> <li>● <b>Read</b>: permission to read data</li> <li>● <b>Write</b>: permission to write data</li> <li>● <b>Create</b>: permission to create data</li> <li>● <b>Admin</b>: permission to manage data</li> <li>● <b>Select/Deselect All</b>: Select or deselect all.</li> </ul> <p>If users or user groups in the current condition need to manage this policy, select <b>Delegate Admin</b>. These users or user groups will become the agent administrators. The agent administrators can update and delete this policy and create sub-policies based on the original policy.</p> <p>To add multiple permission control rules, click . To delete a permission control rule, click .</p> <p>Exclude from Allow Conditions: policy exception conditions</p>
Deny All Other Accesses	<p>Whether to reject all other access requests.</p> <ul style="list-style-type: none"> <li>● <b>True</b>: All other access requests are rejected.</li> <li>● <b>False</b>: <b>Deny Conditions</b> can be configured.</li> </ul>
Deny Conditions	<p>Policy rejection condition, which is used to configure the permissions and exceptions to be denied in the policy. The configuration method is similar to that of <b>Allow Conditions</b>.</p> <p>The priority of <b>Deny Conditions</b> is higher than that of allowed conditions configured in <b>Allow Conditions</b>.</p> <p><b>Exclude from Deny Conditions</b>: exception rules excluded from the denied conditions</p>

**Table 22-7** Setting permissions



Task	Role Authorization
Setting the HBase administrator permission	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>HBase</b> area, for example, <b>HBase</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - table, column-family, column</b> and click  to edit the policy.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> </ol>
Setting the permission for users to create tables	<ol style="list-style-type: none"> <li>1. In <b>HBase Table</b>, specify a table name.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Create</b>.</li> <li>4. This user has the following permissions: create table drop table truncate table alter table enable table flush table flush region compact disable enable desc</li> </ol>
Setting the permission for users to write data to tables	<ol style="list-style-type: none"> <li>1. In <b>HBase Table</b>, specify a table name.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Write</b>.</li> <li>4. The user has the <b>put, delete, append, incr</b> and <b>bulkload</b> operation permissions.</li> </ol>
Setting the permission for users to read data from tables	<ol style="list-style-type: none"> <li>1. In <b>HBase Table</b>, specify a table name.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Read</b>.</li> <li>4. This user has the <b>get</b> and <b>scan</b> permissions.</li> </ol>




Task	Role Authorization
Setting the permission for users to manage namespaces or tables	<ol style="list-style-type: none"> <li>1. In <b>HBase Table</b>, specify a table name.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Admin</b>.</li> <li>4. The user has the <b>rsgroup</b>, <b>peer</b>, <b>assign</b> and <b>balance</b> operation permissions.</li> </ol>
Setting the permission for reading data from or writing data to columns	<ol style="list-style-type: none"> <li>1. In <b>HBase Table</b>, specify a table name.</li> <li>2. In <b>HBase Column-family</b>, specify the column family name.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Read</b> and <b>Write</b>.</li> </ol>

 **NOTE**

If a user performs the **desc** operation in **hbase shell**, the user must be granted the read permission on the **hbase:quota** table.

**Step 5** (Optional) Add the validity period of the policy. Click **Add Validity period** in the upper right corner of the page, set **Start Time** and **End Time**, and select **Time Zone**. Click **Save**. To add multiple policy validity periods, click . To delete a policy validity period, click .

**Step 6** Click **Add** to view the basic information about the policy in the policy list. After the policy takes effect, check whether the related permissions are normal.

To disable a policy, click  to edit the policy and set the policy to **Disabled**.

If a policy is no longer used, click  to delete it.

----End

## 22.10 Adding a Ranger Access Permission Policy for Hive

### Scenario

You can use Ranger to set permissions for Hive users. The default administrator account of Hive is **hive** and the initial password is **Hive@123**.

### Prerequisites

- The Ranger service has been installed and is running properly.


- You have created users, user groups, or roles for which you want to configure permissions.
- The users must be added to the **hive** group.

## Procedure

- Step 1** Log in to the Ranger management page.
- Step 2** On the home page, click the component plug-in name in the **HADOOP SQL** area, for example, **Hive**.
- Step 3** On the **Access** tab page, click **Add New Policy** to add a Hive permission control policy.
- Step 4** Configure the parameters listed in the table below based on the service demands.

**Table 22-8** Hive permission parameters

Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
database	Name of the Hive database to which the policy applies. The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
table	Name of the Hive table to which the policy applies. To add a UDF-based policy, switch to UDF and enter the UDF name. The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
Hive Column	Name of the column to which the policy applies. The value * indicates all columns. The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
Description	Policy description.
Audit Logging	Whether to audit the policy.

Parameter	Description
Allow Conditions	<p>Policy allowed condition. You can configure permissions and exceptions allowed by the policy.</p> <p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the role, user group, or user to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, and click <b>Add Permissions</b> to add the corresponding permission.</p> <ul style="list-style-type: none"> <li>● select: permission to query data</li> <li>● update: permission to update data</li> <li>● Create: permission to create data</li> <li>● Drop: permission to drop data</li> <li>● Alter: permission to alter data</li> <li>● Index: permission to index data</li> <li>● All: all permissions</li> <li>● Read: permission to read data</li> <li>● Write: permission to write data</li> <li>● Temporary UDF Admin: temporary UDF management permission</li> <li>● Select/Deselect All: Select or deselect all.</li> </ul> <p>To add multiple permission control rules, click .</p> <p>If users or user groups in the current condition need to manage this policy, select <b>Delegate Admin</b>. These users will become the agent administrators. The agent administrators can update and delete this policy and create sub-policies based on the original policy.</p>
Deny Conditions	<p>Policy rejection condition, which is used to configure the permissions and exceptions to be denied in the policy. The configuration method is similar to that of <b>Allow Conditions</b>.</p>

**Table 22-9** Setting permissions

Task	Role Authorization
<p><b>role admin</b> operation</p>	<ol style="list-style-type: none"> <li>1. On the home page, click <b>Settings</b> and choose <b>Roles</b>.</li> <li>2. Click the role with <b>Role Name</b> set to <b>admin</b>. In the <b>Users</b> area, click <b>Select User</b> and select a username.</li> <li>3. Click <b>Add Users</b>, select <b>Is Role Admin</b> in the row where the username is located, and click <b>Save</b>.</li> </ol> <p><b>NOTE</b> Only user <b>rangeradmin</b> has the permission to access the <b>Settings</b> option on the Ranger page. After being bound to the Hive administrator role, perform the following operations during each maintenance operation:</p> <ol style="list-style-type: none"> <li>1. Log in to the node where the Hive client is installed as the client installation user.</li> <li>2. Run the following command to configure environment variables: For example, if the Hive client installation directory is <b>/opt/hiveclient</b>, run <b>source /opt/hiveclient/bigdata_env</b>.</li> <li>3. Run the following command to authenticate the user: <b>kinit Hive service user</b></li> <li>4. Run the following command to log in to the client tool: <b>beeline</b></li> <li>5. Run the following command to update the administrator permissions: <b>set role admin;</b></li> </ol>
<p>Creating a database table</p>	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter or select the corresponding database on the right side of <b>database</b> and enter or select * on the right side of <b>column</b>. (To create a table, enter or select the corresponding table on the right side of <b>table</b>.)</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Create</b>.</li> </ol>
<p>Deleting a table</p>	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter or select the corresponding database on the right side of <b>database</b> and enter and select * on the right side of <b>column</b>. (To delete a table, enter or select the corresponding table on the right side of <b>table</b>.)</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Drop</b>.</li> </ol>


Task	Role Authorization
Query operation ( <b>select</b> , <b>desc</b> , and <b>show</b> )	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter or select the corresponding database on the right side of <b>database</b> and enter or select * (* indicates all columns) on the right side of <b>column</b>. (To create a table, enter or select the corresponding table on the right side of <b>table</b>.)</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>select</b>.</li> </ol>
<b>Alter</b> operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter and select the corresponding database on the right side of <b>database</b> and enter or select * on the right side of <b>column</b>. (For tables, enter or select the corresponding table on the right side of <b>table</b>.)</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Alter</b>.</li> </ol>
<b>LOAD</b> operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. On the right side of <b>database</b>, enter or select the corresponding database. On the right side of <b>table</b>, enter or select the corresponding table. On the right side of <b>column</b>, enter a column and select *.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>update</b>.</li> </ol>
<b>INSERT</b> and <b>DELETE</b> operations	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. On the right side of <b>database</b>, enter or select the corresponding database. On the right side of <b>table</b>, enter or select the corresponding table. On the right side of <b>column</b>, enter a column and select *.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>update</b>.</li> <li>5. Configure the <b>submit</b> permission on the Yarn task queue. For details about how to configure the permission, see <a href="#">Adding a Ranger Access Permission Policy for Yarn</a>.</li> </ol>


Task	Role Authorization
GRANT/REVOKE operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. On the right side of <b>database</b>, enter or select the corresponding database. On the right side of <b>table</b>, enter or select the corresponding table. On the right side of <b>column</b>, enter a column and select *.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Select <b>Delegate Admin</b>.</li> </ol>
ADD JAR operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Click <b>database</b>, and select <b>global</b> from the drop-down list. On the right of <b>global</b>, enter related information or select *.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Temporary UDF Admin</b>.</li> </ol>
UDF operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter or select the corresponding database on the right of <b>database</b>, and enter the corresponding udf function name on the right of <b>udf</b>.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select required permissions for the user (<b>udf</b> supports the <b>Create</b>, <b>select</b>, and <b>Drop</b> permissions).</li> </ol>
VIEW operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. On the right side of <b>database</b>, enter or select the corresponding database. On the right side of <b>table</b>, enter or select the corresponding table to be viewed. On the right side of <b>column</b>, enter a column and select *.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select permissions for the user as required.</li> </ol>
dfs command operation	<p>The <b>dfs</b> operation can be performed only after you have run the <b>set role admin</b> command.</p>
Operations on other user database tables	<ol style="list-style-type: none"> <li>1. Perform the preceding operations to add the corresponding permissions.</li> <li>2. Grant the read, write, and execution permissions on the HDFS paths of other user database tables to the user. For details, see <a href="#">Adding a Ranger Access Permission Policy for HDFS</a>.</li> </ol>

 NOTE

- If you have specified an HDFS path when running commands, you need to be granted with the read, write, and execution permissions on the HDFS paths. For details, see [Adding a Ranger Access Permission Policy for HDFS](#). You do not need to configure the Ranger policy of HDFS. You can use the Hive permission plug-in to add permissions to the role and assign the role to the corresponding user. If the HDFS Ranger policy can match the file or directory permission of the Hive database table, the HDFS Ranger policy is preferentially used.
- The URL policy in the Ranger policy is involved in the scenario where the Hive table is stored on OBS. Set the URL to the complete path of the object on OBS. The Read and Write permissions are used together with the URL. URL policies are not involved in other scenarios.
- The global policy in the Ranger policy is used only with the **Temporary UDF Admin** permission to control the upload of UDF packages.
- The **hiveservice** policy in the Ranger policy is used only with the **Service Admin** permission to control the permission to run the **kill query <queryId>** command to end the task that is being executed.
- The **lock**, **index**, **refresh**, and **replAdmin** permissions are not supported.
- Run the **show grant** command to view the table permission. The **grantor** column of the table **owner** is displayed as user **hive**. If the Ranger page is used or the **grant** command is used to grant permissions in the background, the **grantor** column is displayed as the corresponding user. To view the result of using the Hive permission plug-in, set **hive-ext.ranger.previous.privileges.enable** to **true** and run the **show grant** command.

**Step 5** Click **Add** to view the basic information about the policy in the policy list. After the policy takes effect, check whether the related permissions are normal.

To disable a policy, click  to edit the policy and set the policy to **Disabled**.

If a policy is no longer used, click  to delete it.

----End

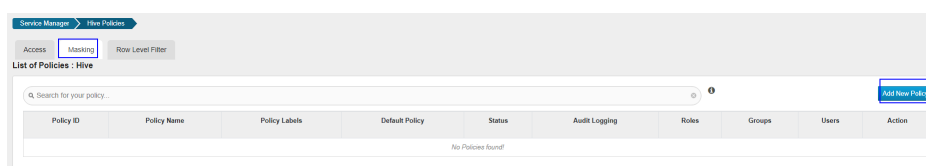
## Hive Data Masking

Ranger supports data masking for Hive data. It can process the returned result of the **select** operation you performed to mask sensitive information.

**Step 1** Log in to the Ranger web UI. Click **Hive** in the **HADOOP SQL** area on the homepage.



**Step 2** On the **Masking** tab page, click **Add New Policy** to add a Hive permission control policy.




**Step 3** Configure the parameters listed in the table below based on the service demands.

**Table 22-10** Hive data masking parameters

Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
Hive Database	Name of the Hive database to which the current policy applies.
Hive Table	Name of the Hive table to which the current policy applies.
Hive Column	Column name.
Description	Policy description.
Audit Logging	Whether to audit the policy.



Parameter	Description
Mask Conditions	<p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the object to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, then click <b>Add Permissions</b>, and select <b>select</b>. Click <b>Select Masking Option</b> and select a data masking policy.</p> <ul style="list-style-type: none"> <li>• Redact: Use <b>x</b> to mask all letters and <b>n</b> to mask all digits.</li> <li>• Partial mask: show last 4: Only the last four characters are displayed, and the rest characters are displayed using <b>x</b>.</li> <li>• Partial mask: show first 4: Only the first four characters are displayed, and the rest characters are displayed using <b>x</b>.</li> <li>• Hash: Replace the original value with the hash value. The Hive built-in function <b>mask_hash</b> is used. This is valid only for fields of the string, character, and varchar types. NULL is returned for fields of other types.</li> <li>• Nullify: Replace the original value with the NULL value.</li> <li>• Unmasked (retain original value): Keep the original value.</li> <li>• Date: show only year: Only the year part of the date string is displayed, and the default month and date start from January and Monday (<b>01/01</b>).</li> <li>• Custom: You customize policies using any valid return data type which is the same as the data type in the masked column.</li> </ul> <p>To add a multi-column masking policy, click .</p>

**Step 4** Click **Add** to view the basic information about the policy in the policy list.

**Step 5** After you perform the **select** operation on a table configured with a data masking policy on the Hive client, the system processes and displays the data.

 **NOTE**

To process data, you must have the permission to submit tasks to the Yarn queue.

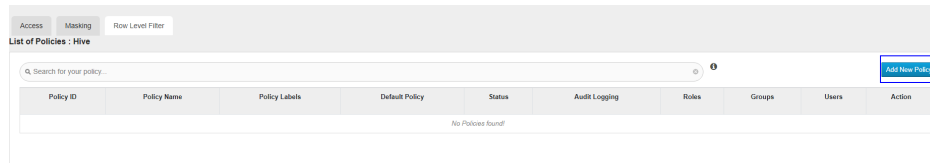
----End

## Hive Row-Level Data Filtering

Ranger allows you to filter data at the row level when you perform the **select** operation on Hive data tables.


**Step 1** Log in to the Ranger web UI. Click **Hive** in the **HADOOP SQL** area on the homepage.

**Step 2** On the **Row Level Filter** tab page, click **Add New Policy** to add a row data filtering policy.



**Step 3** Configure the parameters listed in the table below based on the service demands.

**Table 22-11** Parameters for filtering Hive row data

Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
Hive Database	Name of the Hive database to which the current policy applies.
Hive Table	Name of the Hive table to which the current policy applies.
Description	Policy description.
Audit Logging	Whether to audit the policy.
Row Filter Conditions	<p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the object to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, then click <b>Add Permissions</b>, and select <b>Select</b>. Click <b>Row Level Filter</b> and enter data filtering rules.</p> <p>For example, if you want to filter the data in the <b>zhangsan</b> row in the <b>name</b> column of <b>table A</b>, the filtering rule is <b>name &lt;&gt;'zhangsan'</b>. For more information, see the official Ranger document.</p> <p>To add more rules, click .</p>

**Step 4** Click **Add** to view the basic information about the policy in the policy list.

**Step 5** After you perform the **select** operation on a table configured with a data masking policy on the Hive client, the system processes and displays the data.

 **NOTE**

To process data, you must have the permission to submit tasks to the Yarn queue.

----**End**

## 22.11 Adding a Ranger Access Permission Policy for Yarn

### Scenario

You can use Ranger to configure Yarn administrator permissions for Yarn users, allowing them to manage Yarn queue resources.

### Prerequisites



- The Ranger service has been installed and is running properly.
- You have created users, user groups, or roles for which you want to configure permissions.

### Procedure


- Step 1** Log in to the Ranger management page.
- Step 2** On the home page, click the component plug-in name in the **YARN** area, for example, **Yarn**.
- Step 3** Click **Add New Policy** to add a Yarn permission control policy.
- Step 4** Configure the parameters listed in the table below based on the service demands.

**Table 22-12** Yarn permission parameters



Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
Queue	Queue name. The wildcard (*) is supported. To enable a sub-queue to inherit the permission of its upper-level queue, enable the recursion function. <ul style="list-style-type: none"> <li>• <b>Non-recursive:</b> recursion disabled</li> <li>• <b>Recursive:</b> recursion enabled</li> </ul>
Description	Policy description.
Audit Logging	Whether to audit the policy.

Parameter	Description
Allow Conditions	<p>Policy allowed condition. You can configure permissions and exceptions allowed by the policy.</p> <p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the role, user group, or user to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, and click <b>Add Permissions</b> to add the corresponding permission.</p> <ul style="list-style-type: none"> <li>• submit-app: permission to submit queue tasks</li> <li>• admin-queue: permission to manage queue tasks</li> <li>• Select/Deselect All: Select or deselect all.</li> </ul> <p>If users or user groups in the current condition need to manage this policy, select <b>Delegate Admin</b>. These users will become the agent administrators. The agent administrators can update and delete this policy and create sub-policies based on the original policy.</p> <p>To add multiple permission control rules, click . To delete a permission control rule, click .</p> <p>Exclude from Allow Conditions: policy exception conditions</p>
Deny All Other Accesses	<p>Whether to reject all other access requests.</p> <ul style="list-style-type: none"> <li>• True: All other access requests are rejected.</li> <li>• <b>False: Deny Conditions</b> can be configured.</li> </ul>
Deny Conditions	<p>Policy rejection condition, which is used to configure the permissions and exceptions to be denied in the policy. The configuration method is similar to that of <b>Allow Conditions</b>. The priority of <b>Deny Conditions</b> is higher than that of allowed conditions configured in <b>Allow Conditions</b>.</p> <p>Exclude from Deny Conditions: exception rules excluded from the denied conditions</p>


**Table 22-13** Setting permissions

Task	Role Authorization
Setting the Yarn administrator permission	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>YARN</b> area, for example, <b>Yarn</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - queue</b> and click  to edit the policy.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> </ol>

Task	Role Authorization
Setting the permission for a user to submit tasks in a specified Yarn queue	<ol style="list-style-type: none"> <li>1. In <b>Queue</b>, specify a queue name.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>submit-app</b>.</li> </ol>
Setting the permission for a user to manage tasks in a specified Yarn queue	<ol style="list-style-type: none"> <li>1. In <b>Queue</b>, specify a queue name.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>admin-queue</b>.</li> </ol>

**Step 5** (Optional) Add the validity period of the policy. Click **Add Validity period** in the upper right corner of the page, set **Start Time** and **End Time**, and select **Time Zone**. Click **Save**. To add multiple policy validity periods, click . To delete a policy validity period, click .

**Step 6** Click **Add** to view the basic information about the policy in the policy list. After the policy takes effect, check whether the related permissions are normal.

To disable a policy, click  to edit the policy and set the policy to **Disabled**.

If a policy is no longer used, click  to delete it.

----End

#### NOTE

The permissions on Ranger Yarn are independent of each other. There is inclusion relationship among the permissions. Currently, the following permissions are supported:

- **submit-app**: permission to submit queue tasks
- **admin-queue**: permission to manage queue tasks

Although the **admin-queue** has the permission to submit tasks, it does not have the inclusion relationship with the **submit-app** permission.

## 22.12 Adding a Ranger Access Permission Policy for Spark2x

### Scenario

You can use Ranger to set permissions for Spark2x users.

**NOTE**

1. After Ranger authentication is enabled or disabled on Spark2x, you need to restart Spark2x.
2. Download the client again or manually update the client configuration file *Client installation directory/Spark2x/spark/conf/spark-defaults.conf*.  
 Enable Ranger: `spark.ranger.plugin.authorization.enable=true`  
 Disable Ranger: `spark.ranger.plugin.authorization.enable=false`
3. In Spark2x, spark-beeline (applications connected to JDBCServer) supports the Ranger IP address filtering policy (**Policy Conditions** in the Ranger permission policy), while spark-submit and spark-sql do not.

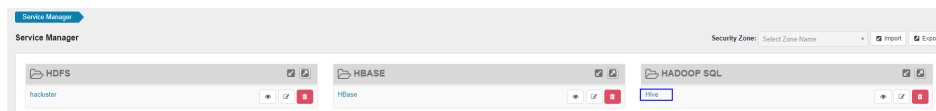
**Prerequisites**

- The Ranger service has been installed and is running properly.
- The Ranger authentication function of the Hive service has been enabled. After the Hive service is restarted, the Spark2x service is restarted.
- You have created users, user groups, or roles for which you want to configure permissions.
- The created user has been added to the **hive** user group.

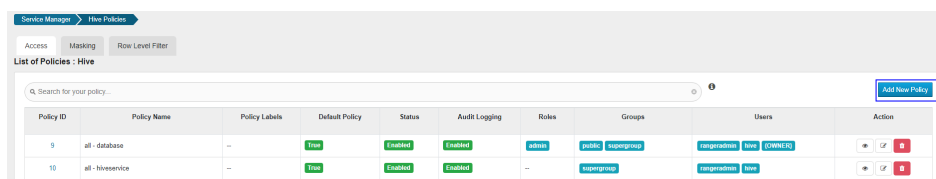
**Procedure**

**Step 1** Log in to the Ranger management page.

**Step 2** On the home page, click the component plug-in name in the **HADOOP SQL** area, for example, **Hive**.



**Step 3** On the **Access** tab page, click **Add New Policy** to add a Spark2x permission control policy.




**Step 4** Configure the parameters listed in the table below based on the service demands.

**Table 22-14** Spark2x permission parameters

Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.

Parameter	Description
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
database	Name of the Spark2x database to which the policy applies. The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
table	Name of the Spark2x table to which the policy applies. To add a UDF-based policy, switch to UDF and enter the UDF name. The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
column	Name of the column to which the policy applies. The value * indicates all columns. The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
Description	Policy description.
Audit Logging	Whether to audit the policy.

Parameter	Description
Allow Conditions	<p>Policy allowed condition. You can configure permissions and exceptions allowed by the policy.</p> <p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the role, user group, or user to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, and click <b>Add Permissions</b> to add the corresponding permission.</p> <ul style="list-style-type: none"> <li>● select: permission to query data</li> <li>● update: permission to update data</li> <li>● Create: permission to create data</li> <li>● Drop: permission to drop data</li> <li>● Alter: permission to alter data</li> <li>● Index: permission to index data</li> <li>● All: all permissions</li> <li>● Read: permission to read data</li> <li>● Write: permission to write data</li> <li>● Temporary UDF Admin: temporary UDF management permission</li> <li>● Select/Deselect All: Select or deselect all.</li> </ul> <p>To add multiple permission control rules, click  .</p> <p>If users or user groups in the current condition need to manage this policy, select <b>Delegate Admin</b>. These users will become the agent administrators. The agent administrators can update and delete this policy and create sub-policies based on the original policy.</p>
Deny Conditions	<p>Policy rejection condition, which is used to configure the permissions and exceptions to be denied in the policy. The configuration method is similar to that of <b>Allow Conditions</b>.</p>



**Table 22-15** Setting permissions

Task	Operation
<p><b>role admin</b> operation</p>	<ol style="list-style-type: none"> <li>1. On the home page, click <b>Settings</b> and choose <b>Roles &gt; Add New Role</b>.</li> <li>2. Set <b>Role Name</b> to <b>admin</b>. In the <b>Users</b> area, click <b>Select User</b> and select a username.</li> <li>3. Click <b>Add Users</b>, select <b>Is Role Admin</b> in the row where the username is located, and click <b>Save</b>.</li> </ol> <p><b>NOTE</b> After being bound to the Hive administrator role, perform the following operations during each maintenance operation:</p> <ol style="list-style-type: none"> <li>1. Log in to the node where the Hive client is installed as the client installation user.</li> <li>2. Run the following command to configure environment variables: For example, if the Spark2x client installation directory is <b>/opt/client</b>, run <b>source /opt/client/bigdata_env</b>.</li> <li>3. Run the following command to perform user authentication: <b>kinit Spark2xService user</b></li> <li>4. Run the following command to log in to the client tool: <b>spark-beeline</b></li> <li>5. Run the following command to update the administrator permissions: <b>set role admin;</b></li> </ol>
<p>Creating a database table</p>	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter and select the corresponding database on the right of <b>database</b>. (If you want to create a database, enter the name of the database to be created or enter <b>*</b> to indicate a database with any name, and then select the name.) Enter and select the corresponding table name on the right of <b>table</b> and <b>column</b>. Wildcard characters (*) are supported.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Create</b>.</li> </ol>

Task	Operation
Deleting a table	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter and select the corresponding database on the right of <b>database</b>. (If you want to delete a database, enter the name of the database to be created or enter * to indicate a database with any name, and then select the name.) Enter and select the corresponding table name on the right of <b>table</b> and <b>column</b>. Wildcard characters (*) are supported.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Drop</b>.</li> </ol> <p><b>NOTE</b> For CarbonData tables, only the owner of the corresponding database or table can perform the <b>drop</b> operation.</p>
<b>ALTER</b> operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter and select the corresponding database on the right of <b>database</b>, enter and select the corresponding table on the right of <b>table</b>, and enter and select the corresponding column name on the right of <b>column</b>. Wildcard characters (*) are supported.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Alter</b>.</li> </ol>
<b>LOAD</b> operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter and select the corresponding database on the right of <b>database</b>, enter and select the corresponding table on the right of <b>table</b>, and enter and select the corresponding column name on the right of <b>column</b>. Wildcard characters (*) are supported.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>update</b>.</li> </ol>

Task	Operation
INSERT operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter and select the corresponding database on the right of <b>database</b>, enter and select the corresponding table on the right of <b>table</b>, and enter and select the corresponding column name on the right of <b>column</b>. Wildcard characters (*) are supported.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>update</b>.</li> <li>5. The user also needs to have the <b>submit-app</b> permission of the Yarn task queue. By default, the Hadoop user group has the <b>submit-app</b> permission of all Yarn task queues. For details about how to load a network instance to a cloud connection, see <a href="#">Adding a Ranger Access Permission Policy for Yarn</a>.</li> </ol>
GRANT operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Enter and select the corresponding database on the right of <b>database</b>, enter and select the corresponding table on the right of <b>table</b>, and enter and select the corresponding column name on the right of <b>column</b>. Wildcard characters (*) are supported.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Select <b>Delegate Admin</b>.</li> </ol>
ADD JAR operation	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. Click <b>database</b>, and select <b>global</b> from the drop-down list. On the right of <b>global</b>, enter related information and select *.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Temporary UDF Admin</b>.</li> </ol>


Task	Operation
<b>VIEW</b> and <b>INDEX</b> permissions	<ol style="list-style-type: none"> <li>1. Enter the policy name in <b>Policy Name</b>.</li> <li>2. On the right side of <b>database</b>, enter the database name and select the corresponding database. (If you want to delete a database, enter the database name and select *.) On the right side of <b>table</b>, enter a table name and select the view and index names. On the right side of <b>column</b>, enter a Hive column name, and select *.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select permissions for the user as required.</li> </ol>
Operations on other user database tables	<ol style="list-style-type: none"> <li>1. Perform the preceding operations to add the corresponding permissions.</li> <li>2. Grant the read, write, and execution permissions on the HDFS paths of other user database tables to the current user. For details, see <a href="#">Adding a Ranger Access Permission Policy for HDFS</a>.</li> </ol>

 **NOTE**

After Spark SQL access policy is added on Ranger, you need to add the corresponding path access policies in the HDFS access policy. Otherwise, data files cannot be accessed. For details, see [Adding a Ranger Access Permission Policy for HDFS](#).

- The global policy in the Ranger policy is only used to associate with the **Temporary UDF Admin** permission to control the upload of UDF packages.
- When Ranger is used to control Spark SQL permissions, the **empower** syntax is not supported.

**Step 5** Click **Add** to view the basic information about the policy in the policy list. After the policy takes effect, check whether the related permissions are normal.

To disable a policy, click  to edit the policy and set the policy to **Disabled**.

If a policy is no longer used, click  to delete it.

----End

## Data Masking of the Spark2x Table

Ranger supports data masking for Spark2x data. It can process the returned result of the **select** operation you performed to mask sensitive information.


**Step 1** Log in to the Ranger WebUI and click the component plug-in name, for example, **Hive**, in the **HADOOP SQL** area on the home page.

**Step 2** On the **Masking** tab page, click **Add New Policy** to add a Spark2x permission control policy.

**Step 3** Configure the parameters listed in the table below based on the service demands.

**Table 22-16** Spark2x data masking parameters

Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
Hive Database	Name of the Spark2x database to which the current policy applies.
Hive Table	Name of the Spark2x table to which the current policy applies.
Hive Column	Name of the Spark2x column to which the current policy applies.
Description	Policy description.
Audit Logging	Whether to audit the policy.

Parameter	Description
Mask Conditions	<p>In the <b>Select Group</b> and <b>Select User</b> columns, select the user group or user to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, then click <b>Add Permissions</b>, and select <b>select</b>.</p> <p>Click <b>Select Masking Option</b> and select a data masking policy.</p> <ul style="list-style-type: none"> <li>● Redact: Use <b>x</b> to mask all letters and <b>n</b> to mask all digits.</li> <li>● Partial mask: show last 4: Only the last four characters are displayed.</li> <li>● Partial mask: show first 4: Only the first four characters are displayed.</li> <li>● Hash: Perform hash calculation for data.</li> <li>● Nullify: Replace the original value with the NULL value.</li> <li>● Unmasked(retain original value): The original data is displayed.</li> <li>● Date: show only year: Only the year information is displayed.</li> <li>● Custom: You can use any valid Hive UDF (returns the same data type as the data type in the masked column) to customize the policy.</li> </ul> <p>To add a multi-column masking policy, click .</p>
Deny Conditions	<p>Policy rejection condition, which is used to configure the permissions and exceptions to be denied in the policy. The configuration method is similar to that of <b>Allow Conditions</b>.</p>


----End

## Spark2x Row-Level Data Filtering

Ranger allows you to filter data at the row level when you perform the **select** operation on Spark2x data tables.

- Step 1** Log in to the Ranger WebUI and click the component plug-in name, for example, **Hive**, in the **HADOOP SQL** area on the home page.
- Step 2** On the **Row Level Filter** tab page, click **Add New Policy** to add a row data filtering policy.
- Step 3** Configure the parameters listed in the table below based on the service demands.

**Table 22-17** Parameters for filtering Spark2x row data

Parameter	Description
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
Hive Database	Name of the Spark2x database to which the current policy applies.
Hive Table	Name of the Spark2x table to which the current policy applies.
Description	Policy description.
Audit Logging	Whether to audit the policy.
Row Filter Conditions	<p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the object to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, then click <b>Add Permissions</b>, and select <b>select</b>. Click <b>Row Level Filter</b> and enter data filtering rules.</p> <p>For example, if you want to filter the data in the <b>zhangsan</b> row in the <b>name</b> column of <b>table A</b>, the filtering rule is <b>name &lt;&gt;'zhangsan'</b>. For more information, see the official Ranger document.</p> <p>To add more rules, click .</p>

**Step 4** Click **Add** to view the basic information about the policy in the policy list.

**Step 5** After you perform the **select** operation on a table configured with a data masking policy on the Spark2x client, the system processes and displays the data.

----End

## 22.13 Adding a Ranger Access Permission Policy for Kafka

### Scenario

You can use Ranger to configure the read, write, and management permissions of the Kafka topic and the management permission of the cluster for the Kafka user. This section describes how to add the production permission of the **test** topic for the **test** user.

## Prerequisites

- The Ranger service has been installed and is running properly.
- You have created users, user groups, or roles for which you want to configure permissions.


## Procedure

- Step 1** Log in to the Ranger management page.
- Step 2** On the home page, click the component plug-in name in the **KAFKA** area, for example, **Kafka**.
- Step 3** Click **Add New Policy** to add a Kafka permission control policy.
- Step 4** Configure the following parameters based on the service demands.

**Table 22-18** Kafka permission parameters

Parameter	Description
Policy Type	Access type.
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10</b> , <b>192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Name	Policy name, which can be customized and must be unique in the service.
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
topic	Name of the topic applicable to the current policy. You can enter multiple values. The value can contain wildcards, such as <b>test</b> , <b>test*</b> , and <b>*</b> .  The <b>Include</b> policy applies to the current input object, and the <b>Exclude</b> policy applies to objects other than the current input object.
Description	Policy description.
Audit Logging	Whether to audit the policy.



Parameter	Description
Allow Conditions	<p>Permission and exception conditions allowed by a policy. The priority of an exception condition is higher than that of a normal condition.</p> <p>In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the role, user group, or user to which you want to assign permissions.</p> <p>Click <b>Add Conditions</b>, add the IP address range to which the policy applies, and click <b>Add Permissions</b> to add corresponding permissions.</p> <ul style="list-style-type: none"> <li>● Publish: production permission</li> <li>● Consume: consumption permission</li> <li>● Describe: query permission</li> <li>● Create: topic creation permission</li> <li>● Delete: topic deletion permission</li> <li>● Describe Configs: configuration query permission</li> <li>● Alter: permission to change the number of partitions of a topic.</li> <li>● Alter Configs: configuration modification permission</li> <li>● Select/Deselect All: Select or deselect all.</li> </ul> <p>To add multiple permission control rules, click .</p> <p>If users or user groups in the current condition need to manage this policy, select <b>Delegate Admin</b>. These users will become the agent administrators. The agent administrators can update and delete this policy and create sub-policies based on the original policy.</p>
Deny Conditions	<p>Policy rejection condition, which is used to configure the permissions and exceptions to be denied in the policy. The configuration method is the same as that of <b>Allow Conditions</b>. The priority of the rejection condition is higher than that of the allowed conditions configured in <b>Allow Conditions</b>.</p>

For example, to add the production permission for the **test** topic of user **testuser**, configure the following information:

Figure 22-2 Kafka permission parameters

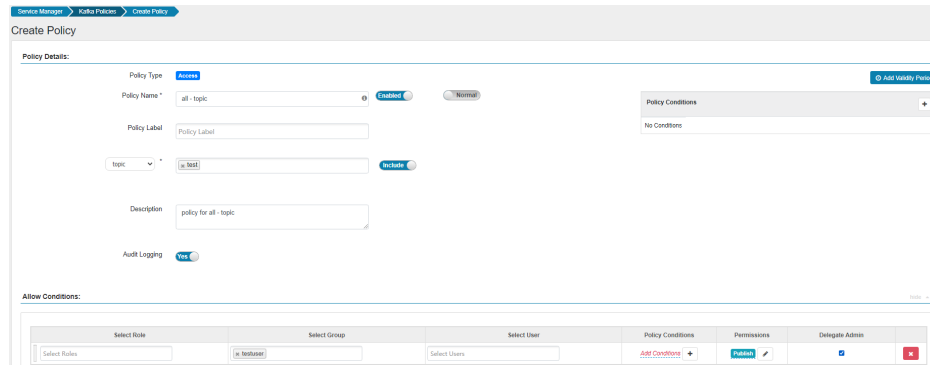





Table 22-19 Setting permissions




Scenario	Role Authorization
Setting the Kafka administrator permissions	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>KAFKA</b> area, for example, <b>Kafka</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - topic</b> and click  to edit the policy.</li> <li>3. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>4. Click <b>Add Permissions</b> and select <b>Select/Deselect All</b>.</li> </ol>
Setting the permission for a user to create a topic	<ol style="list-style-type: none"> <li>1. Specify a topic name in <b>topic</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Create</b>.</li> </ol> <p><b>NOTE</b> Currently, the Kafka kernel supports the <b>--zookeeper</b> and <b>--bootstrap-server</b> methods to create topics. The <b>--zookeeper</b> method will be deleted from the community in later versions. Therefore, you are advised to use the <b>--bootstrap-server</b> method to create topics.</p> <p>Note: Currently, Kafka supports only the authentication of topic creation in <b>--bootstrap-server</b> mode and does not support that in <b>--zookeeper</b> mode.</p>


Scenario	Role Authorization
Setting the permission for a user to delete a topic	<ol style="list-style-type: none"> <li>1. Specify a topic name in <b>topic</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Delete</b>.</li> </ol> <p><b>NOTE</b> Currently, the Kafka kernel supports the <b>--zookeeper</b> and <b>--bootstrap-server</b> methods to delete topics. The <b>--zookeeper</b> method will be deleted from the community in later versions. Therefore, you are advised to use the <b>--bootstrap-server</b> method to delete topics.</p> <p>Note: Currently, Kafka supports only the authentication of topic deletion in <b>--bootstrap-server</b> mode and does not support that in <b>--zookeeper</b> mode.</p>
Setting the permission for a user to query a topic	<ol style="list-style-type: none"> <li>1. Specify a topic name in <b>topic</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Describe</b> and <b>Describe Configs</b>.</li> </ol> <p><b>NOTE</b> Currently, the Kafka kernel supports the <b>--zookeeper</b> and <b>--bootstrap-server</b> methods to query topics. The <b>--zookeeper</b> method will be deleted from the community in later versions. Therefore, you are advised to use the <b>--bootstrap-server</b> method to query topics.</p> <p>Note: Currently, Kafka supports only the authentication of topic query in <b>--bootstrap-server</b> mode and does not support that in <b>--zookeeper</b> mode.</p>
Setting the production permission of a user on a topic	<ol style="list-style-type: none"> <li>1. Specify a topic name in <b>topic</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Publish</b>.</li> </ol>
Setting the consumption permission of a user on a topic	<ol style="list-style-type: none"> <li>1. Specify a topic name in <b>topic</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Consume</b>.</li> </ol> <p><b>NOTE</b> During topic consumption, offset management is involved. Therefore, the <b>Consume</b> permission of <b>ConsumerGroup</b> must be enabled at the same time. For details, see <b>Setting a User's Permission to Submit ConsumerGroup Offsets</b>.</p>
Setting the permission for a user to expand a topic (by adding partitions)	<ol style="list-style-type: none"> <li>1. Specify a topic name in <b>topic</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Alter</b>.</li> </ol>



Scenario	Role Authorization
Setting the permission for a user to modify the topic configuration	Currently, the Kafka kernel does not support to modify topic parameters based on <b>--bootstrap-server</b> . Therefore, Ranger does not support authentication for this behavior.
Setting all the management permissions of a user on a cluster	<ol style="list-style-type: none"> <li>1. Enter a cluster name and select the cluster on the right side of <b>cluster</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Kafka Admin</b>.</li> </ol>
Setting the permission for a user to create a cluster	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>KAFKA</b> area, for example, <b>Kafka</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - cluster</b> and click  to edit the policy.</li> <li>3. Enter a cluster name and select the cluster on the right side of <b>cluster</b>.</li> <li>4. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>5. Click <b>Add Permissions</b> and select <b>Create</b>.</li> </ol> <p><b>NOTE</b> The authentication of the <b>Create</b> operation of a cluster involves the following two scenarios:</p> <ol style="list-style-type: none"> <li>1. After the <b>auto.create.topics.enable</b> parameter is enabled in the cluster, the client sends data to a topic that has not been created in the service. In this case, the system checks whether the user has the <b>Create</b> permission of the cluster.</li> <li>2. If a user creates a large number of topics and is granted the <b>Cluster Create</b> permission, the user can create any topic in the cluster.</li> </ol>
Setting the permission for a user to modify the cluster configuration	<ol style="list-style-type: none"> <li>1. Enter a cluster name and select the cluster on the right side of <b>cluster</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Alter Configs</b>.</li> </ol> <p><b>NOTE</b> The configuration modification permission allows you to modify the Broker and Broker Logger configurations. After the configuration modification permission is granted to a user, the user can query configuration details even if the user does not have the query permission. (The configuration modification permission includes the configuration query permission.)</p>

Scenario	Role Authorization
<p>Setting the permission for a user to query the cluster configuration</p>	<ol style="list-style-type: none"> <li>1. Enter a cluster name and select the cluster on the right side of <b>cluster</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Describe</b> and <b>Describe Configs</b>.</li> </ol> <p><b>NOTE</b> You can only query Broker and Broker Logger information in the cluster, excluding topics.</p>
<p>Setting the Idempotent Write permission in a cluster for a user</p>	<ol style="list-style-type: none"> <li>1. Enter a cluster name and select the cluster on the right side of <b>cluster</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Idempotent Write</b>.</li> </ol> <p><b>NOTE</b> This permission authenticates the <b>Idempotent Produce</b> behavior of the user's client.</p>
<p>Setting the permission to migrate partitions in a cluster for a user</p>	<ol style="list-style-type: none"> <li>1. Enter a cluster name and select the cluster on the right side of <b>cluster</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Alter</b>.</li> </ol> <p><b>NOTE</b> The <b>Alter</b> permission of a cluster can be used to control permissions in the following scenarios:</p> <ol style="list-style-type: none"> <li>1. In the <b>Partition Reassign</b> scenario, migrate the storage directory of replicas.</li> <li>2. Elect a leader replica in each partition of the cluster.</li> <li>3. Add or delete ACLs.</li> </ol> <p>Operations in scenarios <a href="#">Step 4.1</a> and <a href="#">Step 4.2</a> are between a controller and broker and between brokers in the cluster. When a cluster is created, this permission is granted to the built-in Kafka user by default. It is meaningless for a common user to be granted with this permission.</p> <p>Scenario <a href="#">Step 4.3</a> involves the ACL management. ACLs are designed for authentication. Currently, Kafka authentication is hosted to Ranger. Therefore, this scenario is not involved (the configuration does not take effect).</p>


Scenario	Role Authorization
<p>Setting the Cluster Action permission in a cluster for a user</p>	<ol style="list-style-type: none"> <li>1. Enter a cluster name and select the cluster on the right side of <b>cluster</b>.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Cluster Action</b>.</li> </ol> <p><b>NOTE</b> This permission controls the synchronization between the leader and follower replicas in the cluster and the communication between nodes. It has been granted to the built-in Kafka user during cluster creation. It is meaningless for a common user to grant this permission.</p>
<p>Setting the TransactionalId permission for a user</p>	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>KAFKA</b> area, for example, <b>Kafka</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - transactionalid</b> and click  to edit the policy. <ol style="list-style-type: none"> <li>1. Set <b>transactionalid</b> to a transaction ID.</li> <li>2. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>3. Click <b>Add Permissions</b> and select <b>Publish</b> and <b>Describe</b>.</li> </ol> </li> </ol> <p><b>NOTE</b> The <b>Publish</b> permission is used to authenticate client requests for which the transaction feature is enabled, for example, starting and ending a transaction, submitting an offset, and generating transactional data. The <b>Describe</b> permission is used to authenticate the requests from the client and coordinator that have enabled the transaction feature. If the transaction feature is enabled, you are advised to grant both the <b>Publish</b> and <b>Describe</b> permissions to users.</p>

Scenario	Role Authorization
<p>Setting the DelegationToken permission for a user</p>	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>KAFKA</b> area, for example, <b>Kafka</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - delegationtoken</b> and click  to edit the policy.</li> <li>3. Set <b>delegationtoken</b> to a delegation token.</li> <li>4. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>5. Click <b>Add Permissions</b> and select <b>Describe</b>.</li> </ol> <p><b>NOTE</b> Currently, Ranger only controls the query permission of DelegationToken, but does not control its <b>create</b>, <b>renew</b>, and <b>expire</b> permissions.</p>
<p>Setting the permission for a user to query ConsumerGroup Offsets</p>	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>KAFKA</b> area, for example, <b>Kafka</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - consumergroup</b> and click  to edit the policy.</li> <li>3. In <b>consumergroup</b>, configure the consumer group to be managed.</li> <li>4. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>5. Click <b>Add Permissions</b> and select <b>Describe</b>.</li> </ol>
<p>Set the user's submission permission on <b>ConsumerGroup Offsets</b>.</p>	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>KAFKA</b> area, for example, <b>Kafka</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - consumergroup</b> and click  to edit the policy.</li> <li>3. In <b>consumergroup</b>, configure the consumer group to be managed.</li> <li>4. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>5. Click <b>Add Permissions</b> and select <b>Consume</b>.</li> </ol> <p><b>NOTE</b> After a user is granted with the <b>Consume</b> permission of <b>ConsumerGroup</b>, the user is also granted with the <b>Describe</b> permission.</p>

Scenario	Role Authorization
Setting the permission for a user to delete ConsumerGroup Offsets	<ol style="list-style-type: none"> <li>1. On the home page, click the component plug-in name in the <b>KAFKA</b> area, for example, <b>Kafka</b>.</li> <li>2. Select the policy whose <b>Policy Name</b> is <b>all - consumergroup</b> and click  to edit the policy.</li> <li>3. In <b>consumergroup</b>, configure the consumer group to be managed.</li> <li>4. In the <b>Allow Conditions</b> area, select a user from the <b>Select User</b> drop-down list.</li> <li>5. Click <b>Add Permissions</b> and select <b>Delete</b>.</li> </ol> <p><b>NOTE</b> When a user is granted with the <b>Delete</b> permission of <b>ConsumerGroup</b>, the user is also granted with the <b>Describe</b> permission.</p>

**Step 5** (Optional) Add the validity period of the policy. Click **Add Validity period** in the upper right corner of the page, set **Start Time** and **End Time**, and select **Time Zone**. Click **Save**. To add multiple policy validity periods, click . To delete a policy validity period, click .

**Step 6** Click **Add** to view the basic information about the policy in the policy list. After the policy takes effect, check whether the related permissions are normal.

To disable a policy, click  to edit the policy and set the policy to **Disabled**.

If a policy is no longer used, click  to delete it.

----End

## 22.14 Adding a Ranger Access Permission Policy for Storm

### Scenario

You can use Ranger to set permissions for Storm users.

### Prerequisites

- The Ranger service has been installed and is running properly.
- You have created users, user groups, or roles for which you want to configure permissions.
- The Ranger authentication function has been enabled on the page. The option in the following figure controls whether to enable the Ranger plug-in




for permission control. If the function is enabled, the Ranger authentication is used. Otherwise, the authentication mechanism of the component is used.



## Procedure

- Step 1** Log in to the Ranger web UI. Click **Storm** in the **STORM** area on the homepage.
- Step 2** Click **Add New Policy** to add a Storm permission control policy.
- Step 3** Configure the parameters listed in the table below based on the service demands.


**Table 22-20** Storm permission parameters


Parameter	Description
Policy Conditions	IP address filtering policy, which can be customized. You can enter one or more IP addresses or IP address segments. The IP address can contain the wildcard character (*), for example, <b>192.168.1.10,192.168.1.20</b> , or <b>192.168.1.*</b> .
Policy Name	Policy name, which can be customized and must be unique in the service. The <b>include</b> policy applies to the current input object, and the <b>exclude</b> policy applies to objects other than the current input object.
Policy Label	A label specified for the current policy. You can search for reports and filter policies based on labels.
Storm Topology	Name of the topology to which the current policy applies. One or more values can be entered.
Description	Policy description.
Audit Logging	Whether to audit the policy.

Parameter	Description
Allow Conditions	<p>Policy allowed condition. You can configure permissions and exceptions allowed by the policy. In the <b>Select Role</b>, <b>Select Group</b>, and <b>Select User</b> columns, select the role, user group, or user to which the permission is to be granted, click <b>Add Conditions</b>, add the IP address range to which the policy applies, and click <b>Add Permissions</b> to add the corresponding permissions.</p> <ul style="list-style-type: none"> <li>● <b>Submit Topology</b>: Submit a topology.</li> </ul> <p><b>NOTE</b> The Submit Topology permission takes effect only when <b>Storm Topology</b> is set to *.</p> <ul style="list-style-type: none"> <li>● <b>File Upload</b>: Upload a file.</li> <li>● <b>File Download</b>: Download a file.</li> <li>● <b>Kill Topology</b>: Delete a topology.</li> <li>● <b>Rebalance</b>: Perform the rebalance operation.</li> <li>● <b>Activate</b>: Activate the topology permission.</li> <li>● <b>Deactivate</b>: Deactivate the topology permission.</li> <li>● <b>Get Topology Conf</b>: Obtain topology configurations.</li> <li>● <b>Get Topology</b>: Obtain a topology.</li> <li>● <b>Get User Topology</b>: Obtain user's topology.</li> <li>● <b>Get Topology Info</b>: Obtain topology information.</li> <li>● <b>Upload New Credential</b>: Upload a new credential.</li> <li>● <b>Select/Deselect All</b>: Select or deselect all.</li> </ul> <p>To add multiple permission control rules, click .</p> <p>If users or user groups in the current condition need to manage this policy, select <b>Delegate Admin</b>. These users will become the agent administrators. The agent administrators can update and delete this policy and create sub-policies based on the original policy.</p>
Deny Conditions	<p>Policy rejection condition, which is used to configure the permissions and exceptions to be denied in the policy. The configuration method is similar to that of <b>Allow Conditions</b>.</p>

**Step 4** (Optional) Add the validity period of the policy. Click **Add Validity period** in the upper right corner of the page, set **Start Time** and **End Time**, and select **Time Zone**. Click **Save**. To add multiple policy validity periods, click . To delete a policy validity period, click .

**Step 5** Click **Add** to view the basic information about the policy in the policy list. After the policy takes effect, check whether the related permissions are normal.

To disable a policy, click  to edit the policy and set the policy to **Disabled**.

If a policy is no longer used, click  to delete it.

----End

## 22.15 Ranger Log Overview

### Log Description

**Log path:** The default storage path of Ranger logs is `/var/log/Bigdata/ranger/Role name`.

- RangerAdmin: `/var/log/Bigdata/ranger/rangeradmin` (run logs)
- TagSync: `/var/log/Bigdata/ranger/tagsync` (run logs)
- UserSync: `/var/log/Bigdata/ranger/usersync` (run logs)

**Log archive rule:** The automatic compression and archive function is enabled for Ranger logs. By default, when the size of a log file exceeds 20 MB, the log file is automatically compressed. The naming rule of the compressed log file is as follows: `<Original log file name>-<yyyy-mm-dd_hh-mm-ss>.[ID].log.zip`. A maximum of 20 compressed file are retained.

**Table 22-21** HDFS log list

Type	Name	Description
RangerAdmin run log file	access_log.<DATE>.log	Tomcat access log
	catalina.out	Tomcat service run log
	gc-worker.log	RangerAdmin garbage collection (GC) log
	postinstallDetail.log	Work log generated after an instance is started before installation
	prestartDetail.log	Log that records preparations before instance startup

Type	Name	Description
	ranger-admin- <hostname>.log	RangerAdmin run log
	ranger_admin_sql- <hostname>.log	RangerAdmin log used to retrieve DBService
	startDetail.log	Instance startup log
TagSync run log	cleanupDetail.log	Instance clearing log
	gc-worker.log	GC log file of an instance
	postinstallDetail.log	Work log generated after an instance is started before installation
	prestartDetail.log	Log that records preparations before instance startup
	ranger-tagsync- <hostname>.log	TagSync run log
	startDetail.log	Instance startup log
	tagsync.out	TagSync run log
UserSync run log	auth.log	UnixAuth service run log
	cleanupDetail.log	Instance clearing log
	gc-worker.log	GC log file of an instance
	postinstallDetail.log	Work log generated after an instance is started before installation
	prestartDetail.log	Log that records preparations before instance startup
	ranger-usersync- <hostname>.log	UserSync run log
	startDetail.log	Instance startup log

## Log Levels

**Table 22-22** describes the log levels provided by HDFS. The priorities of log levels are FATAL, ERROR, WARN, INFO, and DEBUG in descending order. Logs whose levels are higher than or equal to the specified level are printed. The number of printed logs decreases as the specified log level increases.

**Table 22-22** Log levels

Level	Description
FATAL	Logs of this level record fatal error information about the current event processing that may result in a system crash.
ERROR	Logs of this level record error information about the current event processing, which indicates that system running is abnormal.
WARN	Logs of this level record abnormal information about the current event processing. These abnormalities will not result in system faults.
INFO	Logs of this level record normal running status information about the system and events.
DEBUG	Logs of this level record the system information and system debugging information.

To modify log levels, perform the following operations:

- Step 1** Log in to FusionInsight Manager.
- Step 2** Choose **Cluster > Services > Ranger > Configurations**.
- Step 3** Select **All Configurations**.
- Step 4** On the menu bar on the left, select the log menu of the target role.
- Step 5** Select a desired log level.
- Step 6** Click **Save**. In the displayed dialog box, click **OK** to make the configuration take effect.

 **NOTE**

The configurations take effect immediately without the need to restart the service.

----**End**

## Log Formats

The following table lists the Ranger log formats.

**Table 22-23** Log formats

Type	Format	Example Value
Run log	<yyyy-MM-dd HH:mm:ss,SSS> <Log level> <Name of the thread that generates the log> <Message in the log> <Location where the log event occurs>	2020-04-29 20:09:28,543   INFO   http-bio-21401- exec-56   Request comes from API call, skip cas filter.   CasAuthenticationFilter- Wrapper.java:25

## 22.16 Common Issues About Ranger

### 22.16.1 Why Ranger Startup Fails During the Cluster Installation?

#### Problem

During cluster installation, Ranger fails to be started, and the error message "ERROR: cannot drop sequence X\_POLICY\_REF\_ACCESS\_TYPE\_SEQ " is displayed in the task list of the Manager process. How do I resolve this problem and properly install Ranger?

#### Answer

This issue may occur when two RangerAmdin instances are installed. If the instance installation fails, manually restart one RangerAdmin instance and then restart the other instance.

### 22.16.2 How Do I Determine Whether the Ranger Authentication Is Used for a Service?

#### Question

How do I determine whether the Ranger authentication is enabled for a service that supports the authentication?

#### Answer

Log in to FusionInsight Manager and choose **Cluster** > **Services** > *Name of the desired service*. On the service details page, click **More** and check whether the **Enable Ranger** option is available.

- If yes, the Ranger authentication plug-in is not enabled for the service. You can click **Enable Ranger** to enable the function.
- If no, the Ranger authentication plug-in has been enabled for the service. You can configure the permission policy for accessing the service resources on the Ranger management page.

## 22.16.3 Why Cannot a New User Log In to Ranger After Changing the Password?

### Question

When a new user logs in to Ranger, why is the 401 error reported after the password is changed?

### Answer

The UserSync synchronizes user data at an interval of 5 minutes by default. Therefore, a new user created on Manager cannot log in to the Ranger before the user data is successfully synchronized because the Ranger database does not have the user information. The user can log in to the Ranger only after the specified interval ends.

In non-security mode, the Ranger does not synchronize user data from Manager. Therefore, only the **admin** user can log in to the Ranger page.

## 22.16.4 When an HBase Policy Is Added or Modified on Ranger, Wildcard Characters Cannot Be Used to Search for Existing HBase Tables


### Question

When a Ranger access permission policy is added for HBase and wildcard characters are used to search for an existing HBase table in the policy, the table cannot be found. The following error is reported in `/var/log/Bigdata/ranger/rangeradmin/ranger-admin-*log`:

```
Caused by: javax.security.sasl.SaslException: No common protection layer between client and server
at com.sun.security.sasl.gsskerb.GssKrb5Client.doFinalHandshake(GssKrb5Client.java:253)
at com.sun.security.sasl.gsskerb.GssKrb5Client.evaluateChallenge(GssKrb5Client.java:186)
at
org.apache.hadoop.hbase.security.AbstractHBaseSaslRpcClient.evaluateChallenge(AbstractHBaseSaslRpcClient.java:142)
at org.apache.hadoop.hbase.security.NettyHBaseSaslRpcClientHandler
$2.run(NettyHBaseSaslRpcClientHandler.java:142)
at org.apache.hadoop.hbase.security.NettyHBaseSaslRpcClientHandler
$2.run(NettyHBaseSaslRpcClientHandler.java:138)
at java.security.AccessController.doPrivileged(Native Method)
at javax.security.auth.Subject.doAs(Subject.java:422)
at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1761)
at
org.apache.hadoop.hbase.security.NettyHBaseSaslRpcClientHandler.channelRead0(NettyHBaseSaslRpcClientHandler.java:138)
at
org.apache.hadoop.hbase.security.NettyHBaseSaslRpcClientHandler.channelRead0(NettyHBaseSaslRpcClientHandler.java:42)
at
org.apache.hadoop.hbase.thirdparty.io.netty.channel.SimpleChannelInboundHandler.channelRead(SimpleChannelInboundHandler.java:105)
at
org.apache.hadoop.hbase.thirdparty.io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChannelHandlerContext.java:362)
```

## Answer

The value of **hbase.rpc.protection** of the HBase service plug-in on Ranger must be the same as that of **hbase.rpc.protection** on the HBase server.

- Step 1** Log in to the Ranger management page. For details, see [Logging In to the Ranger Web UI](#).
  - Step 2** In the **HBASE** area on the home page, click the component plug-in name, for example, the  button of HBase.
  - Step 3** Search for the configuration item **hbase.rpc.protection** and change its value to the value of **hbase.rpc.protection** on the HBase server.
  - Step 4** Click **Save**.
- End



# 23 Using Spark

---

## 23.1 Precautions

This section applies to versions earlier than MRS 3.x.

## 23.2 Getting Started with Spark

This section describes how to use Spark to submit a SparkPi job. SparkPi, a typical Spark job, is used to calculate the value of Pi ( $\pi$ ).

### Procedure

**Step 1** Prepare the SparkPi program.

Multiple open-source Spark sample programs are provided, including SparkPi. Click <https://archive.apache.org/dist/spark/spark-2.1.0/spark-2.1.0-bin-hadoop2.7.tgz> to download the software package.

Decompress the software package to obtain the `spark-examples_2.11-2.1.0.jar` file, the sample program package, in the `spark-2.1.0-bin-hadoop2.7/examples/jars` directory. The `spark-examples_2.11-2.1.0.jar` sample program package contains the SparkPi program.

**Step 2** Upload data to OBS.

1. Log in to OBS Console.
2. Choose **Parallel File System > Create Parallel File System** to create a file system named **sparkpi**.  
**sparkpi** is only an example. The file system name must be globally unique. Otherwise, the parallel file system fails to be created. Use the default values for other parameters.
3. Click the file system name **sparkpi** and click **Files**.
4. Click **Create Folder** to create the **program** folder..
5. Go to the **program** folder, click **Upload Object**, select the program package downloaded in **Step 1** from the local PC, and set **Storage Class** to **Standard**.

**Step 3** Log in to the MRS console. In the left navigation pane, choose **Clusters > Active Clusters**, and click a cluster name.

**Step 4** Submit the SparkPi job.

On the MRS console, click the **Jobs** tab and click **Create**. The **Create Job** page is displayed. For details about how to submit the job, see [Running a Spark Job](#).

- Set **Type** to **SparkSubmit**.
- Set **Name** to **sparkPi**.
- Set **Program Path** to the path where programs are stored on OBS, for example, **obs://sparkpi/program/spark-examples\_2.11-2.1.0.jar**.
- In **Program Parameter**, select **--class** for **Parameter** and set **Value** to **org.apache.spark.examples.SparkPi**.
- Set **Parameters** to **10**.
- Leave **Service Parameter** blank.

A job can be submitted only when the cluster is in the **Running** state.

After a job is submitted successfully, it is in the **Accepted** state by default. You do not need to manually execute the job.

**Step 5** View the job execution result.

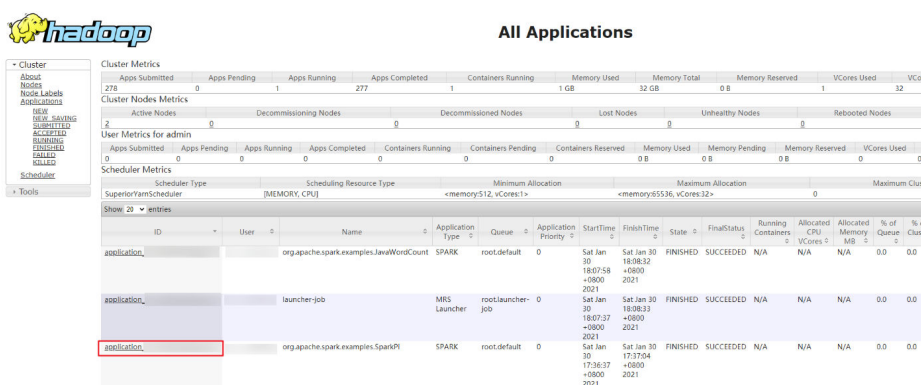
1. Go to the **Jobs** tab page and view job execution status.

The job execution takes a while. After the jobs are complete, refresh the job list.

Once a job has succeeded or failed, you cannot execute it again. However, you can add or copy a job, and set job parameters to submit a job again.

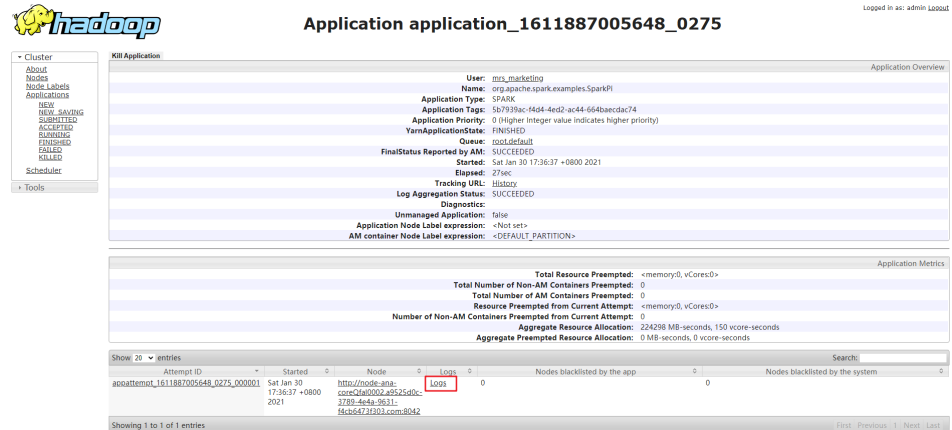
2. Go to the native Yarn page and view the job output information.
  - a. On the **Jobs** tab page, locate the row that contains the target job and click **View Details** in the **Operation** column to obtain the actual job ID.
  - b. Log in to Manager and choose **Services > Yarn > ResourceManager WebUI > ResourceManager (Active)**. The Yarn page is displayed.
  - c. Click the ID corresponding to the actual job ID.

**Figure 23-1** Yarn Web UI



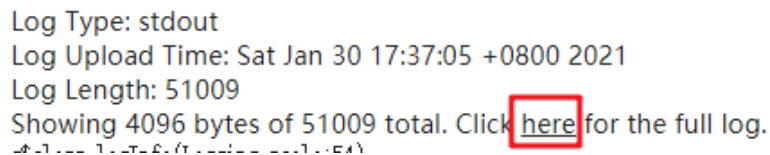
- d. Click **Logs** in the job log area.

Figure 23-2 sparkPi job logs



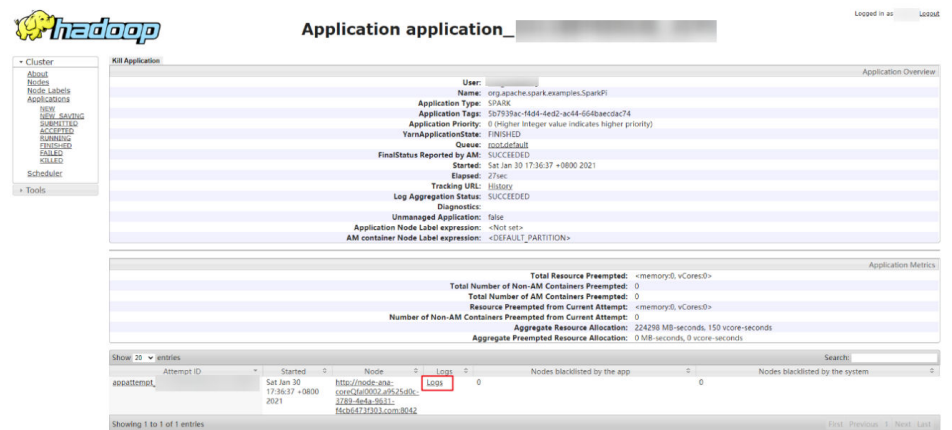
- e. Click [here](#) to obtain more detailed logs.

Figure 23-3 More detailed logs of sparkPi jobs



- f. Obtain the job execution result.

Figure 23-4 SparkPi job logs



----End

## 23.3 Getting Started with Spark SQL

Spark provides the Spark SQL language that is similar to SQL to perform operations on structured data. This section describes how to use Spark SQL from scratch. Create a table named `src_data`, write a data record in each row of the table, and store the data in the `mrs_20160907` cluster. Then use SQL statements to query data in the table, and delete the table at last.

## Prerequisites

You have obtained the AK/SK for writing data from an OBS data source to a Spark SQL table. To obtain it, perform as follows:

1. Log in to the management console.
2. Click the username and select **My Credentials** from the drop-down list.
3. On the displayed **My Credentials** page, click **Access Keys**.
4. Click **Create Access Key** to switch to the **Create Access Key** dialog box.
5. Enter the password and , and click **OK** to download the access key. Keep the access key secure.

## Procedure

**Step 1** Prepare data sources for Spark SQL analysis.

The sample text file is as follows:

```
abcd3ghji
efgh658ko
1234jjyu9
7h8kodfg1
kk99icxz3
```

**Step 2** Upload data to OBS.

1. Log in to OBS Console.
2. Choose **Parallel File System > Create Parallel File System** to create a file system named **sparksql**.  
**sparksql** is only an example. The file system name must be globally unique. Otherwise, the parallel file system fails to be created.
3. Click the name of the **sparksql** file system and click **Files**.
4. Click **Create Folder** to create the **input** folder.
5. Go to the **input** folder, choose **Upload File > add file**, select the local TXT file, and click **Upload**.

**Step 3** Log in to the MRS console. In the left navigation pane, choose **Clusters > Active Clusters**, and click a cluster name.

**Step 4** Import the text file from OBS to HDFS.

1. Click the **Files** tab.
2. On the **HDFS File List** tab page, click **Create Folder**, and create a folder named **userinput**.
3. Go to the **userinput** folder, and click **Import Data**.
4. Select the OBS and HDFS paths and click **OK**.

**OBS Path:** `obs://sparksql/input/sparksql-test.txt`

**HDFS Path:** `/user/userinput`

**Step 5** Submit the SQL statement.

1. On the MRS console, select **Job Management**. For details about how to submit the statement, see [Running a Spark Job](#).

A job can be submitted only when the **mrs\_20160907** cluster is in the **Running** state.

2. Enter the Spark SQL statement for table creation.

When entering Spark SQL statements, ensure that the statement characters are not more than 10,000.

Syntax:

```
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] table_name [(col_name data_type [COMMENT col_comment], ...)] [COMMENT table_comment] [PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)] [CLUSTERED BY (col_name, col_name, ...) [SORTED BY (col_name [ASC|DESC], ...)] INTO num_buckets BUCKETS] [ROW FORMAT row_format] [STORED AS file_format] [LOCATION hdfs_path];
```

You can use the following two methods to create a table example:

- Method 1: Create table **src\_data** and write data in every row.
  - The data source is stored in the folder of HDFS: ***create external table src\_data(line string) row format delimited fields terminated by '\|n' stored as textfile location '/user/userinput'***;
  - The data source is stored in the **/sparksql/input** folder of OBS: ***create external table src\_data(line string) row format delimited fields terminated by '\\n' stored as textfile location 'obs://AK:SK@sparksql/input'***;  
For details about how to obtain the AK/SK, see [Prerequisites](#).
- Method 2: Create table **src\_data1** and load data to the table in batches.  
***create table src\_data1 (line string) row format delimited fields terminated by ',' ;***  
***load data inpath '/user/userinput/sparksql-test.txt' into table src\_data1;***

 NOTE

When method 2 is used, the data from OBS cannot be loaded to the created tables directly.

3. Enter the Spark SQL statement for table query.

Syntax:

```
SELECT col_name FROM table_name;
```

Example of querying all data in the **src\_data** table:

```
select * from src_data;
```

4. Enter the Spark SQL statement for table deletion.

Syntax:

```
DROP TABLE [IF EXISTS] table_name;
```

Example of deleting the **src\_data** table:

```
drop table src_data;
```

5. Click **Check** to check the statement correctness.
6. Click **OK**.

After the Spark SQL statements are submitted, the statement execution results are displayed in the result column.

**Step 6** Delete the cluster.

----End

## 23.4 Using the Spark Client

After an MRS cluster is created, you can create and submit jobs on the client. The client can be installed on nodes inside or outside the cluster.

- Nodes inside the cluster: After an MRS cluster is created, the client has been installed on the master and core nodes in the cluster by default. For details, see . Then, log in to the node where the MRS client is installed..
- Nodes outside the cluster: You can install the client on nodes outside a cluster. For details about how to install a client, see , and log in to the node where the MRS client is installed..

### Using the Spark Client

**Step 1** Based on the client location, log in to the node where the client is installed. For details, see , or .

**Step 2** Run the following command to go to the client installation directory:

```
cd /opt/client
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 4** If the cluster is in security mode, run the following command to authenticate the user. In normal mode, user authentication is not required.

```
kinit Component service user
```

**Step 5** Run the Spark shell command. The following provides an example:

```
spark-beeline
```

```
----End
```

## 23.5 Accessing the Spark Web UI

The Spark web UI is used to view the running status of Spark applications. Google Chrome is recommended for better user experience.

Spark has two web UIs.

- Spark UI: used to display the status of running applications.  
The UI includes the following parts: Jobs, Stages, Storage, Environment, Executors, SQL, and JDBC/ODBC Server. The Streaming application has the Streaming tab in addition to the preceding parts.
- History Server UI: used to display the status of Spark applications that are complete or incomplete.  
The UI includes the application ID, application name, start time, end time, execution time, and owner information.

## Spark UI

### Step 1 Access the component management page.

- For versions earlier than MRS 3.x, click the cluster name to go to the cluster details page and choose **Components**.

 **NOTE**

- If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)
- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services**.

### Step 2 Select Yarn. In the Yarn Summary area, click ResourceManager in ResourceManager Web UI to access the web UI.

### Step 3 Locate the Spark application. Click ApplicationMaster in the last column of the application information. The Spark UI is displayed.

Figure 23-5 ApplicationMaster

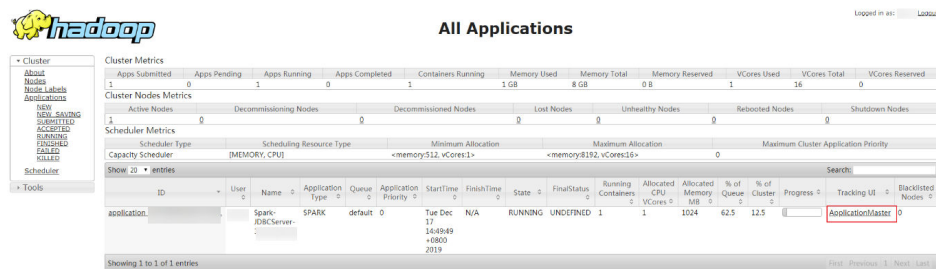


Figure 23-6 Spark UI



----End

## History Server

### Step 1 Access the component management page.

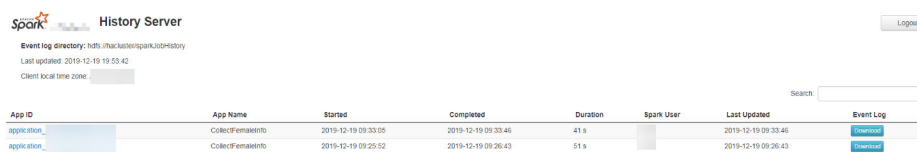
- For versions earlier than MRS 3.x, click the cluster name to go to the cluster details page and choose **Components**.

 **NOTE**

- If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)
- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services**.

### Step 2 Select Spark. In the Spark Summary area, click JobHistory corresponding to Spark Web UI to access the web UI.

Figure 23-7 Spark History Server



----End

## 23.6 Interconnecting Spark with OpenTSDB

### 23.6.1 Creating a Table and Associating It with OpenTSDB

#### Function

MRS Spark can be used to access the data source of OpenTSDB, create and associate tables in the Spark, and query and insert the OpenTSDB data.

Use the **CREATE TABLE** command to create a table and associate it with an existing metric in OpenTSDB.

#### NOTE

If no metric exists in OpenTSDB, an error will be reported when the corresponding table is queried.

#### Syntax

```
CREATE TABLE [IF NOT EXISTS] OPENTSDB_TABLE_NAME USING OPENTSDB OPTIONS (
'metric' = 'METRIC_NAME',
'tags' = 'TAG1,TAG2'
);
```

#### Keyword

Parameter	Description
metric	Indicates the name of the metric in OpenTSDB corresponding to the table to be created.
tags	Indicates the tags corresponding to the metric. The tags are used for classification, filtering, and quick retrieval. You can set 1 to 8 tags, which are separated by commas (.). The parameter value includes values of all tagKs in the corresponding metric.

#### Precautions

When creating a table, you do not need to specify the **timestamp** and **value** fields. The system automatically builds the following fields based on the specified tags. The fields **TAG1** and **TAG2** are specified by tags.



- TAG1 String
- TAG2 String
- timestamp Timestamp
- value double

## Example

Create table **opentsdb\_table** and associate it with metric **city.temp** of the OpenTSDB component.

```
CREATE table opentsdb_table using opentsdb OPTIONS ('metric='city.temp', 'tags'='city,location');
```

## 23.6.2 Inserting Data to the OpenTSDB Table

### Function

Run the **INSERT INTO** statement to insert the data in the table to the associated OpenTSDB metric.

### Syntax

```
INSERT INTO TABLE_NAME SELECT * FROM SRC_TABLE;
INSERT INTO TABLE_NAME VALUES(XXX);
```

### Keyword

Parameter	Description
TABLE_NAME	Indicates the name of the associated OpenTSDB table.
SRC_TABLE	Indicates the name of the table from which data is obtained. This parameter can be set to a name of a common table.

### Precautions

- The inserted data cannot be **null**. If the inserted data is the same as the original data or only the **value** is different, the inserted data overwrites the original data.
- **INSERT OVERWRITE** is not supported.
- You are advised not to concurrently insert data into a table. If you concurrently insert data into a table, there is a possibility that conflicts occur, leading to data insertion failures.
- The **TIMESTAMP** format supports only yyyy-MM-dd hh:mm:ss.

### Example

Insert data into table **opentsdb\_table**.

```
insert into opentsdb_table values('aaa','bbb','2021-05-03 00:00:00',21);
```

### 23.6.3 Querying an OpenTSDB Table

This **SELECT** command is used to query data in an OpenTSDB table.

#### Syntax

```
SELECT * FROM table_name WHERE tagk=tagv LIMIT number;
```

#### Keyword

Parameter	Description
LIMIT	Used to limit the query results.
number	Only the INT type is supported.

#### Precautions

- The to-be-queried table must exist. Otherwise, an error is reported.
- The value of **tagv** must exist. Otherwise, an error occurs.

#### Example

Query data in the **opentsdb\_table** table.

```
SELECT * FROM opentsdb_table LIMIT 100;
SELECT * FROM opentsdb_table WHERE city='aaa';
```

### 23.6.4 Modifying the Default Configuration Data

By default, OpenTSDB connects to the local TSD process of the node where the Spark executor resides. In MRS, use the default configuration.

**Table 23-1** OpenTSDB data source configuration

Parameter	Description	Example Value
spark.sql.datasource.opentsdb.host	Indicates the IP address of the connected TSD process.	Null (default value) <b>xx.xx.xx.xx</b> indicates the IP address. Separate multiple IP addresses with commas (,).
spark.sql.datasource.opentsdb.port	Indicates the port number of the TSD process.	4242 (default value)

Parameter	Description	Example Value
spark.sql.datasource.opentsdb.randomSeed	Indicates whether to use the random seed when the <b>spark.sql.datasource.opentsdb.host</b> is set to multiple addresses. If this parameter is set to <b>false</b> , all executors on the same node are connected to the same host. In this way, <b>spark.blacklist.enabled=true</b> can be used to implement task fault tolerance.	false (default value)

## Example

Run the **set** statement in **spark-sql** and **spark-beeline**, and then run other SQL statements.

```
set spark.sql.datasource.opentsdb.host = 192.168.2.143,192.168.2.158;
SELECT * FROM opentsdb_table ;
```

# 24 Using Spark2x

---

## 24.1 Precautions

This section applies to MRS 3.x or later.

## 24.2 Basic Operation

### 24.2.1 Getting Started

This section describes how to use Spark2x to submit Spark applications, including Spark Core and Spark SQL. Spark Core is the kernel module of Spark. It executes tasks and is used to compile Spark applications. Spark SQL is a module that executes SQL statements.

#### Scenario Description

Develop a Spark application to perform the following operations on logs about netizens' dwell time for online shopping on a weekend.

- Collect statistics on female netizens who dwell on online shopping for more than 2 hours on the weekend.
- The first column in the log file records names, the second column records genders, and the third column records the dwell durations in the unit of minute. Three columns are separated by comma (,).

**log1.txt:** logs collected on Saturday

```
LiuYang,female,20
YuanJing,male,10
GuoYijun,male,5
CaiXuyu,female,50
Liyuan,male,20
FangBo,female,50
LiuYang,female,20
YuanJing,male,10
GuoYijun,male,50
CaiXuyu,female,50
FangBo,female,60
```

**log2.txt:** logs collected on Sunday

```
LiuYang,female,20
YuanJing,male,10
CaiXuyu,female,50
FangBo,female,50
GuoYijun,male,5
CaiXuyu,female,50
Liyuan,male,20
CaiXuyu,female,50
FangBo,female,50
LiuYang,female,20
YuanJing,male,10
FangBo,female,50
GuoYijun,male,50
CaiXuyu,female,50
FangBo,female,60
```

## Prerequisites

- On Manager, you have created a user and granted the HDFS, Yarn, Kafka, and Hive permissions to the user.
- You have installed and configured tools such as IntelliJ IDEA and JDK based on the development language.
- You have installed the Spark2x client and configured the client network connection.
- For Spark SQL programs, you have started Spark SQL or Beeline on the client to enter SQL statements.

## Procedure

**Step 1** Obtain the sample project and import it to IDEA. Import the JAR package on which the sample project depends. Use IDEA to configure and generate JAR packages.

**Step 2** Prepare the data required by the sample project.

Save the original log files in the scenario description to the HDFS system.

1. Create two text files (**input\_data1.txt** and **input\_data2.txt**) on the local host and copy the content in the **log1.txt** and **log2.txt** files to the **input\_data1.txt** and **input\_data2.txt** files, respectively.
2. Create the **/tmp/input** directory in HDFS, and upload **input\_data1.txt** and **input\_data2.txt** to the **/tmp/input** directory:

**Step 3** Upload the generated JAR package to the Spark2x running environment (Spark2x client), for example, **/opt/female**.

**Step 4** Go the client directory, configure the environment variables, and log in to the system. When you use a client to connect to a specific instance in a scenario where multiple Spark2x instances are installed or Spark and Spark2x instances are installed, run the following commands to load the environment variables of the instance.

```
source bigdata_env
```

```
source Spark2x/component_env
```

```
kinit <service user for authentication>
```

**Step 5** Run the following script in the **bin** directory to submit the Spark application:

```
spark-submit --class com.xxx.bigdata.spark.examples.FemaleInfoCollection --master yarn-client /opt/female/FemaleInfoCollection.jar <inputPath>
```

**NOTE**

- **FemaleInfoCollection.jar** is the JAR package generated in **Step 1**.
- **<inputPath>** is the directory created in **Step 2.2**.

**Step 6** (Optional) After calling the **spark-sql** or **spark-beeline** script in the **bin** directory, directly enter SQL statements to perform operations such as query.

For example, create a table, insert a piece of data, and then query the table.

```
spark-sql> CREATE TABLE TEST(NAME STRING, AGE INT);
Time taken: 0.348 seconds
spark-sql>INSERT INTO TEST VALUES('Jack', 20);
Time taken: 1.13 seconds
spark-sql> SELECT * FROM TEST;
Jack 20
Time taken: 0.18 seconds, Fetched 1 row(s)
```

**Step 7** View the running result of the Spark application.

- View the running result data in a specified file.
 

The storage path and format of the result data are specified by the Spark application.
- Check the running status on the web page.
  - Log in to Manager. Select **Spark2x** from the **Service** drop-down list.
  - Go to the Spark2x overview page and click an instance in the Spark web UI, for example, **JobHistory2x(host2)**.
  - The History Server UI is displayed.

The History Server UI is used to display the status of Spark applications that are complete or incomplete.

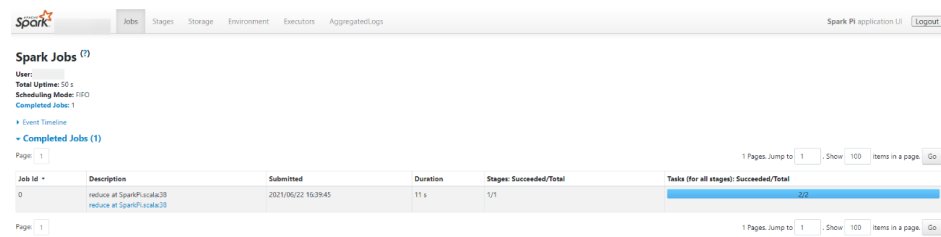
**Figure 24-1** History Server UI

Version	App ID	App Name	Started	Completed	Duration	Spark User	Last Updated	Event Log
	application_...	Spark Pi	2021-06-22 16:39:06	2021-06-22 16:39:56	50 s		2021-06-22 16:39:56	Download
	application_...	Spark Pi	2021-06-22 16:39:11	2021-06-22 16:39:55	45 s		2021-06-22 16:39:55	Download
	application_...	Spark Pi	2021-06-22 16:39:10	2021-06-22 16:39:55	44 s		2021-06-22 16:39:55	Download
	application_...	Spark Pi	2021-06-22 16:39:10	2021-06-22 16:39:46	35 s		2021-06-22 16:39:46	Download
	application_...	Spark Pi	2021-06-22 16:39:06	2021-06-22 16:39:44	38 s		2021-06-22 16:39:44	Download
	application_...	Spark Pi	2021-06-22 16:39:05	2021-06-22 16:39:26	21 s		2021-06-22 16:39:26	Download
	application_...	Spark Pi	2021-06-22 16:38:13	2021-06-22 16:39:05	52 s		2021-06-22 16:39:05	Download
	application_...	Spark Pi	2021-06-22 16:38:13	2021-06-22 16:38:57	45 s		2021-06-22 16:38:58	Download
	application_...	Spark Pi	2021-06-22 16:38:12	2021-06-22 16:38:57	45 s		2021-06-22 16:38:57	Download
	application_...	Spark Pi	2021-06-22 16:38:12	2021-06-22 16:38:54	42 s		2021-06-22 16:38:54	Download
	application_...	Spark Pi	2021-06-22 16:38:09	2021-06-22 16:38:47	38 s		2021-06-22 16:38:47	Download
	application_...	Spark Pi	2021-06-22 16:38:05	2021-06-22 16:38:46	41 s		2021-06-22 16:38:46	Download
	application_...	Spark Pi	2021-06-22 16:38:06	2021-06-22 16:38:27	21 s		2021-06-22 16:38:27	Download
	application_...	Spark Pi	2021-06-22 16:38:55	2021-06-22 16:39:06	1.2 min		2021-06-22 16:39:06	Download

- Select an application ID and click this page to go to the Spark UI of the application.

Spark UI: used to display the status of running applications.

Figure 24-2 Spark UI



- View Spark logs to learn application runtime conditions. View [Spark2x Logs](#) to learn application running status, and adjust applications based on log information.

----End

## 24.2.2 Configuring Parameters Rapidly

### Overview

This section describes how to quickly configure common parameters and lists parameters that are not recommended to be modified when Spark2x is used.

### Common parameters to be configured

Some parameters have been adapted during cluster installation. However, the following parameters need to be adjusted based on application scenarios. Unless otherwise specified, the following parameters are configured in the **spark-defaults.conf** file on the Spark2x client.

Table 24-1 Common parameters to be configured

Configuration Item	Description	Default Value
spark.sql.parquet.compression.codec	Used to set the compression format of a non-partitioned Parquet table. Set the queue in the <b>spark-defaults.conf</b> configuration file on the JDBCServer server.	snappy
spark.dynamicAllocation.enabled	Indicates whether to use dynamic resource scheduling, which is used to adjust the number of executors registered with the application according to scale. Currently, this parameter is valid only in Yarn mode. The default value for JDBCServer is <b>true</b> , and that for the client is <b>false</b> .	false

Configuration Item	Description	Default Value
spark.executor.memory	Indicates the memory size used by each executor process. Its character sting is in the same format as the JVM memory (example: 512 MB or 2 GB).	4G
spark.sql.autoBroadcastJoinThreshold	Indicates the maximum value for the broadcast configuration when two tables are joined. <ul style="list-style-type: none"> <li>When the size of a field in a table involved in an SQL statement is less than the value of this parameter, the system broadcasts the SQL statement.</li> <li>If the value is set to <b>-1</b>, broadcast is not performed.</li> </ul>	10485760
spark.yarn.queue	Specifies the Yarn queue where JDBCServer resides. Set the queue in the <b>spark-defaults.conf</b> configuration file on the JDBCServer server.	default
spark.driver.memory	In a large cluster, you are advised to configure the memory used by the 32 GB to 64 GB driver process, that is, the SparkContext initialization process (for example, 512 MB and 2 GB).	4G
spark.yarn.security.credentials.hbase.enabled	Indicates whether to enable the function of obtaining HBase tokens. If the Spark on HBase function is required and a security cluster is configured, set this parameter to <b>true</b> . Otherwise, set this parameter to <b>false</b> .	false
spark.serializer	Used to serialize the objects that are sent over the network or need to be cached. The default value of Java serialization applies to any Serializable Java object, but the running speed is slow. Therefore, you are advised to use <b>org.apache.spark.serializer.KryoSerializer</b> and configure Kryo serialization. It can be any subclass of <b>org.apache.spark.serializer.Serializer</b> .	org.apache.spark.serializer.JavaSerializer



Configuration Item	Description	Default Value
spark.executor.cores	Indicates the number of kernels used by each executor. Set this parameter in standalone mode and Mesos coarse-grained mode. When there are sufficient kernels, the application is allowed to execute multiple executable programs on the same worker. Otherwise, each application can run only one executable program on each worker.	1
spark.shuffle.service.enabled	Indicates a long-term auxiliary service in NodeManager for improving shuffle computing performance.	false
spark.sql.adaptive.enabled	Indicates whether to enable the adaptive execution framework.	false
spark.executor.memoryOverhead	Indicates the heap memory to be allocated to each executor, in MB. This is the memory that occupies the overhead of the VM, similar to the internal string and other built-in overhead. The value increases with the executor size (usually 6% to 10%).	1 GB
spark.streaming.kafka.direct.lifo	Indicates whether to enable the LIFO function of Kafka.	false

## Parameters Not Recommended to Be Modified

The following parameters have been adapted during cluster installation. You are not advised to modify them.

**Table 24-2** Parameters not recommended to be modified

Configuration Item	Description	Default Value or Configuration Example
spark.password.factory	Selects the password parsing mode.	org.apache.spark.om.util.FIPasswordFactory
spark.ssl.ui.protocol	Sets the SSL protocol of the UI.	TLSv1.2

Configuration Item	Description	Default Value or Configuration Example
spark.yarn.archive	Archives Spark JAR files, which are distributed to Yarn cache. If this parameter is set, the value will replace <code>&lt;code&gt;spark.yarn.jars &lt;/code&gt;</code> and be archived in the containers of all applications. The archive should contain the JAR files in its root directory. Archives can also be hosted on HDFS to speed up file distribution.	hdfs://hacluster/user/spark2x/jars/8.0.2.1/spark-archive-2x.zip <b>NOTE</b> The version 8.0.2.1 is used as an example. Replace it with the actual version number.
spark.yarn.am.extraJavaOptions	Indicates a string of extra JVM options to pass to the YARN ApplicationMaster in client mode. Use <b>spark.driver.extraJavaOptions</b> in cluster mode.	-Dlog4j.configuration=./__spark_conf__/_hadoop_conf__/log4j-executor.properties -Djava.security.auth.login.config=./__spark_conf__/_hadoop_conf__/jaas-zk.conf - Dzookeeper.server.principal=zookeeper/hadoop.<system domain name> - Djava.security.krb5.conf=./__spark_conf__/_hadoop_conf__/kdc.conf - Djdk.tls.ephemeralDHKeySize=2048
spark.shuffle.servicev2.port	Indicates the port for the shuffle service to monitor requests for obtaining data.	27338
spark.ssl.historyServer.enabled	Sets whether the history server uses SSL.	true
spark.files.override	When the target file exists and its content does not match that of the source file, whether to overwrite the file added through <b>SparkContext.addFile()</b> .	false

Configuration Item	Description	Default Value or Configuration Example
spark.yarn.cluster.driver.extraClassPath	Indicates the extraClassPath of the driver in Yarn-cluster mode. Set the parameter to the path and parameters of the server.	<code>\${BIGDATA_HOME}/common/runtime/security</code>
spark.driver.extraClassPath	Indicates the extra class path entries attached to the class path of the driver.	<code>\${BIGDATA_HOME}/common/runtime/security</code>
spark.yarn.dist.innerfiles	Sets the files that need to be uploaded to HDFS from Spark in Yarn mode.	<code>/Spark_path/spark/conf/s3p.file,/Spark_path/spark/conf/locals3.jceks</code> <i>Spark_path</i> is the installation path of the Spark client.
spark.sql.bigdata.register.dialect	Registers the SQL parser.	<code>org.apache.spark.sql.hbase.HBaseSQLParser</code>
spark.shuffle.manager	Indicates the data processing mode. There are two implementation modes: sort and hash. The sort shuffle has a higher memory utilization. It is the default option in Spark 1.2 and later versions.	<code>SORT</code>
spark.deploy.zookeeper.url	Indicates the address of ZooKeeper. Multiple addresses are separated by commas (,).	For example: <code>host1:2181,host2:2181,host3:2181</code>
spark.broadcast.factory	Indicates the broadcast mode.	<code>org.apache.spark.broadcast.TorrentBroadcastFactory</code>
spark.sql.session.state.builder	Session state constructor.	<code>org.apache.spark.sql.hive.FIHiveACLSessionStateBuilder</code>

Configuration Item	Description	Default Value or Configuration Example
spark.executor.extraLibraryPath	Sets the special library path used when the executor JVM is started.	<code>\${BIGDATA_HOME}/FusionInsight_HD_8.0.2.1/install/FusionInsight-Hadoop-3.1.1/hadoop/lib/native</code>
spark.ui.customErrorPage	Indicates whether to display the custom error information page when an error occurs on the page.	true
spark.httpdProxy.enable	Indicates whether to use the httpd proxy.	true
spark.ssl.ui.enabledAlgorithms	Sets the SSL algorithm of UI.	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,TLS_DHE_RSA_WITH_AES_256_GCM_SHA384,TLS_DHE_DSS_WITH_AES_256_GCM_SHA384,TLS_DHE_RSA_WITH_AES_128_GCM_SHA256,TLS_DHE_DSS_WITH_AES_128_GCM_SHA256
spark.ui.logout.enabled	Sets the logout button for the web UI of the Spark component.	true
spark.security.hideInfo.enabled	Indicates whether to hide sensitive information on the UI.	true
spark.yarn.cluster.driver.extraLibraryPath	Indicates the <b>extraLibraryPath</b> of the driver in Yarn-cluster mode. Set this parameter to the path and parameters of the server.	<code>\${BIGDATA_HOME}/FusionInsight_HD_8.0.2.1/install/FusionInsight-Hadoop-3.1.1/hadoop/lib/native</code>
spark.driver.extraLibraryPath	Sets a special library path for starting the driver JVM.	<code>\${DATA_NODE_INSTALL_HOME}/hadoop/lib/native</code>

Configuration Item	Description	Default Value or Configuration Example
spark.ui.killEnabled	Allows stages and jobs to be stopped on the web UI.	true
spark.yarn.access.hadoopFileSystems	Spark can access multiple NameService instances. If there are multiple NameService instances, set this parameter to all the NameService instances and separate them with commas (,).	hdfs://hacluster,hdfs://hacluster
spark.yarn.cluster.driver.extraJavaOptions	Indicates extra JVM option passed to the executor, for example, GC setting and logging. Do not set Spark attributes or heap size using this option. Instead, set Spark attributes using the SparkConf object or the <b>spark-defaults.conf</b> file specified when the spark-submit script is called. Set heap size using <b>spark.executor.memory</b> .	-Xloggc:<LOG_DIR>/gc.log -XX: +PrintGCDetails -XX:-OmitStackTraceln- FastThrow -XX:+PrintGCTimeStamps -XX: +PrintGCDateStamps -XX: +UseGCLogFileRotation - XX:NumberOfGCLogFiles=20 - XX:GCLogFileSize=10M - Dlog4j.configuration=../_spark_conf_ _hadoop_conf_/log4j-executor.properties -Djava.security.auth.login.config=/ _spark_conf_/_hadoop_conf_/jaas- zk.conf - Dzookeeper.server.principal=zookeeper/ hadoop.<system domain name> - Djava.security.krb5.conf=../_spark_conf_ _hadoop_conf_/kdc.conf - Djetty.version=x.y.z - Dorg.xerial.snappy.tmpdir=\${ BIGDATA_HOME}/tmp/spark2x_app - Dcarbon.properties.filepath=/ _spark_conf_/_hadoop_conf_ carbon.properties - Djdk.tls.ephemeralDHKeySize=2048

Configuration Item	Description	Default Value or Configuration Example
spark.driver.extraJavaOptions	Indicates a series of extra JVM options passed to the driver,	<pre>-Xloggc:\${SPARK_LOG_DIR}/indexserver-omm-%p-gc.log -XX:+PrintGCDetails -XX:-OmitStackTracelnFastThrow -XX:+PrintGCTimeStamps -XX:+PrintGCDateStamps -XX:MaxDirectMemorySize=512M -XX:MaxMetaspaceSize=512M -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=20 -XX:GCLogFileSize=10M -XX:OnOutOfMemoryError='kill -9 %p' -Djetty.version=x.y.z -Dorg.xerial.snappy.tmpdir=\${BIGDATA_HOME}/tmp/spark2x/JDBCServer/snappy_tmp -Djava.io.tmpdir=\${BIGDATA_HOME}/tmp/spark2x/JDBCServer/io_tmp -Dcarbon.properties.filepath=\${SPARK_CONF_DIR}/carbon.properties -Djdk.tls.ephemeralDHKeySize=2048 -Dspark.ssl.keyStore=\${SPARK_CONF_DIR}/child.keystore #{java_stack_prefer}</pre>
spark.eventLog.overwrite	Indicates whether to overwrite any existing file.	false
spark.eventLog.dir	Indicates the directory for logging Spark events if <b>spark.eventLog.enabled</b> is set to <b>true</b> . In this directory, Spark creates a subdirectory for each application and logs events of the application in the subdirectory. You can also set a unified address similar to the HDFS directory so that the History Server can read historical files.	hdfs://hacluster/spark2xJobHistory2x

Configuration Item	Description	Default Value or Configuration Example
spark.random.port.min	Sets the minimum random port.	22600
spark.authenticate	Indicates whether Spark authenticates its internal connections. If the application is not running on Yarn, see <b>spark.authenticate.secret</b> .	true
spark.random.port.max	Sets the maximum random port.	22899
spark.eventLog.enabled	Indicates whether to log Spark events, which are used to reconstruct the web UI after the application execution is complete.	true

Configuration Item	Description	Default Value or Configuration Example
spark.executor.extraJavaOptions	Indicates extra JVM option passed to the executor, for example, GC setting and logging. Do not set Spark attributes or heap size using this option.	<pre>-Xloggc:&lt;LOG_DIR&gt;/gc.log -XX: +PrintGCDetails -XX:-OmitStackTraceln- FastThrow -XX:+PrintGCTimeStamps -XX: +PrintGCDateStamps -XX: +UseGCLogFileRotation - XX:NumberOfGCLogFiles=20 - XX:GCLogFileSize=10M - Dlog4j.configuration=./log4j- executor.properties - Djava.security.auth.login.config=./jaas- zk.conf - Dzookeeper.server.principal=zookeeper/ hadoop.&lt;system domain name&gt; - Djava.security.krb5.conf=./kdc.conf - Dcarbon.properties.filepath=./ carbon.properties  -Xloggc:&lt;LOG_DIR&gt;/gc.log -XX: +PrintGCDetails -XX:-OmitStackTraceln- FastThrow -XX:+PrintGCTimeStamps -XX: +PrintGCDateStamps -XX: +UseGCLogFileRotation - XX:NumberOfGCLogFiles=20 - XX:GCLogFileSize=10M - Dlog4j.configuration=./_spark_conf_/ _hadoop_conf_/log4j-executor.properties -Djava.security.auth.login.config=./ _spark_conf_/_hadoop_conf_/jaas- zk.conf - Dzookeeper.server.principal=zookeeper/ hadoop.&lt;system domain name&gt; - Djava.security.krb5.conf=./_spark_conf_/ _hadoop_conf_/kdc.conf - Dcarbon.properties.filepath=./ _spark_conf_/_hadoop_conf_/ carbon.properties - Djdk.tls.ephemeralDHKeySize=2048</pre>
spark.sql.authorization.enabled	Indicates whether to enable authentication for the Hive client.	true

## 24.2.3 Common Parameters

### Overview

This section describes common configuration items used in Spark. This section is divided into sub-sections based on features to help you quickly find required



configuration items. If you use a MRS cluster, most parameters described in this section have been adapted and you do not need to configure them again. For details about the parameters that need to be configured based on the site requirements, see [Configuring Parameters Rapidly](#).

## Configuring the Number of Stage Retries

When `FetchFailedException` occurs in a Spark task, a stage retry is triggered. To prevent infinite stage retries, the number of stage retries is limited. The number of retry times can be adjusted based on the site requirements.

Configure the following parameters in the `spark-defaults.conf` file on the Spark client.

**Table 24-3** Parameter description

Parameter	Description	Default Value
<code>spark.stage.maxConsecutiveAttempts</code>	Indicates the maximum number of stage retries.	4

## Configuring Whether to Use Cartesian Product

To enable the Cartesian product function, configure the following parameter in the `spark-defaults.conf` configuration file of Spark.

**Table 24-4** Cartesian product parameters

Parameter	Description	Default Value
<code>spark.sql.crossJoin.enabled</code>	Indicates whether to allow implicit Cartesian product execution. <ul style="list-style-type: none"> <li><b>true:</b> Implicit Cartesian product execution is allowed.</li> <li><b>false:</b> Implicit Cartesian product execution is not allowed. In this case, only CROSS JOIN can be explicitly included in the query.</li> </ul>	true

### NOTE

- For JDBC applications, configure this parameter in the `spark-defaults.conf` configuration file of the server.
- For tasks submitted by the Spark client, configure this parameter in the `spark-defaults.conf` configuration file of the client.

## Configuring Security Authentication for Long-Time Spark Tasks

In security mode, if the **kinit** command is used for security authentication when the Spark CLI (such as `spark-shell`, `spark-sql`, or `spark-submit`) is used, the task fails due to authentication expiration when the task is running for a long time.

Set the following parameters in the **spark-defaults.conf** configuration file on the client. After the configuration is complete, run the Spark CLI again.

 **NOTE**

If this parameter is set to **true**, ensure that the values of **keytab** and **principal** in **spark-defaults.conf** and **hive-site.xml** are the same.

**Table 24-5** Parameter description

Parameter	Description	Default Value
spark.kerberos.principal	Indicates the principal user who has the Spark operation permission. Contact the MRS cluster administrator to obtain the principal user.	-
spark.kerberos.keytab	Indicates the name and path of the keytab file used to configure Spark operation permissions. Contact the MRS cluster administrator to obtain the keytab file.	-
spark.security.bigdata.loginOnce	<p>Indicates whether the principal user logs in to the system only once. <b>true</b>: single login; <b>false</b>: multiple logins.</p> <p>The difference between a single login and multiple logins is as follows: The Spark community uses the Kerberos user to log in to the system for multiple times. However, the TGT or token may expire, causing the application to fail to run for a long time. The Kerberos login mode of DataSight is modified to allow users to log in only once, which effectively resolves the expiration problem. The restrictions are as follows: The principal and keytab configuration items of Hive must be the same as those of Spark.</p> <p><b>NOTE</b> If this parameter is set to <b>true</b>, ensure that the values of <b>keytab</b> and <b>principal</b> in <b>spark-defaults.conf</b> and <b>hive-site.xml</b> are the same.</p>	true

## Python Spark

Python Spark is the third programming language of Spark except Scala and Java. Different from Java and Scala that run on the JVM platform, Python Spark has its own Python process as well as the JVM process. The following configuration items

apply only to Python Spark scenarios. However, other configuration items can also take effect in Python Spark scenarios.

**Table 24-6** Parameter description

Parameter	Description	Default Value
spark.python.profile	Indicates whether to enable profiling on the Python worker. Use <b>sc.show_profiles()</b> to display the analysis results or display the analysis results before the Driver exits. You can use <b>sc.dump_profiles(path)</b> to dump the results to a disk. If some analysis results have been manually displayed, they will not be automatically displayed before the driver exits.  By default, <b>pyspark.profiler.BasicProfiler</b> is used. You can transfer the specified profiler during SparkContext initialization to overwrite the default profiler.	false
spark.python.worker.memory	Indicates the memory size that can be used by each Python worker process during aggregation. The value format is the same as that of the specified JVM memory, for example, 512 MB and 2 GB. If the memory used by a process during aggregation exceeds the value of this parameter, data will be written to disks.	512m
spark.python.worker.reuse	Indicates whether to reuse Python workers. If the reuse function is enabled, a fixed number of Python workers will be reused by the next batch of submitted tasks instead of forking a Python process for each task. This function is useful in large-scale broadcasting because the data does not need to be transferred from the JVM to the Python workers again for the next batch of submitted tasks.	true

## Dynamic Allocation

Dynamic resource scheduling is a unique feature of the On Yarn mode. This function can be used only after Yarn External Shuffle is enabled. When Spark is used as a resident service, dynamic resource scheduling greatly improves resource utilization. For example, the JDBCServer process does not accept JDBC requests in most of the time. Therefore, releasing resources in this period greatly reduces the waste of cluster resources.

**Table 24-7** Parameter description

Parameter	Description	Default Value
spark.dynamicAllocation.enabled	Indicates whether to use dynamic resource scheduling, which is used to adjust the number of executors registered with the application according to scale. Currently, this parameter is valid only in Yarn mode.  To enable dynamic resource scheduling, set <b>spark.shuffle.service.enabled</b> to <b>true</b> . Related parameters are as follows: <b>spark.dynamicAllocation.minExecutors</b> , <b>spark.dynamicAllocation.maxExecutors</b> , and <b>spark.dynamicAllocation.initialExecutors</b> .	<ul style="list-style-type: none"> <li>JDBCServer2x: true</li> <li>SparkResource2x: false</li> </ul>
spark.dynamicAllocation.minExecutors	Indicates the minimum number of executors.	0
spark.dynamicAllocation.initialExecutors	Indicates the number of initial executors.	spark.dynamicAllocation.minExecutors
spark.dynamicAllocation.maxExecutors	Indicates the maximum number of executors.	2048
spark.dynamicAllocation.schedulerBacklogTimeout	Indicates the first timeout period for scheduling. The unit is second.	1s
spark.dynamicAllocation.sustainedSchedulerBacklogTimeout	Indicates the second and later timeout interval for scheduling.	1s
spark.dynamicAllocation.executorIdleTimeout	Indicates the idle timeout interval for common executors. The unit is second.	60

Parameter	Description	Default Value
spark.dynamicAllocation.cachedExecutorIdleTimeout	Indicates the idle timeout interval for executors with cached blocks.	<ul style="list-style-type: none"> <li>JDBCServer2x: 2147483647s</li> <li>IndexServer2x: 2147483647s</li> <li>SparkResource2x: 120</li> </ul>

## Spark Streaming

Spark Streaming is a streaming data processing function provided by the Spark batch processing platform. It processes data input from external systems in **mini-batch** mode.

Configure the following parameters in the **spark-defaults.conf** file on the Spark client.

**Table 24-8** Parameter description

Parameter	Description	Default Value
spark.streaming.receiver.writeAheadLog.enable	Indicates whether to enable the write-ahead log (WAL) function. After this function is enabled, all input data received by the receiver is saved in the WAL. WAL ensures that data can be restored if the driver program becomes faulty.	false
spark.streaming.unpersist	Determines whether to automatically remove RDDs generated and saved by Spark Streaming from the Spark memory. If this function is enabled, original data received by Spark Streaming is also automatically cleared. If this function is disabled, original data and RDDs cannot be automatically cleared. External applications can access the data in Streaming. This, however, occupies more Spark memory resources.	true

## Spark Streaming Kafka

The receiver is an important component of Spark Streaming. It receives external data, encapsulates the data into blocks, and provides the blocks for Streaming to

consume. The most common data source is Kafka. Spark Streaming integrates Kafka to ensure reliability and can directly use Kafka as the RDD input.

**Table 24-9** Parameter description

Parameter	Description	Default Value
spark.streaming.kafka.maxRatePerPartition	Indicates the maximum rate (number of records per second) for reading data from each Kafka partition if the Kafka direct stream API is used.	-
spark.streaming.blockInterval	Indicates the interval (ms) for accumulating data received by a Spark Streaming receiver into a data block before the data is stored in Spark. A minimum value of 50 ms is recommended.	200ms
spark.streaming.receiver.maxRate	Indicates the maximum rate (number of records per second) for each receiver to receive data. The value <b>0</b> or a negative value indicates no limit to the rate.	-
spark.streaming.receiver.writeAheadLog.enabled	Indicates whether to use ReliableKafkaReceiver. This receiver ensures the integrity of streaming data.	false

## Netty/NIO and Hash/Sort Configuration

Shuffle is critical for big data processing, and the network is critical for the entire shuffle process. Currently, Spark supports two shuffle modes: hash and sort. There are two network modes: Netty and NIO.

**Table 24-10** Parameter description

Parameter	Description	Default Value
spark.shuffle.manager	Indicates the data processing mode. There are two implementation modes: sort and hash. The sort shuffle has a higher memory utilization. It is the default option in Spark 1.2 and later versions.	SORT

Parameter	Description	Default Value
spark.shuffle.consolidateFiles	(Only in hash mode) To merge intermediate files created during shuffle, set this parameter to <b>true</b> . Decreasing the number of files to be created can improve the processing performance of the file system and reduce risks. If the <b>ext4</b> or <b>xfs</b> file system is used, you are advised to set this parameter to <b>true</b> . Due to file system restrictions, this setting on <b>ext3</b> may reduce the processing performance of a server with more than eight cores.	false
spark.shuffle.sort.byPassMergeThreshold	This parameter is valid only when <b>spark.shuffle.manager</b> is set to <b>sort</b> . When Map aggregation is not performed and the number of partitions for Reduce tasks is less than or equal to the value of this parameter, do not merge and sort data to prevent performance deterioration caused by unnecessary sorting.	200
spark.shuffle.io.maxRetries	(Only in Netty mode) If this parameter is set to a non-zero value, fetch failures caused by I/O-related exceptions will be automatically retried. This retry logic helps the large shuffle keep stable when long GC pauses or intermittent network disconnections occur.	12
spark.shuffle.io.numConnectionsPerPeer	(Only in Netty mode) Connections between hosts are reused to reduce the number of connections between large clusters. For a cluster with many disks but a few hosts, this function may make concurrent requests unable to occupy all disks. Therefore, you can increase the value of this parameter.	1
spark.shuffle.io.preferDirectBufs	(Only in Netty mode) The off-heap buffer is used to reduce GC during shuffle and cache block transfer. In an environment where off-heap memory is strictly limited, you can disable it to force all applications from Netty to use heap memory.	true
spark.shuffle.io.retryWait	(Only in Netty mode) Specifies the duration for waiting for fetch retry, in seconds. The maximum delay caused by retry is <b>maxRetries</b> x <b>retryWait</b> . The default value is 15 seconds.	5

## Common Shuffle Configuration

**Table 24-11** Parameter description

Parameter	Description	Default Value
spark.shuffle.spill	If this parameter is set to <b>true</b> , data is overflowed to the disk to limit the memory usage during a Reduce task.	true
spark.shuffle.spill.compress	Indicates whether to compress the data overflowed during shuffle. The algorithm specified by <b>spark.io.compression.codec</b> is used for data compression.	true
spark.shuffle.file.buffer	Specifies the size of the memory buffer for storing output streams of each shuffle file, in KB. These buffers can reduce the number of disk seek and system calls during the creation of intermediate shuffle file streams. You can also set this parameter by setting <b>spark.shuffle.file.buffer.kb</b> .	32KB
spark.shuffle.compress	Indicates whether to compress the output files of a Map task. You are advised to compress the broadcast variables. using <b>spark.io.compression.codec</b> .	true
spark.reducer.maxSizeInFlight	Specifies the maximum output size of the Map task that fetches data from each Reduce task, in MB. Each output requires a buffer, which is the fixed memory overhead of each Reduce task. Therefore, keep the value small unless there is a large amount of memory. You can also set this parameter by setting <b>spark.reducer.maxMbInFlight</b> .	48MB

## Driver Configuration

Spark driver can be considered as the client of Spark applications. All code parsing is completed in this process. Therefore, the parameters of this process are especially important. The following describes how to configure parameters for Spark driver.

- **JavaOptions:** parameter following **-D** in the Java command, which can be obtained by **System.getProperty**
- **ClassPath:** path for loading the Java classes and Native library
- **Java Memory and Cores:** memory and CPU usage of the Java process
- **Spark Configuration:** Spark internal parameter, which is irrelevant to the Java process



**Table 24-12** Parameter description

Parameter	Description	Default Value
spark.driver.extraJavaOptions	<p>Indicates a series of extra JVM options passed to the driver, for example, GC setting and logging.</p> <p>Note: In client mode, this configuration cannot be set directly in the application using SparkConf because the driver JVM has been started. You can use <b>--driver-java-options</b> or the default property file to set the parameter.</p>	<p>For details, see <a href="#">Configuring Parameters Rapidly</a>.</p>
spark.driver.extraClassPath	<p>Indicates the extra class path entries attached to the class path of the driver.</p> <p>Note: In client mode, this configuration cannot be set directly in the application using SparkConf because the driver JVM has been started. You can use <b>--driver-java-options</b> or the default property file to set the parameter.</p>	<p>For details, see <a href="#">Configuring Parameters Rapidly</a>.</p>
spark.driver.userClassPathFirst	<p>(Trial) Indicates whether to allow JAR files added by users to take precedence over Spark JAR files when classes are loaded in the driver. This feature can be used to mitigate conflicts between Spark dependencies and user dependencies. This feature is in the trial phase and is used only in cluster mode.</p>	false
spark.driver.extraLibraryPath	<p>Sets a special library path for starting the driver JVM.</p> <p>Note: In client mode, this configuration cannot be set directly in the application using SparkConf because the driver JVM has been started. You can use <b>--driver-java-options</b> or the default property file to set the parameter.</p>	<ul style="list-style-type: none"> <li>JDBCServer2x: \$ {SPARK_INSTALLED_HOME}/spark/native</li> <li>SparkResource2x: \$ {DATA_NODE_INSTANCE_HOME}/hadoop/lib/native</li> </ul>

Parameter	Description	Default Value
spark.driver.cores	Specifies the number of cores used by the driver process. This parameter is available only in cluster mode.	1
spark.driver.memory	Indicates the memory used by the driver process, that is, the memory used by the SparkContext initialization process (for example, 512 MB and 2 GB).  Note: In client mode, this configuration cannot be set directly in the application using SparkConf because the driver JVM has been started. You can use <b>--driver-java-options</b> or the default property file to set the parameter.	4G
spark.driver.maxResultSize	Indicates the total size of serialization results of all partitions for each Spark action operation (for example, collect). The value must be at least 1 MB. If this parameter is set to <b>0</b> , the size is not limited. If the total amount exceeds this limit, the task will be aborted. If the value is too large, the memory of the driver may be insufficient (depending on the object memory overhead of <b>spark.driver.memory</b> and JVM). Set a proper limit to ensure sufficient memory for the driver.	1G
spark.driver.host	Specifies the host name or IP address for the driver to listen on, which is used for the driver to communicate with the executor.	(local hostname)
spark.driver.port	Specifies the port for the driver to listen on, which is used for the driver to communicate with the executor.	(random)

## ExecutorLauncher Configuration

ExecutorLauncher exists only in Yarn-client mode. In Yarn-client mode, ExecutorLauncher and the driver are not in the same process. Therefore, you need to configure parameters for ExecutorLauncher.

**Table 24-13** Parameter description

Parameter	Description	Default Value
spark.yarn.am.extraJavaOptions	Indicates a string of extra JVM options to pass to the YARN ApplicationMaster in client mode. Use <b>spark.driver.extraJavaOptions</b> in cluster mode.	For details, see <a href="#">Configuring Parameters Rapidly</a> .
spark.yarn.am.memory	Indicates the amount of memory to use for the YARN ApplicationMaster in client mode, in the same format as JVM memory strings (for example, 512 MB or 2 GB). In cluster mode, use <b>spark.driver.memory</b> instead.	1G
spark.yarn.am.memoryOverhead	This parameter is the same as <b>spark.yarn.driver.memoryOverhead</b> . However, this parameter applies only to ApplicationMaster in client mode.	-
spark.yarn.am.cores	Indicates the number of cores to use for the YARN ApplicationMaster in client mode. Use <b>spark.driver.cores</b> in cluster mode.	1

## Executor Configuration

An executor is a Java process. However, unlike the driver and ApplicationMaster, an executor can have multiple processes. Spark supports only same configurations. That is, the process parameters of all executors must be the same.

**Table 24-14** Parameter description

Parameter	Description	Default Value
spark.executor.extraJavaOptions	Indicates extra JVM option passed to the executor, for example, GC setting and logging. Do not set Spark attributes or heap size using this option. Instead, set Spark attributes using the SparkConf object or the <b>spark-defaults.conf</b> file specified when the spark-submit script is called. Set heap size using <b>spark.executor.memory</b> .	For details, see <a href="#">Configuring Parameters Rapidly</a> .

Parameter	Description	Default Value
spark.executor.extraClassPath	Indicates the extra classpath attached to the executor classpath. This parameter ensures compatibility with historical versions of Spark. Generally, you do not need to set this parameter.	-
spark.executor.extraLibraryPath	Sets the special library path used when the executor JVM is started.	For details, see <a href="#">Configuring Parameters Rapidly</a> .
spark.executor.userClassPathFirst	(Trial) Same function as <b>spark.driver.userClassPathFirst</b> . However, this parameter applies to executor instances.	false
spark.executor.memory	Indicates the memory size used by each executor process. Its character string is in the same format as the JVM memory (example: 512 MB or 2 GB).	4G
spark.executorEnv. [EnvironmentVariableName]	Adds the environment variable specified by <b>EnvironmentVariableName</b> to the executor process. You can specify multiple environment variables.	-
spark.executor.logs.rolling.maxRetainedFiles	Sets the number of latest log files to be retained by the system during rolling. The old log files are deleted. This function is disabled by default.	-
spark.executor.logs.rolling.size.maxBytes	Sets the maximum size of the executor log file for rolling. This function is disabled by default. The value is in bytes. To automatically clear old logs, see <b>spark.executor.logs.rolling.maxRetainedFiles</b> .	-
spark.executor.logs.rolling.strategy	Sets the executor log rolling policy. Rolling is disabled by default. The value can be <b>time</b> (time-based rolling) or <b>size</b> (size-based rolling). If this parameter is set to <b>time</b> , the value of the <b>spark.executor.logs.rolling.time.interval</b> attribute is used as the log rolling interval. If this parameter is set to <b>size</b> , <b>spark.executor.logs.rolling.size.maxBytes</b> is used to set the maximum size of the file for rolling.	-

Parameter	Description	Default Value
spark.executor.logs.rolling.time.interval	Sets the time interval for executor log rolling. This function is disabled by default. The value can be <b>daily</b> , <b>hourly</b> , <b>minutely</b> , or any number of seconds. To automatically clear old logs, see <b>spark.executor.logs.rolling.maxRetainedFiles</b> .	daily

## WebUI

The Web UI displays the running process and status of the Spark application.

**Table 24-15** Parameter description

Parameter	Description	Default Value
spark.ui.killEnabled	Allows stages and jobs to be stopped on the web UI. <b>NOTE</b> For security purposes, the default value of this parameter is set to <b>false</b> to prevent misoperations. To enable this function, set this parameter to <b>true</b> in the <b>spark-defaults.conf</b> configuration file. Exercise caution when performing this operation.	true
spark.ui.port	Specifies the port for your application's dashboard, which displays memory and workload data.	<ul style="list-style-type: none"> <li>• JDBC Server2x: <b>4040</b></li> <li>• Spark Resource2x: 0</li> <li>• Index Server2x: 22901</li> </ul>
spark.ui.retainedJobs	Specifies the number of jobs recorded by the Spark UI and status API before GC.	1000
spark.ui.retainedStages	Specifies the number of stages recorded by the Spark UI and status API before GC.	1000

## HistoryServer

A History Server reads the **EventLog** file in the file system and displays the running status of the Spark application.

**Table 24-16** Parameter description

Parameter	Description	Default Value
spark.history.fs.logDirectory	Specifies the log directory of a History Server.	-
spark.history.ui.port	Specifies the port for JobHistory listening to connection.	18080
spark.history.fs.updateInterval	Specifies the update interval of the information displayed on a History Server, in seconds. Each update checks for changes made to the event logs in the persistent store.	10s
spark.history.fs.updateInterval.seconds	Specifies the interval for checking the update of each event log. This parameter has the same function as <b>spark.history.fs.updateInterval</b> . <b>spark.history.fs.updateInterval</b> is recommended.	10s
spark.history.updateInterval	This parameter has the same function as <b>spark.history.fs.updateInterval.seconds</b> and <b>spark.history.fs.updateInterval</b> . <b>spark.history.fs.updateInterval</b> is recommended.	10s

## History Server UI Timeout and Maximum Number of Access Times

**Table 24-17** Parameter description

Parameter	Description	Default Value
spark.session.maxAge	Specifies the session timeout interval, in seconds. This parameter applies only to the security mode. This parameter cannot be set in normal mode.	600
spark.connection.maxRequest	Specifies the maximum number of concurrent client access requests to JobHistory.	5000

## EventLog

During the running of Spark applications, the running status is written into the file system in JSON format in real time for the History Server service to read and reproduce the application running status.

**Table 24-18** Parameter description

Parameter	Description	Default Value
spark.eventLog.enabled	Indicates whether to log Spark events, which are used to reconstruct the web UI after the application execution is complete.	true
spark.eventLog.dir	Indicates the directory for logging Spark events if <b>spark.eventLog.enabled</b> is set to <b>true</b> . In this directory, Spark creates a subdirectory for each application and logs events of the application in the subdirectory. You can also set a unified address similar to the HDFS directory so that the History Server can read historical files.	hdfs://hacluster/spark2x/jobHistory2x
spark.eventLog.compress	Indicates whether to compress logged events when <b>spark.eventLog.enabled</b> is set to <b>true</b> .	false

## Periodic Clearing of Event Logs

Event logs on JobHistory increases with submitted tasks. Too many event log files exist as the number of submitted tasks increases. Spark provides the function for periodically clearing event logs. You can enable this function and set the clearing interval using related parameters.

**Table 24-19** Parameter description

Parameter	Description	Default Value
spark.history.fs.cleaner.enabled	Indicates whether to enable the clearing function.	true
spark.history.fs.cleaner.interval	Indicates the check interval of the clearing function.	1d
spark.history.fs.cleaner.maxAge	Indicates the maximum duration for storing logs.	4d

## Kryo

Kryo is a highly efficient Java serialization framework, which is integrated into Spark by default. Almost all Spark performance tuning requires the process of converting the default serializer of Spark into a Kryo serializer. Kryo serialization

supports only serialization at the Spark data layer. To configure Kryo serialization, set **spark.serializer** to **org.apache.spark.serializer.KryoSerializer** and configure the following parameters to optimize Kryo serialization performance:

**Table 24-20** Parameter description

Parameter	Description	Default Value
spark.kryo.classesToRegister	Specifies the name of the class that needs to be registered with Kryo when Kryo serialization is used. Multiple classes are separated by commas (,).	-
spark.kryo.referenceTracking	Indicates whether to trace the references to the same object when Kryo is used to serialize data. This function is applicable to the scenario where the object graph has circular references or the same object has multiple copies. Otherwise, you can disable this function to improve performance.	true
spark.kryo.registrationRequired	Indicates whether Kryo is used to register an object. When this parameter is set to <b>true</b> , an exception is thrown if an object that is not registered with Kryo is serialized. When it is set to <b>false</b> (default value), Kryo writes unregistered class names to the serialized object. This operation causes a large amount of performance overhead. Therefore, you need to enable this option before deleting a class from the registration queue.	false
spark.kryo.registrationRequired	If Kryo serialization is used, use Kryo to register the class with the custom class. Use this property if you need to register a class in a custom way, such as specifying a custom field serializer. Otherwise, use <b>spark.kryo.classesToRegister</b> , which is simpler. Set this parameter to a class that extends KryoRegistrar.	-
spark.kryo.serializer.buffer.max	Specifies the maximum size of the Kryo serialization buffer, in MB. The value must be greater than the object that attempts to be serialized. If the error "buffer limit exceeded" occurs in Kryo, increase the value of this parameter. You can also set this parameter by setting <b>spark.kryo.serializer.buffer.max</b> .	64MB



Parameter	Description	Default Value
spark.kryoserializer.buffer	Specifies the initial size of the Kryo serialization buffer, in MB. Each core of each worker has a buffer. If necessary, the buffer size will be increased to the value of <b>spark.kryoserializer.buffer.max</b> . You can also set this parameter by setting <b>spark.kryoserializer.buffer</b> .	64KB

## Broadcast

Broadcast is used to transmit data blocks between Spark processes. In Spark, broadcast can be used for JAR packages, files, closures, and returned results. Broadcast supports two modes: Torrent and HTTP. The Torrent mode divides data into small fragments and distributes them to clusters. Data can be obtained remotely if necessary. The HTTP mode saves files to the local disk and transfers the entire files to the remote end through HTTP if necessary. The former is more stable than the latter. Therefore, Torrent is the default broadcast mode.

**Table 24-21** Parameter description

Parameter	Description	Default Value
spark.broadcast.factory	Indicates the broadcast mode.	org.apache.spark.broadcast.TorrentBroadcastFactory
spark.broadcast.blockSize	Indicates the block size of <b>TorrentBroadcastFactory</b> . If the value is too large, the concurrency during broadcast is reduced (the speed is slow). If the value is too small, BlockManager performance may be affected.	4096
spark.broadcast.compress	Indicates whether to compress broadcast variables before sending them. You are advised to compress the broadcast variables.	true

## Storage

Spark features in-memory computing. Spark Storage is used to manage memory resources. Storage stores data blocks generated during RDD caching. The heap memory in the JVM acts as a whole. Therefore, **Storage Memory Size** is an important concept during Spark Storage management.

**Table 24-22** Parameter description

Parameter	Description	Default Value
spark.storage.memoryMapThreshold	Specifies the block size. If the size of a block exceeds the value of this parameter, Spark performs memory mapping for the disk file. This prevents Spark from mapping too small blocks during memory mapping. Generally, memory mapping for blocks whose page size is close to or less than that of the operating system has high overhead.	2m

## PORT

**Table 24-23** Parameter description

Parameter	Description	Default Value
spark.ui.port	Specifies the port for your application's dashboard, which displays memory and workload data.	<ul style="list-style-type: none"> <li>JDBC Server2x: <b>4040</b></li> <li>SparkResource2x: 0</li> </ul>
spark.blockManager.port	Specifies all ports listened by BlockManager. These ports are on both the driver and executor.	<b>Range of Random Ports</b>
spark.driver.port	Specifies the port for the driver to listen on, which is used for the driver to communicate with the executor.	<b>Range of Random Ports</b>

### Range of Random Ports

All random ports must be within a certain range.

**Table 24-24** Parameter description

Parameter	Description	Default Value
spark.random.port.min	Sets the minimum random port.	22600
spark.random.port.max	Sets the maximum random port.	22899

## TIMEOUT

By default, computation tasks that can well process medium-scale data are configured in Spark. However, if the data volume is too large, the tasks may fail due to timeout. In the scenario with a large amount of data, the timeout parameter in Spark needs to be assigned a larger value.

**Table 24-25** Parameter description

Parameter	Description	Default Value
spark.files.fetchTimeout	Specifies the communication timeout (in seconds) when fetching files added using <b>SparkContext.addFile()</b> of the driver.	60s
spark.network.timeout	Specifies the default timeout for all network interactions, in seconds. You can use this parameter to replace <b>spark.core.connection.ack.wait.timeout</b> , <b>spark.akka.timeout</b> , <b>spark.storage.blockManagerSlaveTimeoutMs</b> , or <b>spark.shuffle.io.connectionTimeout</b> .	360s
spark.core.connection.ack.wait.timeout	Specifies the timeout for a connection to wait for a response, in seconds. To avoid long-time waiting caused by GC, you can set this parameter to a larger value.	60

## Encryption

Spark supports SSL for Akka and HTTP (for the broadcast and file server) protocols, but does not support SSL for the web UI and block transfer service.

SSL must be configured on each node and configured for each component involved in communication using a particular protocol.

**Table 24-26** Parameter description

Parameter	Description	Default Value
spark.ssl.enabled	Indicates whether to enable SSL connections for all supported protocols.  All SSL settings similar to <b>spark.ssl.xxx</b> indicate the global configuration of all supported protocols. To override the global configuration of a particular protocol, you must override the property in the namespace specified by the protocol.  Use <b>spark.ssl.YYY.XXX</b> to overwrite the global configuration of the particular protocol specified by <b>YYY</b> . <b>YYY</b> can be either <b>akka</b> for Akka-based connections or <b>fs</b> for the broadcast and file server.	false
spark.ssl.enabledAlgorithms	Indicates the comma-separated list of passwords. The specified passwords must be supported by the JVM.	-
spark.ssl.keyPassword	Specifies the password of a private key in the keystore.	-
spark.ssl.keystore	Specifies the path of the keystore file. The path can be absolute or relative to the directory where the component is started.	-
spark.ssl.keystorePassword	Specifies the password of the keystore.	-
spark.ssl.protocol	Specifies the protocol name. This protocol must be supported by the JVM. The reference list of protocols is available on this page.	-
spark.ssl.trustStore	Specifies the path of the truststore file. The path can be absolute or relative to the directory where the component is started.	-
spark.ssl.trustStorePassword	Specifies the password of the truststore.	-

## Security

Spark supports shared key-based authentication. You can use **spark.authenticate** to configure authentication. This parameter controls whether the Spark communication protocol uses the shared key for authentication. This authentication is a basic handshake that ensures that both sides have the same shared key and are allowed to communicate. If the shared keys are different, the communication is not allowed. You can create shared keys as follows:

- For Spark on Yarn deployments, set **spark.authenticate** to **true**. Then, shared keys are automatically generated and distributed. Each application exclusively occupies a shared key.

- For other types of Spark deployments, configure Spark parameter **spark.authenticate.secret** on each node. All masters, workers, and applications use this key.

**Table 24-27** Parameter description

Parameter	Description	Default Value
spark.acls.enable	Indicates whether to enable Spark ACLs. If Spark ACLs are enabled, the system checks whether the user has the permission to access and modify jobs. Note that this requires the user to be identifiable. If the user is identified as invalid, the check will not be performed. Filters can be used to verify and set users on the UI.	true
spark.admin.acls	Specifies the comma-separated list of users/administrators that have the permissions to view and modify all Spark jobs. This list can be used if you are running on a shared cluster and working with the help of an administrator or developer.	admin
spark.authenticate	Indicates whether Spark authenticates its internal connections. If the application is not running on Yarn, see <b>spark.authenticate.secret</b> .	true
spark.authenticate.secret	Sets the key for authentication between Spark components. This parameter must be set if Spark does not run on Yarn and authentication is disabled.	-
spark.modify.acls	Specifies the comma-separated list of users who have the permission to modify Spark jobs. By default, only users who have enabled Spark jobs have the permission to modify the list (for example, delete the list).	-
spark.ui.view.acls	Specifies the comma-separated list of users who have the permission to access the Spark web UI. By default, only users who have enabled Spark jobs have the access permission.	-

## Enabling the Authentication Mechanism Between Spark Processes

Spark processes support shared key-based authentication. You can configure **spark.authenticate** to control whether Spark performs authentication during communication. In this authentication mode, the two communication parties share the same key only using simple handshakes.

Configure the following parameters in the **spark-defaults.conf** file on the Spark client.

**Table 24-28** Parameter description

Parameter	Description	Default Value
spark.authenticate	For Spark on Yarn deployments, set this parameter to <b>true</b> . Then, keys are automatically generated and distributed, and each application uses a unique key.	true

## Compression

Data compression is policy that optimizes memory usage at the expense of CPU. Therefore, when the Spark memory is severely insufficient (this issue is common due to the characteristics of in-memory computing), data compression can greatly improve performance. Spark supports three types of compression algorithm: Snappy, LZ4, and LZF. Snappy is the default compression algorithm and invokes the native method to compress and decompress data. In Yarn mode, pay attention to the impact of non-heap memory on the container process.

**Table 24-29** Parameter description

Parameter	Description	Default Value
spark.io.compression.codec	Indicates the codec for compressing internal data, such as RDD partitions, broadcast variables, and shuffle output. By default, Spark supports three types of compression algorithm: LZ4, LZF, and Snappy. You can specify algorithms using fully qualified class names, such as <b>org.apache.spark.io.LZ4CompressionCodec</b> , <b>org.apache.spark.io.LZFCompressionCodec</b> , and <b>org.apache.spark.io.SnappyCompressionCodec</b> .	lz4
spark.io.compression.lz4.block.size	Indicates the block size (bytes) used in LZ4 compression when the LZ4 compression algorithm is used. When LZ4 is used, reducing the block size also reduces the shuffle memory usage.	32768
spark.io.compression.snappy.block.size	Indicates the block size (bytes) used in Snappy compression when the Snappy compression algorithm is used. When Snappy is used, reducing the block size also reduces the shuffle memory usage.	32768
spark.shuffle.compress	Indicates whether to compress the output files of a Map task. You are advised to compress the broadcast variables. using <b>spark.io.compression.codec</b> .	true

Parameter	Description	Default Value
spark.shuffle.spill.compress	Indicates whether to compress the data overflowed during shuffle using <b>spark.io.compression.codec</b> .	true
spark.eventLog.compress	Indicates whether to compress logged events when <b>spark.eventLog.enabled</b> is set to <b>true</b> .	false
spark.broadcast.compress	Indicates whether to compress broadcast variables before sending them. You are advised to compress the broadcast variables.	true
spark.rdd.compress	Indicates whether to compress serialized RDD partitions (for example, the <b>StorageLevel.MEMORY_ONLY_SER</b> partition). Substantial space can be saved at the cost of some extra CPU time.	false

## Reducing the Probability of Abnormal Client Application Operations When Resources Are Insufficient

When resources are insufficient, ApplicationMaster tasks must wait and will not be processed until enough resources are available for use. If the actual waiting time exceeds the configured waiting time, the ApplicationMaster tasks will be deleted. Adjust the following parameters to reduce the probability of abnormal client application operation.

Configure the following parameters in the **spark-defaults.conf** file on the client.

**Table 24-30** Parameter description

Parameter	Description	Default Value
spark.yarn.applicationMaster.waitTries	Specifies the number of the times that ApplicationMaster waits for Spark master, which is also the times that ApplicationMaster waits for SparkContext initialization. Enlarge this parameter value to prevent ApplicationMaster tasks from being deleted and reduce the probability of abnormal client application operations.	10
spark.yarn.am.memory	Specifies the ApplicationMaster memory. Enlarge this parameter value to prevent ApplicationMaster tasks from being deleted by ResourceManager due to insufficient memory and reduce the probability of abnormal client application operations.	1G

## 24.2.4 Spark on HBase Overview and Basic Applications

### Scenario

Spark on HBase allows users to query HBase tables in Spark SQL and to store data for HBase tables by using the Beeline tool. You can use HBase APIs to create, read data from, and insert data into tables.

### Procedure

- Step 1** Log in to Manager and choose **Cluster** > *Name of the desired cluster* > **Cluster Properties** to check whether the cluster is in security mode.
- If yes, go to [Step 2](#).
  - If no, go to [Step 5](#).
- Step 2** Choose **Cluster** > *Name of the desired cluster* > **Service** > **Spark2x** > **Configuration** > **All Configurations** > **JDBCServer2x** > **Default**, and modify the following parameter.

**Table 24-31** Parameter list 1

Parameter	Default Value	Changed To
spark.yarn.security.credentials.hbase.enabled	false	true

#### NOTE

To ensure that Spark2x can access HBase for a long time, do not modify the following parameters of the HBase and HDFS services:

- dfs.namenode.delegation.token.renew-interval
- dfs.namenode.delegation.token.max-lifetime
- hbase.auth.key.update.interval
- hbase.auth.token.max.lifetime (The value is fixed to **604800000** ms, that is, 7 days.)

If the preceding parameter configuration must be modified based on service requirements, ensure that the value of the HDFS parameter **dfs.namenode.delegation.token.renew-interval** is not greater than the values of the HBase parameters **hbase.auth.key.update.interval**, **hbase.auth.token.max.lifetime**, and **dfs.namenode.delegation.token.max-lifetime**.

- Step 3** Choose **SparkResource2x** > **Default** and modify the following parameters.

**Table 24-32** Parameter list 2

Parameter	Default Value	Changed To
spark.yarn.security.credentials.hbase.enabled	false	true



**Step 4** Restart the Spark2x service for the configuration to take effect.

 **NOTE**

To use the Spark on HBase function on the Spark2x client, you need to download and install the Spark2x client again.

**Step 5** On the Spark2x client, use the spark-sql or spark-beeline connection to query tables created by Hive on HBase. You can create an HBase table by running SQL commands or create an external table to associate the HBase table. Before creating tables, ensure that HBase tables exist in HBase. The HBase table **table1** is used as an example.

1. Run the following commands to create the HBase table using the Beeline tool:

```
create table hbaseTable
(
 id string,
 name string,
 age int
)
using org.apache.spark.sql.hbase.HBaseSource
options(
 hbaseTableName "table1",
 keyCols "id",
 colsMapping "
name=cf1.cq1,
age=cf1.cq2
");
```

 **NOTE**

- **hbaseTable**: name of the created Spark table
  - **id string, name string, age int**: field name and field type of the Spark table
  - **table1**: name of the HBase table
  - *id*: row key column name of the HBase table
  - *name=cf1.cq1, age=cf1.cq2*: mapping between columns in the Spark table and columns in the HBase table. The **name** column of the Spark table maps the **cq1** column in the **cf1** column family of the HBase table, and the **age** column of the Spark table maps the **cq2** column in the **cf1** column family of the HBase table.
2. Run the following command to import data to the HBase table using a CSV file:

```
hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -
Dimporttsv.separator="," -
Dimporttsv.columns=HBASE_ROW_KEY,cf1:cq1,cf1:cq2,cf1:cq3,cf1:cq4,cf1:cq5
table1 /hperson
```

Where **table1** indicates the name of the HBase table, and **/hperson** indicates the path where the CSV file is stored.

- Run the following command to query data in spark-sql or spark-beeline, where *hbaseTable* is the corresponding Spark table name: The command is as follows:

```
select * from hbaseTable;
```

----End

## 24.2.5 Spark on HBase V2 Overview and Basic Applications

### Scenario

Spark on HBase V2 allows users to query HBase tables in Spark SQL and to store data for HBase tables by using the Beeline tool. You can use HBase APIs to create, read data from, and insert data into tables.

### Procedure

- Log in to Manager and choose **Cluster** > *Name of the desired cluster* > **Cluster Properties** to check whether the cluster is in security mode.
  - If yes, go to [Step 2](#).
  - If no, go to [Step 5](#).
- Choose **Cluster** > *Name of the desired cluster* > **Service** > **Spark2x** > **Configuration** > **All Configurations** > **JDBCServer2x** > **Default**, and modify the following parameter.

**Table 24-33** Parameter list 1

Parameter	Default Value	Changed To
spark.yarn.security.credentials.hbase.enabled	false	true

#### NOTE

To ensure that Spark2x can access HBase for a long time, do not modify the following parameters of the HBase and HDFS services:

- dfs.namenode.delegation.token.renew-interval
- dfs.namenode.delegation.token.max-lifetime
- hbase.auth.key.update.interval
- hbase.auth.token.max.lifetime (The value is fixed to **604800000** ms, that is, 7 days.)

If the preceding parameter configuration must be modified based on service requirements, ensure that the value of the HDFS parameter **dfs.namenode.delegation.token.renew-interval** is not greater than the values of the HBase parameters **hbase.auth.key.update.interval**, **hbase.auth.token.max.lifetime**, and **dfs.namenode.delegation.token.max-lifetime**.

- Choose **SparkResource2x** > **Default** and modify the following parameters.

**Table 24-34** Parameter list 2

Parameter	Default Value	Changed To
spark.yarn.security.credentials.hbase.enabled	false	true

**Step 4** Restart the Spark2x service for the configuration to take effect.

 **NOTE**

If you need to use the Spark on HBase function on the Spark2x client, download and install the Spark2x client again.

**Step 5** On the Spark2x client, use the spark-sql or spark-beeline connection to query tables created by Hive on HBase. You can create an HBase table by running SQL commands or create an external table to associate the HBase table. For details, see the following description. The following uses the HBase table **table1** as an example.

1. Run the following commands to create a table using the spark-beeline tool:

```
create table hbaseTable1
(id string, name string, age int)
using org.apache.spark.sql.hbase.HBaseSourceV2
options(
hbaseTableName "table2",
keyCols "id",
colsMapping "name=cf1.cq1,age=cf1.cq2");
```

 **NOTE**

- **hbaseTable1**: name of the created Spark table
- **id string, name string, age int**: field name and field type of the Spark table
- **table2**: name of the HBase table
- **id**: row key column name of the HBase table
- **name=cf1.cq1, age=cf1.cq2**: mapping between columns in the Spark table and columns in the HBase table. The **name** column of the Spark table maps the **cq1** column in the **cf1** column family of the HBase table, and the **age** column of the Spark table maps the **cq2** column in the **cf1** column family of the HBase table.

2. Run the following command to import data to the HBase table using a CSV file:

```
hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -
Dimporttsv.separator="," -
Dimporttsv.columns=HBASE_ROW_KEY,cf1:cq1,cf1:cq2,cf1:cq3,cf1:cq4,cf1:cq5
table2 /hperson
```

Where **table2** indicates the name of the HBase table, and **/hperson** indicates the path where the CSV file is stored.

3. Run the following command to query data in spark-sql or spark-beeline. **hbaseTable1** indicates the corresponding Spark table name.

```
select * from hbaseTable1;
```

----End

## 24.2.6 SparkSQL Permission Management(Security Mode)

### 24.2.6.1 Spark SQL Permissions

#### SparkSQL Permissions

Similar to Hive, Spark SQL is a data warehouse framework built on Hadoop, providing storage of structured data like structured query language (SQL).

MRS supports users, user groups, and roles. Permission must be assigned to roles and then roles are bound to users or user groups. Users can obtain permissions only by binding a role or joining a group that is bound with a role.

#### NOTE

- If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. For details, see [Adding a Ranger Access Permission Policy for Spark2x](#).
- After Ranger authentication is enabled or disabled on Spark2x, you need to restart Spark2x and download the client again or update the client configuration file `spark/conf/spark-defaults.conf`.

Enable Ranger authentication: `spark.ranger.plugin.authorization.enable=true`

Disable Ranger authentication: `spark.ranger.plugin.authorization.enable=false`

#### Permission Management

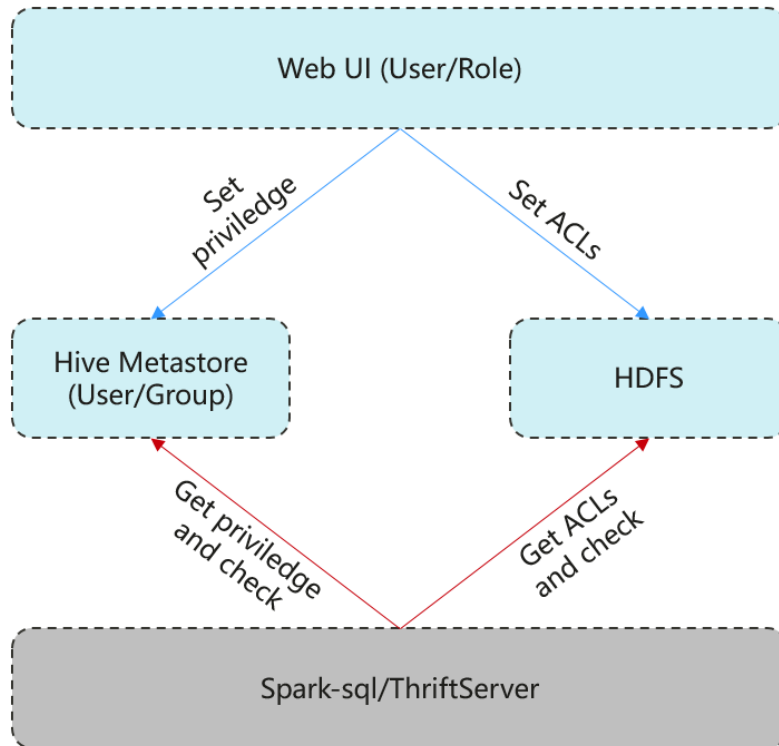
Spark SQL permission management indicates the permission system for managing and controlling users' operations on databases, to ensure that different users can operate databases separately and securely. A user can operate another user's tables and databases only with the corresponding permissions. Otherwise, operations will be rejected.

Spark SQL permission management integrates the functions of Hive management. The MetaStore service of Hive and the permission granting function on the page are required to enable Spark SQL permission management.

**Figure 24-3** shows the basic architecture of SparkSQL permission management. This architecture includes two parts: granting permissions on the page, and obtaining and judging a service.

- Granting permissions on the page: Spark SQL only supports granting permissions on the page. On FusionInsight Manager, choose **System > Permission** to add or delete a user, user group, or a role, and to grant permissions or cancel permissions.
- Obtaining and judging a service: When the DDL and DML commands are received from a client, Spark SQL will obtain the client's permissions on database information from MetaStore, and check whether the required permissions are included. If the required permissions are included, continue the execution. If the required permissions are not included, reject the user's operations. After the MetaStore permissions are checked, ACL permission also needs to be checked on HDFS.

Figure 24-3 Spark SQL permission management architecture



Additionally, Spark SQL provides column and view permissions to meet requirements of different scenarios.

- Column permission
    - Spark SQL permission control consists of metadata permission control and HDFS ACL permission control. When Hive MetaStore automatically synchronizes table permissions to the HDFS ACL, column-level permissions are not synchronized. In other words, a user with partial or all column-level permissions cannot access the entire HDFS file using the HDFS client.
      - In **spark-sql** mode, users with only column-level permissions cannot access HDFS files. Therefore, they cannot access the columns of the corresponding tables.
      - In Beeline/JDBCServer mode, permissions are assigned among users, for example, the permissions on the table created by user A are assigned to user B.
        - **hive.server2.enable.doAs=true** (configured in the **hive-site.xml** file on the Spark server)  
In this case, user B cannot query the information. You need to manually assign the read permission on the file in HDFS.
        - **hive.server2.enable.doAs=false**
          - Users A and B are connected by Beeline. User B can query the information.
          - User A creates a table using SQL statements, and user B can query the table in Beeline.
- However, information query is not supported in other scenarios, for example, user A uses Beeline to create a table and user B uses SQL

to query the table, or user A uses SQL to create a table and user B uses SQL to query the table. You need to manually assign the read permission on the file in HDFS.

 **NOTE**

The **spark** user is an administrator in HDFS ACL permission control. The permission control of the Beeline client user depends only on the metadata permission on Spark.

- **View permission**

View permission indicates the operation permission such as query and modification on the view of a table, regardless of the corresponding permission of a table. Namely, if you have the permission to query the view of a table, the permission to query the table is not mandatory. The view permission is applicable to the whole table but not to the columns.

Restrictions of view and column permissions on SparkSQL are similar. The following uses the view permission as an example:

- In spark-sql mode, if you have only the view permission but not the table permission and do not have the permission to read HDFS, you cannot access the table data stored in HDFS. That is, you cannot query the view of the table.
- In Beeline/JDBCServer mode, permissions are assigned among users, for example, the permissions on the view created by user A are assigned to user B.

- **hive.server2.enable.doAs=true** (configured in the **hive-site.xml** file on the Spark server)

In this case, user B cannot query the information. You need to manually assign the read permission on the file in HDFS.

- **hive.server2.enable.doAs=false**

- Users A and B are connected by Beeline. User B can query the information.
- User A creates a view using SQL statements, and user B can query the view in Beeline.

However, information query is not supported in other scenarios. For example, user A uses Beeline to create a view but user B cannot use SQL to query the view, or user A uses SQL to create a view but user B cannot use SQL to query the view. You need to manually assign the read permission on the file in HDFS.

Permission of operations on the view of a table is as follows:

- To create a view, you must have the CREATE permission on the database and the SELECT and SELECT\_of\_GRANT permissions on the tables.
- Creating and describing a view only entail the SELECT permission on the view. Querying views and tables at the same time entails the SELECT permission on other tables. For example, to perform **select \* from v1 join t1**, you must have the SELECT permission on the **v1** view and **t1** table, even through the **v1** view depends on the **t1** table.

 NOTE

In Beeline/JDBCServer mode, to query a view, you must have the SELECT permission on the tables. In spark-sql mode, to query a view, you must have the SELECT permission on the view and tables.

- Deleting and modifying a view entail the permission of owner on the view.

## SparkSQL Permission Model

If you want to perform SQL operations using SparkSQL, you must be granted with permissions of SparkSQL databases and tables (include external tables and views). The complete permission model of SparkSQL consists of the meta data permission and HDFS file permission. Permissions required to use a database or a table is just one type of SparkSQL permission.

- Metadata permissions

Metadata permissions are controlled at the metadata layer. Similar to traditional relational databases, SparkSQL databases involve the CREATE and SELECT permissions, and tables and columns involve the SELECT, INSERT, UPDATE, and DELETE permissions. SparkSQL also supports the permissions of **OWNERSHIP** and **ADMIN**.

- Data file permissions (that is, HDFS file permissions)

SparkSQL database and table files are stored in HDFS. The created databases or tables are saved in the **/user/hive/warehouse** directory of HDFS by default. The system automatically creates subdirectories named after database names and database table names. To access a database or table, you must have the **Read**, **Write** and **Execute** permissions on the corresponding file in HDFS.

To perform various operations on SparkSQL databases or tables, you need to associate the metadata permission and HDFS file permission. For example, to query SparkSQL data tables, you need to associate the metadata permission **SELECT** and HDFS file permissions **Read** and **Execute**.

Using the management function of Manager GUI to manage the permissions of SparkSQL databases and tables, only requires the configuration of metadata permission, and the system will automatically associate and configure the HDFS file permission. In this way, operations on the interface are simplified, and the efficiency is improved.

## Usage Scenarios and Related Permissions

Creating a database with SparkSQL service requires users to join in the hive group, without granting a role. Users have all permissions on the databases or tables created by themselves in Hive or HDFS. They can create tables, select, delete, insert, or update data, and grant permissions to other users to allow them to access the tables and corresponding HDFS directories and files.

A user can access the tables or database only with permissions. Users' permissions vary depending on different SparkSQL scenarios.

**Table 24-35** SparkSQL scenarios

Typical Scenario	Required Permission
Using SparkSQL tables, columns, or databases	Permissions required in different scenarios are as follows: <ul style="list-style-type: none"> <li>To create a table, the CREATE permission is required.</li> <li>To query data, the SELECT permission is required.</li> <li>To insert data, the INSERT permission is required.</li> </ul>
Associating and using other components	In some scenarios, except the SparkSQL permission, other permissions may be also required. For example: Using Spark on HBase to query HBase data in SparkSQL requires HBase permissions.

In some special SparkSQL scenarios, other permissions must be configured separately.

**Table 24-36** SparkSQL scenarios and required permissions

Scenario	Required Permission
Creating SparkSQL databases, tables, and external tables, or adding partitions to created Hive tables or external tables when data files specified by Hive users are saved to other HDFS directories except <b>/user/hive/warehouse</b>	<ul style="list-style-type: none"> <li>The directory must exist, the client user must be the owner of the directory, and the user must have the <b>Read</b>, <b>Write</b>, and <b>Execute</b> permissions on the directory. The user must have the <b>Read</b> and <b>Execute</b> permissions of all the upper-layer directories of the directory.</li> <li>If the Spark version is later than 2, the <b>Create</b> permission of the Hive database is required if you want to create a HBase table. However, in Spark 1.5, the <b>Create</b> permissions of both the Hive database and HBase namespace are required if you want to create a HBase table.</li> </ul>



Scenario	Required Permission
Importing all the files or specified files in a specified directory to the table using load	<ul style="list-style-type: none"> <li>The data source is a Linux local disk, the specified directory exists, and the system user <b>omm</b> has read and execute permission of the directory and all its upper-layer directories. The specified file exists, and user <b>omm</b> has the <b>Read</b> permission on the file and has the <b>Read</b> and <b>Execute</b> permissions on all the upper-layer directories of the file.</li> <li>The data source is HDFS, the specified directory exists, and the SparkSQL user is the owner of the directory and has the <b>Read, Write, and Execute</b> permissions on the directory and its subdirectories, and has the <b>Read</b> and <b>Execute</b> permissions on all its upper-layer directories. The specified file exists, and the SparkSQL user is the owner of the file and has the <b>Read, Write, and Execute</b> permissions on the file and has the <b>Read</b> and <b>Execute</b> permissions on all its upper-layer directories.</li> </ul>
Creating or deleting functions or modifying any database	The <b>ADMIN</b> permission is required.
Performing operations on all databases and tables in Hive	The user must be added to the <b>supergroup</b> user group, and be assigned the <b>ADMIN</b> permission.
After assigning the <b>Insert</b> permission on some DataSource tables, assigning the <b>Write</b> permission on table directories in HDFS before performing the insert or analyze operation	When the <b>Insert</b> permission is assigned to the <b>spark datasource</b> table, if the table format is text, CSV, JSON, Parquet, or ORC, the permission on the table directory is not changed. After the <b>Insert</b> permission is assigned to the DataSource table of the preceding formats, you need to assign the <b>Write</b> permission to the table directories in HDFS separately so that users can perform the insert or analyze operation on the tables.

### 24.2.6.2 Creating a Spark SQL Role

#### Scenario

This section describes how to create and configure a SparkSQL role on Manager as the MRS cluster administrator. The Spark SQL role can be configured with the administrator permission or the permission of performing operations on the table data.

Creating a database with Hive requires users to join in the **hive** group, without granting a role. Users have all permissions on the databases or tables created by themselves in Hive or HDFS. They can create tables, select, delete, insert, or update data, and grant permissions to other users to allow them to access the

tables and corresponding HDFS directories and files. The created databases or tables are saved in the `/user/hive/warehouse` directory of HDFS by default.

 **NOTE**

- If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. For details, see [Adding a Ranger Access Permission Policy for Spark2x](#).
- After Ranger authentication is enabled or disabled on Spark2x, you need to restart Spark2x and download the client again or update the client configuration file `spark/conf/spark-defaults.conf`.

Enable Ranger authentication: `spark.ranger.plugin.authorization.enable=true`

Disable Ranger authentication: `spark.ranger.plugin.authorization.enable=false`

## Procedure

1. Log in to Manager, and choose **System > Permission > Role**.
2. Click **Create Role** and set a role name and enter description.
3. Set **Configure Resource Permission**. For details, see [Table 24-37](#).
  - **Hive Admin Privilege**: Hive administrator permissions.
  - **Hive Read Write Privileges**: Hive data table management permission, which is the operation permission to set and manage the data of created tables.

 **NOTE**

- Hive role management supports the administrator permission, and the permissions of accessing tables and views, without granting the database permission.
- The permissions of the Hive administrator do not include the permission to manage HDFS.
- If there are too many tables in the database or too many files in tables, the permission granting may last a while. For example, if a table contains 10,000 files, the permission granting lasts about 2 minutes.

**Table 24-37** Setting a role

Task	Operation
Hive administrator permission	<p>In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> &gt; <b>Hive</b> and select <b>Hive Admin Privilege</b>.</p> <p>After being bound to the Hive administrator role, perform the following operations during each maintenance operation:</p> <ol style="list-style-type: none"> <li>1. Log in to the node where the Spark2x client is installed as the client installation user.</li> <li>2. Run the following command to configure environment variables: For example, if the Spark2x client installation directory is <code>/opt/client</code>, run <b>source /opt/client/bigdata_env</b>. <b>source /opt/client/Spark2x/component_env</b></li> <li>3. Run the following command to perform user authentication: <b>kinit Hive service user</b></li> <li>4. Run the following command to log in to the client tool: <b>/opt/client/Spark2x/spark/bin/beeline -u "jdbc:hive2://&lt;zkNode1_IP&gt;:&lt;zkNode1_Port&gt;,&lt;zkNode2_IP&gt;:&lt;zkNode2_Port&gt;,&lt;zkNode3_IP&gt;:&lt;zkNode3_Port&gt;/;serviceDiscovery-Mode=zooKeeper;zooKeeperNamespace=sparkthriftserver2x;user.principal=spark2x/hadoop.&lt;system domain name&gt;@&lt;system domain name&gt;;sasLQop=auth-conf;auth=KERBEROS;principal=spark2x/hadoop.&lt;system domain name&gt;@&lt;system domain name&gt;;"</b></li> </ol>

Task	Operation
	<p><b>NOTE</b></p> <ul style="list-style-type: none"> <li>• <code>&lt;zkNode1_IP&gt;:&lt;zkNode1_Port&gt;</code>, <code>&lt;zkNode2_IP&gt;:&lt;zkNode2_Port&gt;</code>, <code>&lt;zkNode3_IP&gt;:&lt;zkNode3_Port&gt;</code> indicates the ZooKeeper URL, for example, 192.168.81.37:2181,192.168.195.232:2181,192.168.169.84:2181.</li> <li>• <code>sparkthriftserver</code> indicates a ZooKeeper directory, from which a random TriftServer or ProxyThriftServer is connected by the client.</li> <li>• You can log in to Manager, choose <b>System &gt; Permission &gt; Domain and Mutual Trust</b>, and view the value of <b>Local Domain</b>, which is the current system domain name. <code>spark2x/hadoop.&lt;System domain name&gt;</code> is the username. All letters in the system domain name contained in the username are lowercase letters. For example, <b>Local Domain</b> is set to <code>9427068F-6EFA-4833-B43E-60CB641E5B6C.COM</code>, and the username is <code>spark2x/hadoo.9427068f-6efa-4833-b43e-60cb641e5b6c.com</code>.</li> </ul> <p>5. Run the following command to update the administrator permissions: <b>set role admin;</b></p>
<p>Setting the permission to query a table of another user in the default database</p>	<ol style="list-style-type: none"> <li>1. In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> &gt; <b>Hive</b> &gt; <b>Hive Read Write Privileges</b>.</li> <li>2. Click the name of the specified database in the database list. Tables in the database are displayed.</li> <li>3. In the <b>Permission</b> column of the specified table, select <b>SELECT</b>.</li> </ol>
<p>Setting the permission to import data to a table of another user in the default database</p>	<ol style="list-style-type: none"> <li>1. In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> &gt; <b>Hive</b> &gt; <b>Hive Read Write Privileges</b>.</li> <li>2. Click the name of the specified database in the database list. Tables in the database are displayed.</li> <li>3. In the <b>Permission</b> column of the specified table, select <b>DELETE</b> and <b>INSERT</b>.</li> </ol>

4. Click **OK**.

### 24.2.6.3 Configuring Permissions for SparkSQL Tables, Columns, and Databases

#### Scenario

You can configure related permissions if you need to access tables or databases created by other users. SparkSQL supports column-based permission control. If a user needs to access some columns in tables created by other users, the user must be granted the permission for columns. The following describes how to grant table, column, and database permissions to users by using the role management function of Manager.

#### Procedure

The operations for granting permissions on SparkSQL tables, columns, and databases are the same as those for Hive. For details, see [Permission Management](#).

#### NOTE

- Any permission for a table in the database is automatically associated with the HDFS permission for the database directory to facilitate permission management. When any permission for a table is canceled, the system does not automatically cancel the HDFS permission for the database directory to ensure performance. In this case, users can only log in to the database and view table names.
- When the query permission on a database is added to or deleted from a role, the query permission on tables in the database is automatically added to or deleted from the role. This mechanism is inherited from Hive.
- In Spark, the column name of the struct data type cannot contain special characters, that is, characters other than letters, digits, and underscores (\_). If the column name of the struct data type contains special characters, the column cannot be displayed on the FusionInsight Manager console when you grant permissions to roles on the role page.

#### Concepts

SparkSQL statements are processed in SparkSQL. [Table 24-38](#) describes the permission requirements.

**Table 24-38** Scenarios of using SparkSQL tables, columns, or databases

Scenario	Required Permission
CREATE TABLE	<b>CREATE</b> , RWX+ownership (for creating external tables - the location) <b>NOTE</b> When creating datasource tables in a specified file path, the RWX and ownership permission on the file next to the path is required.
DROP TABLE	<b>Ownership</b> (of table)
DROP TABLE PROPERTIES	<b>Ownership</b>
DESCRIBE TABLE	<b>Select</b>

Scenario	Required Permission
SHOW PARTITIONS	<b>Select</b>
ALTER TABLE LOCATION	<b>Ownership</b> , RWX+ownership (for new location)
ALTER PARTITION LOCATION	<b>Ownership</b> , RWX+ownership (for new partition location)
ALTER TABLE ADD PARTITION	<b>Insert</b> , RWX and ownership (for partition location)
ALTER TABLE DROP PARTITION	<b>Delete</b>
ALTER TABLE(all of them except the ones above)	<b>Update, Ownership</b>
TRUNCATE TABLE	<b>Ownership</b>
CREATE VIEW	<b>Select, Grant Of Select, CREATE</b>
ALTER VIEW PROPERTIES	<b>Ownership</b>
ALTER VIEW RENAME	<b>Ownership</b>
ALTER VIEW ADD PARTS	<b>Ownership</b>
ALTER VIEW AS	<b>Ownership</b>
ALTER VIEW DROPPARTS	<b>Ownership</b>
ANALYZE TABLE	<b>Search, Insert</b>
SHOW COLUMNS	<b>Select</b>
SHOW TABLE PROPERTIES	<b>Select</b>
CREATE TABLE AS SELECT	<b>Select, CREATE</b>
SELECT	<b>Select</b> <b>NOTE</b> The same as tables, you need to have the <b>Select</b> permission on a view when performing a SELECT operation on the view.
INSERT	<b>Insert, Delete (for overwrite)</b>
LOAD	<b>Insert, Delete</b> , RWX+ownership(input location)
SHOW CREATE TABLE	<b>Select, Grant Of Select</b>
CREATE FUNCTION	<b>ADMIN</b>
DROP FUNCTION	<b>ADMIN</b>
DESC FUNCTION	-
SHOW FUNCTIONS	-

Scenario	Required Permission
MSCK (metastore check)	<b>Ownership</b>
ALTER DATABASE	<b>ADMIN</b>
CREATE DATABASE	-
SHOW DATABASES	-
EXPLAIN	<b>Select</b>
DROP DATABASE	<b>Ownership</b>
DESC DATABASE	-
CACHE TABLE	<b>Select</b>
UNCACHE TABLE	<b>Select</b>
CLEAR CACHE TABLE	<b>ADMIN</b>
REFRESH TABLE	<b>Select</b>
ADD FILE	<b>ADMIN</b>
ADD JAR	<b>ADMIN</b>
HEALTHCHECK	-

## 24.2.6.4 Configuring Permissions for SparkSQL to Use Other Components

### Scenario

SparkSQL may need to be associated with other components. For example, Spark on HBase requires HBase permissions. The following describes how to associate SparkSQL with HBase.

### Prerequisites

- The Spark client has been installed. For example, the installation directory is **/opt/client**.
- You have obtained a user account with the MRS cluster administrator permissions, such as **admin**.

### Procedure

- **Spark on HBase authorization**  
After the permissions are assigned, you can use statements that are similar to SQL statements to access HBase tables from SparkSQL. The following uses the procedure for assigning a user the permissions to query HBase tables as an example.

#### NOTE

Set `spark.yarn.security.credentials.hbase.enabled` to `true`.

- a. On Manager, create a role, for example, **hive\_hbase\_create**, and grant the permission to create HBase tables to the role.

In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **HBase** > **HBase Scope** > **global**. Select **create** of the namespace **default**, and click **OK**.

 **NOTE**

In this example, the created table is saved in the default database of Hive and has the CREATE permission of the default database. If you save the table to a Hive database other than **default**, perform the following operations:

In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **Hive** > **Hive Read Write Privileges**, select **CREATE** for the desired database, and click **OK**.

- b. On Manager, create a role, for example, **hive\_hbase\_submit**, and grant the permission to submit tasks to the Yarn queue.

In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **Yarn** > **Scheduling Queue** > **root**. Select **Submit** of **default**, and click **OK**.

- c. On Manager, create a human-machine user, for example, **hbase\_creates\_user**, add the user to the **hive** group, and bind the **hive\_hbase\_create** and **hive\_hbase\_submit** roles to create SparkSQL and HBase tables.

- d. Log in to the node where the client is installed as the client installation user.

- e. Run the following command to configure environment variables:

```
source /opt/client/bigdata_env
source /opt/client/Spark2x/component_env
```

- f. Run the following command to authenticate the user:

```
kinit hbase_creates_user
```

- g. Run the following commands to enter the shell environment on the Spark JDBCServer client:

```
/opt/client/Spark2x/spark/bin/beeline -u "jdbc:hive2://
<zkNode1_IP>:<zkNode1_Port>,<zkNode2_IP>:<zkNode2_Port>,<zkNode3_IP>:<zkNode3_Port>";serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=sparkthriftserver2x;user.principal=spark2x/hadoop.<system domain name>@<system domain name>;sasLQop=auth-conf;auth=KERBEROS;principal=spark2x/hadoop.<system domain name>@<system domain name>;"
```

- h. Run the following command to create a table in SparkSQL and HBase, for example, create the **hbaseTable** table:

```
create table hbaseTable (id string, name string, age int) using
org.apache.spark.sql.hbase.HBaseSource options (hbaseTableName
"table1", keyCols "id", colsMapping = "", name=cf1.cq1, age=cf1.cq2");
```

The created SparkSQL table and the HBase table are stored in the Hive database **default** and the HBase namespace **default**, respectively.

- i. On Manager, create a role, for example, **hive\_hbase\_select**, and grant the role the permission to query SparkSQL on HBase table **hbaseTable** and HBase table **hbaseTable**.



- In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **HBase** > **HBase Scope** > **global** > **default**. Select **read** for the **hbaseTable** table, and click **OK** to grant the table query permission to the HBase role.
- Edit the role. In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **HBase** > **HBase Scope** > **global** > **hbase**. Select **Execute** for **hbase:meta**, and click **OK**.
- Edit the role. In the **Configure Resource Permission** table, choose *Name of the desired cluster* > **Hive** > **Hive Read Write Privileges** > **default**. Select **SELECT** for the **hbaseTable** table, and click **OK**.
- j. On Manager, create a human-machine user, for example, **hbase\_select\_user**, add the user to the **hive** group, and bind the **hive\_hbase\_select** role to the user for querying SparkSQL and HBase tables.
- k. Run the following command to configure environment variables:  
**source /opt/client/bigdata\_env**  
**source /opt/client/Spark2x/component\_env**
- l. Run the following command to authenticate users:  
**kinit hbase\_select\_user**
- m. Run the following commands to enter the shell environment on the Spark JDBCServer client:  
**/opt/client/Spark2x/spark/bin/beeline -u "jdbc:hive2://**  
**<zkNode1\_IP>:<zkNode1\_Port>,<zkNode2\_IP>:<zkNode2\_Port>,<zkNode3\_IP>:<zkNode3\_Port>;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=sparkthriftserver2x;user.principal=spark2x/hadoop.<system domain name>@<system domain name>;sasLQop=auth-conf;auth=KERBEROS;principal=spark2x/hadoop.<system domain name>@<system domain name>;"**
- n. Run the following command to use a SparkSQL statement to query HBase table data:  
**select \* from thh;**

### 24.2.6.5 Configuring the Client and Server

This section describes how to configure SparkSQL permission management functions (client configuration is similar to server configuration). To enable table permission, add following configurations on the client and server:

- **spark-defaults.conf** configuration file

**Table 24-39** Parameter description (1)

Parameter	Description	Default Value
spark.sql.authorization.enabled	Specifies whether to enable permission authentication of the datasource statement. It is recommended that the parameter value be set to <b>true</b> to enable permission authentication.	true

- **hive-site.xml** configuration file

**Table 24-40** Parameter description (2)

Parameter	Description	Default Value
hive.metastore.uris	Specifies the MetaStore service address of the Hive component, for example, <b>thrift://10.10.169.84:21088,thrift://10.10.81.37:21088</b> .	-
hive.metastore.sasl.enabled	Specifies whether the MetaStore service uses SASL to improve security. The table permission function must be enabled.	true
hive.metastore.kerberos.principal	Specifies the principal of the MetaStore service in the Hive component, for example, <b>hive/hadoop.&lt;system domain name&gt;@&lt;system domain name&gt;</b> .	hive-metastore/_HOST@EXAMPLE.COM
hive.metastore.thrift.sasl.qop	After the SparkSQL permission management function is enabled, set the parameter to <b>auth-conf</b> .	auth-conf
hive.metastore.token.signature	Specifies the token identifier of the MetaStore service, which is set to <b>HiveServer2ImpersonationToken</b> .	HiveServer2ImpersonationToken
hive.security.authentication.manager	Specifies the manager authenticated by the Hive client, which is set to <b>org.apache.hadoop.hive.ql.security.SessionStateUserGroupAuthenticator</b> .	org.apache.hadoop.hive.ql.security.SessionStateUserMSGroupAuthenticator
hive.security.authorization.enabled	Specifies whether to enable client authentication, which is set to <b>true</b> .	true

Parameter	Description	Default Value
hive.security.authorization.createtable.owner.grants	Specifies which permissions are granted to the owner who creates the table, which is set to <b>ALL</b> .	ALL

- **core-site.xml** configuration file of the MetaStore service

**Table 24-41** Parameter description (3)

Parameter	Description	Default Value
hadoop.proxyuser.spark.hosts	Specifies the hosts from which Spark users can be masqueraded, which is set to *, indicating all hosts.	-
hadoop.proxyuser.spark.groups	Specifies the user groups from which Spark users can be masqueraded, which is set to *, indicating all user groups.	-

## 24.2.7 Scenario-Specific Configuration

### 24.2.7.1 Configuring Multi-active Instance Mode

#### Scenarios

In this mode, multiple ThriftServers coexist in the cluster and the client can randomly connect any ThriftServer to perform service operations. When one or multiple ThriftServers stop working, a client can connect to another functional ThriftServer.

#### Configuration Description

Log in to Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x** > **Configurations**, click **All Configurations**, and search for and modify the following parameters.

**Table 24-42** Parameter description

Parameter	Description	Default Value
spark.thriftserver.zookeeper.connection.timeout	Specifies the timeout interval of connection between ZooKeeper client and ThriftServer. The unit is millisecond.	60000

Parameter	Description	Default Value
spark.thriftserver.zookeeper.session.timeout	Specifies the timeout interval of a ZooKeeper client session. The unit is millisecond.	90000
spark.thriftserver.zookeeper.retry.times	Specifies the retry times after ZooKeeper disconnection.	3
spark.yarn.queue	Specifies the Yarn queue where the JDBCServer service resides.	default

## 24.2.7.2 Configuring the Multi-tenant Mode

### Scenarios

In multi-tenant mode, JDBCServer are bound with tenants. Each tenant corresponds to one or more JDBCServer, and a JDBCServer provides services for only one tenant. Different tenants can be configured with different Yarn queues to implement resource isolation.

### Configuration Description

Log in to Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x** > **Configurations**, click **All Configurations**, and search for and modify the following parameters.

**Table 24-43** Parameter description

Parameter	Description	Default Value
spark.proxyserver.hash.enabled	Specifies whether to connect to ProxyServer using the Hash algorithm. <ul style="list-style-type: none"> <li><b>true</b> indicates using the Hash algorithm. In multi-tenant mode, this parameter must be configured to <b>true</b>.</li> <li><b>false</b> indicates using random connection. In multi-active instance mode, this parameter must be configured to <b>false</b>.</li> </ul>	true <b>NOTE</b> After this parameter is modified, you need to download the client again.
spark.thriftserver.proxy.enabled	Specifies whether to use the multi-tenant mode. <ul style="list-style-type: none"> <li><b>false</b>: The multi-instance mode is used.</li> <li><b>true</b>: The multi-tenant mode is used.</li> </ul>	true

Parameter	Description	Default Value
spark.thriftserver.proxy.maxThriftServerPerTenancy	Specifies the maximum number of JDBCServer instances that can be started by a tenant in multi-tenant mode.	1
spark.thriftserver.proxy.maxSessionPerThriftServer	Specifies the maximum number of sessions in a single JDBCServer instance in multi-tenant mode. If the number of sessions exceeds this value and the number of JDBCServer instances does not exceed the upper limit, a new JDBCServer instance is started. Otherwise, an alarm log is output.	50
spark.thriftserver.proxy.sessionWaitTime	Specifies the wait time before a JDBCServer instance is stopped when it has no session connections in multi-tenant mode.	180000
spark.thriftserver.proxy.sessionThreshold	In multi-tenant mode, when the session usage (formula: number of current sessions / spark.thriftserver.proxy.maxSessionPerThriftServer x number of current JDBCServer instances) of the JDBCServer instance reaches the threshold, a new JDBCServer instance is automatically added.	100
spark.thriftserver.proxy.healthcheck.period	Specifies the period of JDBCServer health checks conducted by the JDBCServer proxy in multi-tenant mode.	60000
spark.thriftserver.proxy.healthcheck.recheckTimes	Specifies the number of JDBCServer health check retries conducted by the JDBCServer proxy in multi-tenant mode.	3
spark.thriftserver.proxy.healthcheck.waitTime	Specifies the wait time for JDBCServer to respond to a health check request sent by the JDBCServer proxy.	10000
spark.thriftserver.proxy.session.check.interval	Specifies the period of JDBCServer proxy sessions in multi-tenant mode.	6h
spark.thriftserver.proxy.idle.session.timeout	Specifies the idle time interval of a JDBCServer proxy session in multi-tenant mode. If no operation is performed within this period, the session is closed.	7d

Parameter	Description	Default Value
spark.thriftserver.proxy.idle.session.check.operation	Specifies whether to check that operations still exist on a JDBCServer proxy session when the session is checked for expiration in multi-tenant mode.	true
spark.thriftserver.proxy.idle.operation.timeout	Specifies the timeout interval of an operation in multi-tenant mode. An operation that times out is closed.	5d

### 24.2.7.3 Configuring the Switchover Between the Multi-active Instance Mode and the Multi-tenant Mode

#### Scenarios

When using a cluster, if you want to switch between multi-active instance mode and multi-tenant mode, the following configurations are required.

- Switch from multi-tenant mode to multi-active instance mode.  
Modify the following parameters of the Spark2x service:
  - spark.thriftserver.proxy.enabled=false
  - spark.scheduler.allocation.file=#{conf\_dir}/fairscheduler.xml
  - spark.proxyserver.hash.enabled=false
- Switch from multi-active instance mode to multi-tenant mode.  
Modify the following parameters of the Spark2x service:
  - spark.thriftserver.proxy.enabled=true
  - spark.scheduler.allocation.file=./\_\_spark\_conf\_\_/\_\_hadoop\_conf\_\_/fairscheduler.xml
  - spark.proxyserver.hash.enabled=true

#### Configuration Description

Log in to Manager, choose **Cluster** > *Name of the desired cluster* > **Service** > **Spark2x** > **Configuration**, click **All Configurations**, and search for and modify the following parameters.

**Table 24-44** Parameter description

Parameter	Description	Default Value
spark.thriftserver.proxy.enabled	Specifies whether to use the multi-tenant mode. <ul style="list-style-type: none"> <li>• <b>false</b>: The multi-instance mode is used.</li> <li>• <b>true</b>: The multi-tenant mode is used.</li> </ul>	true
spark.scheduler.allocation.file	Specifies the fair scheduling file path. <ul style="list-style-type: none"> <li>• If the multi-active instance mode is used, the path is changed to <b>#{conf_dir}/fairscheduler.xml</b>.</li> <li>• If multi-tenant mode is used, the path is changed to <b>./__spark_conf__/_hadoop_conf_/fairscheduler.xml</b>.</li> </ul>	./__spark_conf__/_hadoop_conf_/fairscheduler.xml
spark.proxyserver.hash.enabled	Specifies whether to connect to ProxyServer using the Hash algorithm. <ul style="list-style-type: none"> <li>• <b>true</b> indicates using the Hash algorithm. In multi-tenant mode, this parameter must be configured to <b>true</b>.</li> <li>• <b>false</b> indicates using random connection. In multi-active instance mode, this parameter must be configured to <b>false</b>.</li> </ul>	true <b>NOTE</b> After this parameter is modified, you need to download the client again.

## 24.2.7.4 Configuring the Size of the Event Queue

### Scenarios

Functions such as UI, EventLog, and dynamic resource scheduling in Spark are implemented through event transfer. Events include SparkListenerJobStart and SparkListenerJobEnd, which record each important process.

Each event is saved to a queue after it occurs. When creating a SparkContext object, Driver starts a thread to obtain an event from the queue in sequence and sends the event to each Listener. Each Listener processes the event after detecting the event.

Therefore, when the queuing speed is faster than the read speed, the queue overflows. As a result, the overflow event is lost, affecting the UI, EventLog, and dynamic resource scheduling functions. Therefore, a configuration item is added for more flexible use. You can set a proper value based on the memory size of the driver.

## Configuration Description

### Navigation path for setting parameters:

Before executing an application, modify the Spark service configuration. On Manager, choose **Cluster** > *Name of the desired cluster* > **Service** > **Spark2x** > **Configuration** and click **All Configurations**. Enter a parameter name in the search box.

**Table 24-45** Parameter description

Parameter	Description	Default Value
spark.scheduler.listenerbus.eventqueue.capacity	Specifies the size of the event queue. Configure this parameter based on the memory of the driver.	100000 0

### NOTE

If the following information is displayed in the Driver log, the queue overflows.

1. Common application:  
Dropping SparkListenerEvent because no remaining room in event queue.  
This likely means one of the SparkListeners is too slow and cannot keep up with the rate at which tasks are being started by the scheduler.
2. Spark Streaming application:  
Dropping StreamingListenerEvent because no remaining room in event queue.  
This likely means one of the StreamingListeners is too slow and cannot keep up with the rate at which events are being started by the scheduler.

## 24.2.7.5 Configuring Executor Off-Heap Memory

### Scenario

When the executor off-heap memory is too small, or processes with higher priority preempt resources, the physical memory usage will exceed the maximal value. To prevent the physical memory usage from exceeding, set the following parameter.

### Configuration

#### Navigation path for setting parameters:

When submitting an application, set the following parameter using **--conf** or adjust the parameter in the **spark-defaults.conf** configuration file on the client.



**Table 24-46** Parameter description

Parameter	Description	Default Value
spark.executor.memoryOverhead	Indicates the off-heap memory of each executor, in MB. Increasing the value of this parameter prevents the physical memory usage from exceeding the maximal value. The value is calculated based on $\max(384, \text{Executor - Memory} \times 0.1)$ . The minimal value is 384.	1024

## 24.2.7.6 Enhancing Stability in a Limited Memory Condition

### Scenario

A large amount of memory is required when Spark SQL executes a query, especially during Aggregate and Join operations. If the memory is limited, OutOfMemoryError may occur. Stability in a limited memory condition ensures queries to be run in limited memory without OutOfMemoryError.

#### NOTE

Limited memory does not mean infinitely small memory, but ensures stable queries by using disks in a scenario where memory fails to store the data amount that is several times larger than the available memory size. For example, for queries involving Join, the data of the same key used for Join needs to be stored in memory. If the data amount is too large to be stored in the available memory, OutOfMemoryError occurs.

Stability in a limited memory condition involves the following sub-functions:

1. ExternalSort  
If the memory is inadequate during sorting, partial data overflows to disks.
2. TungstenAggregate  
By default, ExternalSort is used to sort data before data aggregation. Therefore, if the memory is inadequate, the data overflows to disks during sorting. The data has been properly sorted before aggregation and only aggregation results of the current key are remained, which use a small amount of memory.
3. SortMergeJoin and SortMergeOuterJoin  
SortMergeJoin and SortMergeOuterJoin are based on the equivalence join of sorted data. By default, ExternalSort is used to sort the data before the equivalence join. Therefore, if the memory is inadequate, the data overflows to disks during sorting. The data has been properly sorted before the equivalence join and only the data of the same key are remained, which uses a small amount of memory.

### Configuration

**Navigation path for setting parameters:**

When submitting an application, set the following parameters using `--conf` or adjust the parameters in the `spark-defaults.conf` configuration file on the client.

**Table 24-47** Parameter description

Parameter	Scenario	Description	Default Value
spark.sql.tungsten.enabled	/	Type: Boolean <ul style="list-style-type: none"> <li>If the value is <b>true</b>, tungsten is enabled. That is, the logic plan is equivalent to the codegeneration function, and the physical plan uses the corresponding tungsten execution plan.</li> <li>If the value is <b>false</b>, tungsten is disabled.</li> </ul>	true
spark.sql.codegen.wholeStage		Type: Boolean <ul style="list-style-type: none"> <li>If the value is <b>true</b>, codegeneration is enabled. That is, for some specified queries, the logic plan code will be generated dynamically when running.</li> <li>If the value is <b>false</b>, codegeneration is disabled and the existing static code is used.</li> </ul>	true

 **NOTE**

- To enable ExternalSort, you need to set `spark.sql.planner.externalSort` to **true** and `spark.sql.unsafe.enabled` to **false** or `spark.sql.codegen.wholeStage` to **false**.
- To enable TungstenAggregate, use either of the following methods:  
Set `spark.sql.codegen.wholeStage` and `spark.sql.unsafe.enabled` to **true** in the configuration file or CLI.  
If neither `spark.sql.codegen.wholeStage` nor `spark.sql.unsafe.enabled` is **true** or either of them is **true**, TungstenAggregate is enabled as long as `spark.sql.tungsten.enabled` is set to **true**.

### 24.2.7.7 Viewing Aggregated Container Logs on the Web UI

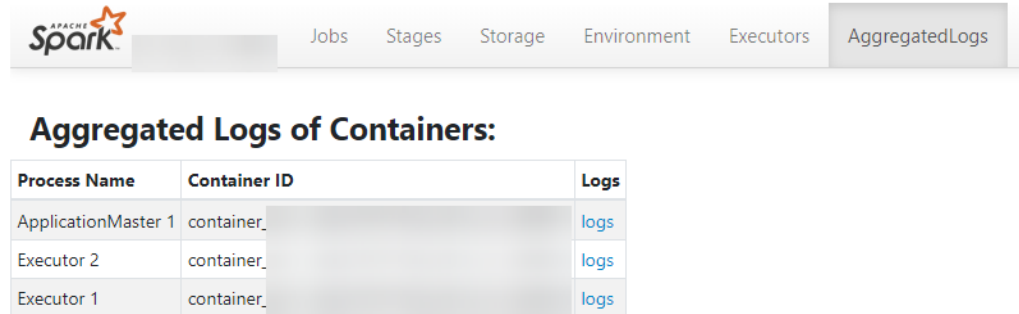
#### Scenarios

When `yarn.log-aggregation-enable` of Yarn is set to **true**, the container log aggregation function is enabled. Log aggregation indicates that after applications are run on Yarn, NodeManager aggregates all container logs of the node to HDFS and deletes local logs. For details, see [Configuring Container Log Aggregation](#).

However, all logs will be aggregated to an HDFS directory and can only be viewed by accessing an HDFS file. Open-source Spark and Yarn do not support the function of viewing aggregated logs on the web UI.

Spark supports this function. As shown in [Figure 24-4](#), the **AggregatedLogs** tab is added to the HistoryServer page. You can click **logs** to view aggregated logs.

**Figure 24-4** Log aggregation page



## Configuration Description

To display logs on the web UI, aggregated logs need to be parsed and presented. Spark parses aggregation logs using JobHistoryServer of Hadoop. Therefore, you can use the **spark.jobhistory.address** parameter to specify the URL of the JobHistoryServer page to parse and present the logs.

### Navigation path for setting parameters:

When submitting an application, set these parameters using **--conf** or adjust the following parameter in the **spark-defaults.conf** configuration file on the client.

#### NOTE

- This function depends on JobHistoryServer of Hadoop. Therefore, ensure that JobHistoryServer is running properly before using the log aggregation function.
- If the parameter value is empty, the **AggregatedLogs** tab page still exists, but you cannot view logs by clicking **logs**.
- The aggregated container logs can be viewed only when the application is running and event log files of the application exist on HDFS.
- You can click the log link on the **Executors** page to view the logs of a running task. After the task completes, the logs are aggregated to HDFS, and the log link on the **Executors** page becomes invalid. In this case, you can click **logs** on the **AggregatedLogs** page to view the aggregated logs.

**Table 24-48** Parameter description

Parameter	Description	Default Value
spark.jobhistory.address	<p>URL of the JobHistoryServer page. The format is <i>http(s)://ip:port/jobhistory</i>. For example, <b>https://10.92.115.1:26014/jobhistory</b>.</p> <p>The default value is empty, indicating that container aggregation logs cannot be viewed on the web UI.</p> <p>Restart the service for the configuration to take effect.</p>	-

### 24.2.7.8 Configuring Environment Variables in Yarn-Client and Yarn-Cluster Modes

#### Scenario

Values of some configuration parameters of Spark client vary depending on its work mode (YARN-Client or YARN-Cluster). If you switch Spark client between different modes without first changing values of such configuration parameters, Spark client fails to submit jobs in the new mode.

To avoid this, configure parameters as described in [Table 24-49](#).

- In Yarn-Cluster mode, use the new parameters (path and parameters of Spark server).
- In Yarn-Client mode, uses the original parameters.  
They are **spark.driver.extraClassPath**, **spark.driver.extraJavaOptions**, and **spark.driver.extraLibraryPath**.

#### NOTE

If you choose not to add the parameters in [Table 24-49](#), Spark client can continue to operate well in either mode but the mode switch requires changes to some of its configuration parameters.

### Configuration Parameters

#### Navigation path for setting parameters:

On Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x** > **Configurations**. Click **All Configurations** and enter a parameter name in the search box.

**Table 24-49** Parameter description

Parameter	Description	Default Value
spark.yarn.cluster.driver.extraClassPath	<p>Indicates the extraClassPath of the driver in Yarn-cluster mode. Set the parameter to the path and parameters of the server.</p> <p>The original parameter <b>spark.driver.extraClassPath</b> indicates the extraClassPath of Spark client. By using different parameters to separate the settings of Spark server from the settings of Spark client, you can switch Spark client to different modes without changing parameter values.</p>	<p><code>\${BIGDATA_HOME}/common/runtime/security</code></p>
spark.yarn.cluster.driver.extraJavaOptions	<p>Indicates the extraJavaOptions of Driver in Yarn-Cluster mode and is set to path and parameters of extraJavaOptions of Spark server.</p> <p>The original parameter <b>spark.driver.extraJavaOptions</b> indicates the path of extraJavaOptions of Spark client. By using different parameters to separate the settings of Spark server from the settings of Spark client, you can switch Spark client to different modes without changing parameter values.</p>	<p><code>-Xloggc:&lt;LOG_DIR&gt;/indexserver-%p-gc.log -XX:+PrintGCDetails -XX:-OmitStackTracelnFastThrow -XX:+PrintGCTimeStamps -XX:+PrintGCDateStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=20 -XX:GCLogFileSize=10M -Dlog4j.configuration=./__spark_conf__/__hadoop_conf__/log4j-executor.properties -Dlog4j.configuration.watch=true -Djava.security.auth.login.config=./__spark_conf__/__hadoop_conf__/jaas-zk.conf -Dzookeeper.server.principal=\${ZOOKEEPER_SERVER_PRINCIPAL} -Djava.security.krb5.conf=./__spark_conf__/__hadoop_conf__/kdc.conf -Djetty.version=x.y.z -Dorg.xerial.snappy.tmpdir=\${BIGDATA_HOME}/tmp -Dcarbon.properties.filepath=./__spark_conf__/__hadoop_conf__/carbon.properties -Djdk.tls.ephemeralDHKeySize=2048 -Dspark.ssl.keyStore=./child.keystore #{java_stack_prefer}</code></p>

## 24.2.7.9 Configuring the Default Number of Data Blocks Divided by SparkSQL

### Scenarios

By default, SparkSQL divides data into 200 data blocks during shuffle. In data-intensive scenarios, each data block may have excessive size. If a single data block of a task is larger than 2 GB, an error similar to the following will be reported while Spark attempts to fetch the data block:

```
Adjusted frame length exceeds 2147483647: 2717729270 - discarded
```

For example, setting the number of default data blocks to 200 causes SparkSQL to encounter an error in running a TPCDS 500-GB test. To avoid this, increase the number of default blocks in data-intensive scenarios.

### Configuration parameters

#### Navigation path for setting parameters:

On Manager, choose **Cluster > Name of the desired cluster > Service > Spark2x > Configuration** and click **All Configurations**. Enter a parameter name in the search box.

**Table 24-50** Parameter description

Parameter	Description	Default Value
spark.sql.shuffle.partitions	Indicates the default number of blocks divided during shuffle.	200

## 24.2.7.10 Configuring the Compression Format of a Parquet Table

### Scenarios

The compression format of a Parquet table can be configured as follows:

1. If the Parquet table is a partitioned one, set the **parquet.compression** parameter of the Parquet table to specify the compression format. For example, set **tblproperties** in the table creation statement:  
**"parquet.compression"="snappy"**.
2. If the Parquet table is a non-partitioned one, set the **spark.sql.parquet.compression.codec** parameter to specify the compression format. The configuration of the **parquet.compression** parameter is invalid, because the value of the **spark.sql.parquet.compression.codec** parameter is read by the **parquet.compression** parameter. If the **spark.sql.parquet.compression.codec** parameter is not configured, the default value is **snappy** and will be read by the **parquet.compression** parameter.

Therefore, the `spark.sql.parquet.compression.codec` parameter can only be used to set the compression format of a non-partitioned Parquet table.

## Configuration parameters

### Navigation path for setting parameters:

On Manager, choose **Cluster** > *Name of the desired cluster* > **Service** > **Spark2x** > **Configuration**. Click **All Configurations** and enter a parameter name in the search box.

**Table 24-51** Parameter description

Parameter	Description	Default Value
spark.sql.parquet.compression.codec	Used to set the compression format of a non-partitioned Parquet table.	snappy

### 24.2.7.11 Configuring the Number of Lost Executors Displayed in WebUI

#### Scenario

In Spark WebUI, the **Executor** page can display information about Lost Executor. Executors are dynamically recycled. If the JDBCServer tasks are large, there may be too many lost executors displayed in WebUI. Therefore, the number of displayed lost executors can be configured.

#### Procedure

Configure the following parameter in the `spark-defaults.conf` file on Spark client.

**Table 24-52** Parameter description

Parameter	Description	Default Value
spark.ui.retainedDeadExecutors	The maximum number of Lost Executors displayed in Spark WebUI.	100

### 24.2.7.12 Setting the Log Level Dynamically

#### Scenarios

In some scenarios, to locate problems or check information by changing the log level,

you can add the `-Dlog4j.configuration.watch=true` parameter to the JVM parameter of a process before the process is started. After the process is started,

you can modify the log4j configuration file corresponding to the process to change the log level.

The following processes support the dynamic setting of log levels: driver, executor, ApplicationMaster, JobHistory and JDBCServer.

Allowed log levels are as follows: FATAL, ERROR, WARN, INFO, DEBUG, TRACE, and ALL.

## Configuration Description

Add the following parameters to the JVM parameter corresponding to a process.

**Table 24-53** Parameter description

Parameter	Description	Default Value
- Dlog4j.configuration.watcher	Indicates a JVM parameter of a process. If this parameter is set to <b>true</b> , the dynamic configuration of log levels is enabled.	Left blank, indicating that the dynamic configuration of log levels is disabled

**Table 24-54** lists the JVM parameters of the driver, executor, and ApplicationMaster processes. Configure the following parameters in the **spark-defaults.conf** file on the Spark client. Set the log levels of the driver, executor, and ApplicationMaster processes in the log4j configuration file specified by the - **Dlog4j.configuration** parameter.

**Table 24-54** JVM parameters of processes (1)

Parameter	Description	Default Log Level
spark.driver.extraJavaOptions	Indicates the JVM parameter of the driver process.	INFO
spark.executor.extraJavaOptions	Indicates the JVM parameter of the executor process.	INFO
spark.yarn.am.extraJavaOptions	Indicates the JVM parameter of the ApplicationMaster process.	INFO

**Table 24-55** describes the JVM parameters of JobHistory Server and JDBCServer. Set the parameters in the **ENV\_VARS** configuration file. Set the log levels of JobHistory Server and JDBCServer in the **log4j.properties** configuration file.



**Table 24-55** JVM parameters of processes (2)

Parameter	Description	Default Log Level
GC_OPTS	Indicates the JVM parameter of the JobHistory Server process.	INFO
SPARK_SUBMIT_OPTS	Indicates the JVM parameter of JDBCServer.	INFO

**Example:**

To change the log level of the executor process to DEBUG dynamically, modify the **spark.executor.extraJavaOptions** JVM parameter of the executor process in the **spark-defaults.conf** file and run the following command to add the following configuration before the process is started:

```
-Dlog4j.configuration.watch=true
```

After the user application is submitted, change the log level in the log4j configuration file (for example, **-Dlog4j.configuration=file:\${BIGDATA\_HOME}/FusionInsight\_Spark2x\_8.0.2.1/install/FusionInsight-Spark2x-2.4.5/spark/conf/log4j-executor.properties**) specified by the **-Dlog4j.configuration** parameter in **spark.executor.extraJavaOptions** to DEBUG:

```
log4j.rootCategory=DEBUG, sparklog
```

It takes several seconds for the DEBUG level to take effect.

### 24.2.7.13 Configuring Whether Spark Obtains HBase Tokens

#### Scenario

When Spark is used to submit tasks, the driver obtains tokens from HBase by default. To access HBase, you need to configure the **jaas.conf** file for security authentication. If the **jaas.conf** file is not configured, the application will fail to run.

Therefore, perform the following operations based on whether the application involves HBase:

- If the application does not involve HBase, you do not need to obtain the HBase tokens. In this case, set **spark.yarn.security.credentials.hbase.enabled** to **false**.
- If the application involves HBase, set **spark.yarn.security.credentials.hbase.enabled** to **true** and configure the **jaas.conf** file on the driver as follows:

```
{client}/spark/bin/spark-sql --master yarn-client --principal {principal} --keytab {keytab} --driver-java-options "-Djava.security.auth.login.config={LocalPath}/jaas.conf"
```

Specify Keytab and Principal in the **jaas.conf** file. The following is an example:

```
Client {
 com.sun.security.auth.module.Krb5LoginModule required
 useKeyTab=true
 keyTab = "{LocalPath}/user.keytab"
```

```
principal="super@<System domain name>"
useTicketCache=false
debug=false;
};
```

## Configuration

Configure the following parameter in the **spark-defaults.conf** file of the Spark client.

**Table 24-56** Parameter description

Parameter	Description	Default Value
spark.yarn.security.credentials.hbase.enabled	Indicates whether HBase obtains a token. <ul style="list-style-type: none"><li>● <b>true</b>: HBase obtains a token.</li><li>● <b>false</b>: HBase does not obtain a token.</li></ul>	false

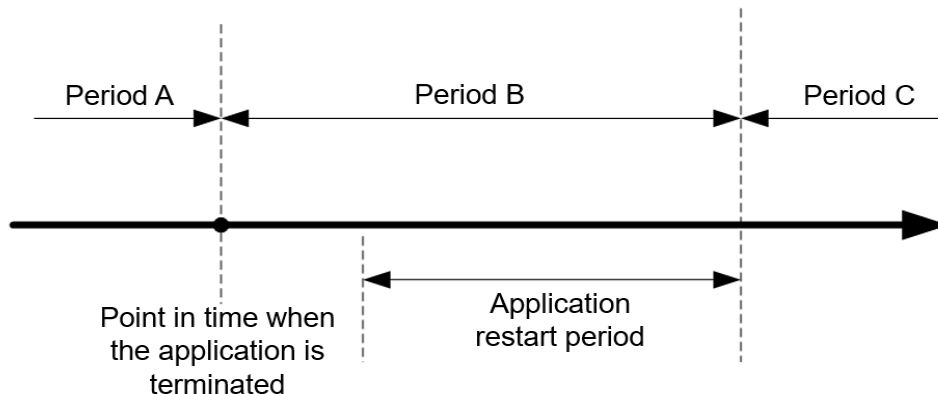
### 24.2.7.14 Configuring LIFO for Kafka

#### Scenario

If the Spark Streaming application is connected to Kafka, after the Spark Streaming application is terminated abnormally and restarted from the checkpoint, the system preferentially processes the tasks that are not completed before the application is terminated (Period A) and the tasks generated based on data that enters Kafka during the period (Period B) from the application termination to the restart. Then the application processes the tasks generated based on data that enters Kafka after the application is restarted (Period C). For data that enters Kafka in period B, Spark generates a corresponding number of tasks based on the end time (**batch** time). The first task reads all data, but other tasks may not read data. As a result, the task processing pressure is uneven.

If the tasks in Period A and Period B are processed slowly, the processing of tasks in period C is affected. To cope with the preceding scenario, Spark provides the last-in first-out (LIFO) function for Kafka.

**Figure 24-5** Time axis for restarting the Spark Streaming application



After this function is enabled, Spark preferentially schedules tasks in Period C. If there are multiple tasks in Period C, Spark schedules and executes the tasks in the sequence of task generation. Then Spark executes the tasks in Periods A and B. For data that enters Kafka in Period B, Spark generates tasks based on the end time and evenly distributes all data that enters Kafka in this period to each task to avoid uneven task processing pressure.

Constraints:

- This function applies only to the direct mode of Spark Streaming, and the execution result does not depend on the processing result of the previous batch (that is, stateless operation, for example, **updatestatebykey**). Multiple data input streams must be comparatively independent from each other. Otherwise, the result may change after the data is divided.
- The Kafka LIFO function can be enabled only when the application is connected to the Kafka input source.
- If both Kafka LIFO and flow control functions are enabled when the application is submitted, the flow control function is not enabled for the data that enters Kafka in Period B to ensure that the task scheduling priority for reading the data is the lowest. Flow control is enabled for the tasks in Period C after the application is restarted.

## Configuration

Configure the following parameters in the **spark-defaults.conf** file on the Spark driver.

**Table 24-57** Parameter description

Parameter	Description	Default Value
spark.streaming.kafka.direct.lifo	Specifies whether to enable the LIFO function of Kafka.	false

Parameter	Description	Default Value
spark.streaming.kafka010.inputstream.class	Obtains the decoupled class on FusionInsight.	org.apache.spark.streaming.kafka010.HWDirectKafkaInputDStream

### 24.2.7.15 Configuring Reliability for Connected Kafka

#### Scenario

When the Spark Streaming application is connected to Kafka and the application is restarted, the application reads data from Kafka based on the last read topic offset and the latest offset of the current topic.

If the leader of a Kafka topic fails and the offset of the Kafka leader is greatly different from that of the Kafka follower, the Kafka follower and leader are switched over after the Kafka service is restarted. As a result, the offset of the topic decreases after the Kafka service is restarted.

- If the Spark Streaming application keeps running, the start position for reading Kafka data is greater than the end position because the offset of the topic in Kafka decreases. As a result, the application cannot read data from Kafka and reports an error.
- Before restarting the Kafka service, stop the Spark Streaming application. After the Kafka service is restarted, restart the Spark Streaming application to restore the application from the checkpoint. In this case, the Spark Streaming application records the offset position read before the termination and uses the position as the reference to read subsequent data. The Kafka offset decreases (for example, from 100,000 to 10,000). Spark Streaming consumes data only after the offset of the Kafka leader increases to 100,000. As a result, the newly sent data whose offset is between 10,000 and 100,000 is lost.

To resolve the preceding problem, you can configure reliability for Kafka connected to Spark Streaming. After the reliability function of connected Kafka is enabled:

- If the offset of a topic in Kafka decreases when the Spark Streaming application is running, the latest offset of the topic in Kafka is used as the start position for reading Kafka data and subsequent data is read.

For a task that has been generated but has not been scheduled, if the read Kafka offset is greater than the latest offset of the topic in Kafka, the task fails to be executed.

#### NOTE

If a large number of tasks fail, the Executor is added to the blacklist. As a result, subsequent tasks cannot be deployed and run. If this happens, you can set **spark.blacklist.enabled** to disable the blacklist function. The blacklist function is enabled by default.

- If the offset of a topic in Kafka decreases, the Spark Streaming application restarts to restore the unfinished tasks. If the read Kafka offset range is greater than the latest offset of the topic in Kafka, the task is directly discarded.

 **NOTE**

If the state function is used in the Spark Streaming application, do not enable the reliability function of connected Kafka.

## Configuration

Configure the following parameter in the **spark-defaults.conf** file of the Spark client.

**Table 24-58** Parameter description

Parameter	Description	Default Value
spark.streaming.Kafka.reliability	Indicates whether to enable the reliability function for Kafka connected to Spark Streaming. <ul style="list-style-type: none"><li>• <b>true</b>: The reliability function is enabled.</li><li>• <b>false</b>: The reliability function is disabled.</li></ul>	false

### 24.2.7.16 Configuring Streaming Reading of Driver Execution Results

#### Scenario

When a query statement is executed, the returned result may be large (containing more than 100,000 records). In this case, JDBCServer out of memory (OOM) may occur. Therefore, the data aggregation function is provided to avoid OOM without sacrificing the performance.

#### Configuration

Two data aggregation function configuration parameters are provided. The two parameters are set in the **tunning** option on the Spark JDBCServer server. After the setting is complete, restart JDBCServer.

**Table 24-59** Parameter description

Parameter	Description	Default Value
spark.sql.bigdata.thriftServer.useHdfsCollect	<p>Indicates whether to save result data to HDFS instead of the memory.</p> <p>Advantages: The query result is stored in HDFS. Therefore, JDBCServer OOM does not occur.</p> <p>Disadvantages: The query is slow.</p> <ul style="list-style-type: none"> <li>● <b>true</b>: Result data is saved to HDFS.</li> <li>● <b>false</b>: This function is disabled.</li> </ul> <p><b>NOTICE</b> When <b>spark.sql.bigdata.thriftServer.useHdfsCollect</b> is set to <b>true</b>, result data is saved to HDFS. However, the job description on the native JobHistory page cannot be associated with the corresponding SQL statement. In addition, the execution ID in the spark-beeline command output is null. To solve the JDBCServer OOM problem and ensure correct information display, you are advised to set <b>spark.sql.userlocalFileCollect</b>.</p>	false
spark.sql.userlocalFileCollect	<p>Indicates whether to save result data to the local disk instead of memory.</p> <p>Advantages: In the case of small data volume, the performance loss can be ignored compared with the data storage mode using the native memory. In the case of large data volume (hundreds of millions of data records), the performance is much better than that when data is stored in the HDFS and native memory.</p> <p>Disadvantages: Optimization is required. In the case of large data volume, it is recommended that the JDBCServer driver memory be 10 GB and each core of the executor be allocated with 3 GB memory.</p> <ul style="list-style-type: none"> <li>● <b>true</b>: This function is enabled.</li> <li>● <b>false</b>: This function is disabled.</li> </ul>	false

Parameter	Description	Default Value
spark.sql.collect.Hive	<p>This parameter is valid only when <b>spark.sql.uselocalFileCollect</b> is set to <b>true</b>. It indicates whether to save the result data to a disk in direct serialization mode or in indirect serialization mode.</p> <p>Advantage: For queries of tables with a large number of partitions, the aggregation performance of the query results is better than that of the storage mode that query results are directly stored on the disk.</p> <p>Disadvantages: The disadvantages are the same as those when <b>spark.sql.uselocalFileCollect</b> is enabled.</p> <ul style="list-style-type: none"> <li>• <b>true</b>: This function is enabled.</li> <li>• <b>false</b>: This function is disabled.</li> </ul>	false
spark.sql.collect.serialize	<p>This parameter takes effect only when both <b>spark.sql.uselocalFileCollect</b> and <b>spark.sql.collect.Hive</b> are set to <b>true</b>.</p> <p>The function is to further improve performance.</p> <ul style="list-style-type: none"> <li>• <b>java</b>: Data is collected in Java serialization mode.</li> <li>• <b>kryo</b>: Data is collected in kryo serialization mode. The performance is better than that when the Java serialization mode is used.</li> </ul>	java

 NOTE

**spark.sql.bigdata.thriftServer.useHdfsCollect** and **spark.sql.uselocalFileCollect** cannot be set to **true** at the same time.

### 24.2.7.17 Filtering Partitions without Paths in Partitioned Tables

#### Scenario

When you perform the *select* query in Hive partitioned tables, the **FileNotFoundException** exception is displayed if a specified partition path does not exist in HDFS. To avoid the preceding exception, configure **spark.sql.hive.verifyPartitionPath** parameter to filter partitions without paths.

#### Procedure

Perform either of the following methods to filter partitions without paths:

- Configure the following parameter in the **spark-defaults.conf** file on Spark client.

**Table 24-60** Parameter description

Parameter	Description	Default Value
spark.sql.hive.verifyPartitionPath	Whether to filter partitions without paths when reading Hive partitioned tables. <b>true</b> : enables the filtering <b>false</b> : disables the filtering	false

- When running the spark-submit command to submit an application, configure the **--conf** parameter to filter partitions without paths.

For example:

```
spark-submit --class org.apache.spark.examples.SparkPi --conf spark.sql.hive.verifyPartitionPath=true $SPARK_HOME/lib/spark-examples_*.jar
```

### 24.2.7.18 Configuring Spark2x Web UI ACLs

#### Scenario

Users need to implement security protection for Spark2x web UI when some data on the UI cannot be viewed by other users. Once a user attempts to log in to the UI, Spark2x can check the view ACL of the user to determine whether to allow the user to access the UI.

Spark2x has two types of web UI. One is for running tasks. You can access the web UI using the application link on the native Yarn page or the REST APIs. The other one is for ended tasks. You can access the web UI using the Spark2x JobHistory service or the REST APIs. Spark2x and Spark configurations differ from each other.

- Configuring the ACL of the web UI for running tasks  
Use the following parameters for configuration:
  - **spark.admin.acls**: specifies the web UI administrator list.
  - **spark.admin.acls.groups**: specifies the administrator group list.
  - **spark.ui.view.acls**: specifies the Yarn page visitor list.
  - **spark.modify.acls.groups**: specifies the Yarn page visitor group list.
  - **spark.modify.acls**: specifies the web UI modifier list.
  - **spark.ui.view.acls.groups**: specifies the web UI modifier group list.

- Configuring the ACL of the web UI for ended tasks  
For ended tasks, use client parameter **spark.history.ui.acls.enable** to enable or disable the ACL access permission.

If ACL control is enabled, configure client parameters **spark.admin.acls** and **spark.admin.acls.groups** to specify the web UI administrator list and administrator group list. Use client parameters **spark.ui.view.acls** and **spark.modify.acls.groups** to specify the visitor list and visitor group list that view web UI task details. Use client parameters **spark.modify.acls** and **spark.ui.view.acls.groups** to specify the visitor list and group list that modify web UI task details.



## Configuration

Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Spark2x > Configurations**, click **All Configurations**, search for **acl**, and modify the following parameters on the JobHistory, JDBCServer, SparkResource, and Spark pages.

**Table 24-61** Parameter description

Parameter	Description	Default Value
spark.history.ui.acls.enable	Indicates whether JobHistory supports the permission verification of a single task.	true
spark.acls.enable	Indicates whether to enable Spark permission management. If this function is enabled, the system checks whether the user has the permission to access and modify task information.	true
spark.admin.acls	Indicates the list of Spark administrators. All members in the list have the rights to manage all Spark tasks. You can configure multiple administrators and separate them from each other using commas (,).	admin
spark.admin.acls.groups	Indicates the list of Spark administrator groups. All groups in the list have the permission to manage all Spark tasks. You can configure multiple administrator groups and separate them from each other using commas (,).	-
spark.modify.acls	Indicates the list of members that have the permission to modify Spark tasks. By default, the user who starts a task has the permission to modify the task. You can configure multiple users and separate them from each other using commas (,).	-
spark.modify.acls.groups	Indicates the list of groups that have the permission to modify Spark tasks. You can configure multiple groups and separate them from each other using commas (,).	-

Parameter	Description	Default Value
spark.ui.view.acls	Indicates the list of members that have the permission to access Spark tasks. By default, the user who starts a task has the permission to modify the task. You can configure multiple users and separate them from each other using commas (,).	-
spark.ui.view.acls.groups	Indicates the list of groups that have the permission to access Spark tasks. You can configure multiple groups and separate them from each other using commas (,).	-

 NOTE

If you use a client to submit tasks, you must download the client again after modifying the `spark.admin.acls`, `spark.admin.acls.groups`, `spark.modify.acls`, `spark.modify.acls.groups`, `spark.ui.view.acls`, and `spark.ui.view.acls.groups` parameters.

### 24.2.7.19 Configuring Vector-based ORC Data Reading

#### Scenario

ORC is a column-based storage format in the Hadoop ecosystem. It originates from Apache Hive and is used to reduce the Hadoop data storage space and accelerate the Hive query speed. Similar to Parquet, ORC is not a pure column-based storage format. In the ORC format, the entire table is split based on the row group, data in each row group is stored by column, and data is compressed as much as possible to reduce storage space consumption. Vector-based ORC data reading significantly improves the ORC data reading performance. In Spark2.3, SparkSQL supports vector-based ORC data reading (this function is supported in earlier Hive versions). Vector-based ORC data reading improves the data reading performance by multiple times.

This feature can be enabled by using the following parameter.

- **spark.sql.orc.enableVectorizedReader**: specifies whether vector-based ORC data reading is supported. The default value is **true**.
- **spark.sql.codegen.wholeStage**: specifies whether to compile all stages of multiple operations into a Java method. The default value is **true**.
- **spark.sql.codegen.maxFields**: specifies the maximum number of fields (including nested fields) supported by all stages of codegen. The default value is **100**.
- **spark.sql.orc.impl**: specifies whether Hive or Spark SQL native is used as the SQL execution engine to read ORC data. The default value is **hive**.

## Parameters

Log in to FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x**, click the **Configurations** tab and then **All Configurations**, and search for the following parameters.

Parameter	Description	Default Value	Value Range
spark.sql.orc.enableVectorizedReader	Specifies whether vector-based ORC data reading is supported. The default value is <b>true</b> .	true	[true,false]
spark.sql.codegen.wholeStage	Specifies whether to compile all stages of multiple operations into a Java method. The default value is <b>true</b> .	true	[true,false]
spark.sql.codegen.maxFields	Specifies the maximum number of fields (including nested fields) supported by all stages of codegen. The default value is <b>100</b> .	100	Greater than 0
spark.sql.orc.impl	Specifies whether Hive or Spark SQL native is used as the SQL execution engine to read ORC data. The default value is <b>hive</b> .	hive	[hive,native]

### NOTE

- To use vector-based ORC data reading of SparkSQL, the following conditions must be met:
  - spark.sql.orc.enableVectorizedReader** must be set to **true** (default value). Generally, the value is not changed.
  - spark.sql.codegen.wholeStage** must be set to **true** (default value). Generally, the value is not changed.
  - The value of **spark.sql.codegen.maxFields** must be greater than or equal to the number of columns in scheme.
  - All data is of the AtomicType. Specifically, data is not null or of the UDT, array, or map type. If there is data of the preceding types, expected performance cannot be obtained.
  - spark.sql.orc.impl** must be set to **native**. The default value is **hive**.
- If a task is submitted using the client, modification of the following parameters takes effect only after you download the client again: **spark.sql.orc.enableVectorizedReader**, **spark.sql.codegen.wholeStage**, **spark.sql.codegen.maxFields**, and **spark.sql.orc.impl**.

## 24.2.7.20 Broaden Support for Hive Partition Pruning Predicate Pushdown

### Scenario

In earlier versions, the predicate for pruning Hive table partitions is pushed down. Only comparison expressions between column names and integers or character strings can be pushed down. In version 2.3, pushdown of the null, in, and, or expressions are supported.

### Parameters

Log in to FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x**. On the displayed page, choose **Configurations** > **All Configurations**. Search for the parameter listed in the following table and configure it as required.

Parameter	Description	Default Value	Value Range
spark.sql.hive.advancedPartitionPredicatePushdown.enabled	Specifies whether to broaden the support for Hive partition pruning predicate pushdown.	true	[true,false]

## 24.2.7.21 Hive Dynamic Partition Overwriting Syntax

### Scenario

In earlier versions, when the **insert overwrite** syntax is used to overwrite partition tables, only partitions with specified expressions are matched, and partitions without specified expressions are deleted. In Spark2.3, partitions without specified expressions are automatically matched. The syntax is the same as that of the dynamic partition matching syntax of Hive.

### Parameters

Log in to FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x** > **Configurations**, click **All Configurations**, and search for the following parameter.

Parameter	Description	Default Value	Value Range
spark.sql.sources.partitionOverwrite-Mode	Specifies the mode for inserting data in partition tables by running the <b>insert overwrite</b> command, which can be <b>STATIC</b> or <b>DYNAMIC</b> . When it is set to <b>STATIC</b> , Spark deletes all partitions based on the matching conditions. When it is set to <b>DYNAMIC</b> , Spark matches partitions based on matching conditions and dynamically matches partitions without specified conditions.	STATIC	[STATIC,DYNAMIC]

## 24.2.7.22 Configuring the Column Statistics Histogram to Enhance the CBO Accuracy

### Scenarios

The execution plan for SQL statements is optimized in Spark. Common optimization rules are heuristic optimization rules. Heuristic optimization rules are provided based on the characteristics of logical plans, and the data characteristics and the execution costs of operators are not considered. Spark 2.20 introduces the Cost-Based Optimization (CBO). CBO collects statistics on tables and columns and estimates the number of output records and size of each operator in bytes based on the input data sets of operators, which is the cost of executing an operator.

CBO will adjust the execution plan to minimize the end-to-end query time. The main points are as follows:

- Filter out irrelevant data as soon as possible.
- Minimize the cost of each operator.

The CBO optimization process is divided into two steps:

1. Collect statistics.
2. Estimate the output data sets of a specific operator based on the input data sets.

Table-level statistics includes: number of records and the total size of a table data file.

Column-level statistics includes: number of unique values, maximum value, minimum value, number of null values, average length, maximum length, and the histogram.

After the statistics is obtained, the execution cost of operators can be estimated. Common operators include filter and join operators.

Histogram is a type of column statistics. It can clearly describe the distribution of column data. The column data is distributed to a specified number of bins that are displayed in ascending order by size. The upper and lower limits of each bin are calculated. The amount of data in all bins is the same (a contour histogram). After the data is distributed, the cost estimation of each operator is more accurate and the optimization effect is better.

This feature can be enabled by using the following parameter.

**spark.sql.statistics.histogram.enabled:** specifies whether to enable the histogram function. The default value is **false**.

## Parameter Configuration

Log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Spark2x > Configurations**, click **All Configurations**, and search for the following parameters.

Parameter	Description	Default Value	Value Range
spark.sql.cbo.enabled	Enables CBO to estimate the statistics for the execution plan.	false	[true,false]
spark.sql.cbo.joinReorder.enabled	Enables CBO join for reordering.	false	[true,false]
spark.sql.cbo.joinReorder.dp.threshold	Specifies the maximum number of nodes that can be joined in the dynamic planning algorithm.	12	>=1
spark.sql.cbo.joinReorder.card.weight	Specifies the proportion of dimension (number of rows) in the comparison of planned cost during reconnection: Number of rows x Proportion of dimension + File size x (1 - Proportion of dimension)	0.7	0-1
spark.sql.statistics.size.autoUpdate.enabled	Enables the function of automatically updating the table size when the table data volume changes. Note: If there are a large number of data files in a table, this operation is time consume, and the data processing speed is reduced.	false	[true,false]

Parameter	Description	Default Value	Value Range
spark.sql.statistics.histogram.enabled	After this function is enabled, a histogram is generated when column statistics is collected. Histogram can improve the estimation accuracy, but collecting histogram information requires additional workload.	false	[true,false]
spark.sql.statistics.histogram.numBins	Specifies the number of bins for the generated histogram.	254	>=2
spark.sql.statistics.ndv.maxError	Specifies the maximum estimation deviation allowed by the HyperLogLog++ algorithm when the column level statistics is generated.	0.05	0-1
spark.sql.statistics.percentile.accuracy	Specifies the accuracy rate of the percentile estimation when the contour histogram is generated. The larger the value is, the more accurate the estimation is. The estimation error value can be obtained through (1.0/Accuracy rate of the percentile estimation).	10000	>=1

 NOTE

- If you want the histogram to take effect in CBO, the following conditions must be met:
  - Set **spark.sql.statistics.histogram.enabled** to **true**. The default value is **false**. Change the value to **true** to enable the histogram function.
  - Set **spark.sql.cbo.enabled** to **true**. The default value is **false**. Change the value to **true** to enable CBO.
  - Set **spark.sql.cbo.joinReorder.enabled** to **true**. The default value is **false**. Change the value to **true** to enable connection reordering.
- If a client is used to submit a task, you need to download the client again after configuring the following parameters: **spark.sql.cbo.enabled**, **spark.sql.cbo.joinReorder.enabled**, **spark.sql.cbo.joinReorder.dp.threshold**, **spark.sql.cbo.joinReorder.card.weight**, **spark.sql.statistics.size.autoUpdate.enabled**, **spark.sql.statistics.histogram.enabled**, **spark.sql.statistics.histogram.numBins**, **spark.sql.statistics.ndv.maxError**, and **spark.sql.statistics.percentile.accuracy**.

### 24.2.7.23 Configuring Local Disk Cache for JobHistory

#### Scenarios

JobHistory can use local disks to cache the historical data of Spark applications to prevent the JobHistory memory from loading a large amount of application data, reducing the memory pressure. In addition, the cached data can be reused to improve the speed for subsequent application access.

#### Parameter Configuration

Log in to FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x** > **Configurations**, click the **All Configurations** tab, and search for the following parameters:

Parameter	Description	Default Value
spark.history.store.path	Specifies the local directory for storing historical information for JobHistory. If this parameter is specified, JobHistory caches historical application data in the local disk instead of the memory.	\${BIGDATA_HOME}/tmp/spark2x_JobHistory
spark.history.store.maxDiskUsage	Specifies the maximum available space of the local disk cache.	10 GB

### 24.2.7.24 Configuring Spark SQL to Enable the Adaptive Execution Feature

#### Scenario

The Spark SQL adaptive execution feature enables Spark SQL to optimize subsequent execution processes based on intermediate results to improve overall execution efficiency. The following features have been implemented:

1. Automatic configuration of the number of shuffle partitions  
Before the adaptive execution feature is enabled, Spark SQL specifies the number of partitions for a shuffle process by specifying the **spark.sql.shuffle.partitions** parameter. This method lacks flexibility when multiple SQL queries are performed on an application and cannot ensure optimal performance in all scenarios. After adaptive execution is enabled, Spark SQL automatically configures the number of partitions for each shuffle process, instead of using the general configuration. In this way, the proper number of partitions is automatically used during each shuffle process.
2. Dynamic adjusting of the join execution plan  
Before the adaptive execution feature is enabled, Spark SQL creates an execution plan based on the optimization results of rule-based optimization (RBO) and Cost-Based Optimization (CBO). This method ignores changes of result sets during data execution. For example, when a view created based on a large table is joined with other large tables, the execution plan cannot be



adjusted to BroadcastJoin even if the result set of the view is small. After the adaptive execution feature is enabled, Spark SQL can dynamically adjust the execution plan based on the execution result of the previous stage to obtain better performance.

3. Automatic processing of data skew

If data skew occurs during SQL statement execution, the memory overflow of an executor or slow task execution may occur. After the adaptive execution feature is enabled, Spark SQL can automatically process data skew scenarios. Multiple tasks are started for partitions where data skew occurs. Each task reads several output files obtained from the shuffle process and performs union operations on the join results of these tasks to eliminate data skew.

## Parameters

Log in to FusionInsight Manager, choose **Cluster > Services > Spark2x > Configurations**, click **All Configurations**, and search for the following parameter.

Parameter	Description	Default Value
spark.sql.adaptive.enabled	Specifies whether to enable the adaptive execution function.  Note: If AQE and Static Partition Pruning (DPP) are enabled at the same time, DPP takes precedence over AQE during SparkSQL task execution. As a result, AQE does not take effect.	false
spark.sql.optimizer.dynamicPartitionPruning.enabled	The switch to enable DPP.	true
spark.sql.adaptive.coalescePartitions.enabled	If this parameter is set to <b>true</b> and <b>spark.sql.adaptive.enabled</b> is set to <b>true</b> , Spark combines partitions that are consecutively random played based on the target size (specified by <b>spark.sql.adaptive.advisoryPartitionSizeInBytes</b> ) to prevent too many small tasks from being executed.	true
spark.sql.adaptive.coalescePartitions.initialPartitionNum	Initial number of shuffle partitions before merge. The default value is the same as the value of <b>spark.sql.shuffle.partitions</b> . This parameter is valid only when <b>spark.sql.adaptive.enabled</b> and <b>spark.sql.adaptive.coalescePartitions.enabled</b> are set to <b>true</b> . This parameter is optional. The initial number of partitions must be a positive number.	200

Parameter	Description	Default Value
spark.sql.adaptive.coalescePartitions.minPartitionNum	Minimum number of shuffle partitions after merge. If this parameter is not set, the default degree of parallelism (DOP) of the Spark cluster is used. This parameter is valid only when <b>spark.sql.adaptive.enabled</b> and <b>spark.sql.adaptive.coalescePartitions.enabled</b> are set to <b>true</b> . This parameter is optional. The initial number of partitions must be a positive number.	1
spark.sql.adaptive.shuffle.targetPostShuffleInputSize	Target size of a partition after shuffling. Spark 3.0 and later versions do not support this parameter.	64MB
spark.sql.adaptive.advisoryPartitionSizeInBytes	Size of a shuffle partition (unit: byte) during adaptive optimization ( <b>spark.sql.adaptive.enabled</b> is set to <b>true</b> ). This parameter takes effect when Spark aggregates small shuffle partitions or splits shuffle partitions where skew occurs.	64MB
spark.sql.adaptive.fetchShuffleBlocksInBatch	Whether to obtain consecutive shuffle blocks in batches. For the same map job, reading consecutive shuffle blocks in batches can reduce I/Os and improve performance, instead of reading blocks one by one. Note that multiple consecutive blocks exist in a single read request only when <b>spark.sql.adaptive.enabled</b> and <b>spark.sql.adaptive.coalescePartitions.enabled</b> are set to <b>true</b> . This feature also relies on a relocatable serializer that uses cascading to support the codec and the latest version of the shuffle extraction protocol.	true
spark.sql.adaptive.localShuffleReader.enabled	If the value of this parameter is <b>true</b> and the value of <b>spark.sql.adaptive.enabled</b> is <b>true</b> , Spark attempts to use the local shuffle reader to read shuffle data when shuffling of partitions is not required, for example, after sort-merge join is converted to broadcast-hash join.	true

Parameter	Description	Default Value
spark.sql.adaptive.skewJoin.enabled	Specifies whether to enable the function of automatic processing of the data skew in join operations. The function is enabled when this parameter is set to <b>true</b> and <b>spark.sql.adaptive.enabled</b> is set to <b>true</b> .	true
spark.sql.adaptive.skewJoin.skewedPartitionFactor	This parameter is a multiplier used to determine whether a partition is a data skew partition. If the data size of a partition exceeds the value of this parameter multiplied by the median of the all partition sizes except this partition and exceeds the value of <b>spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes</b> , this partition is considered as a data skew partition.	5
spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes	If the partition size (unit: byte) is greater than the threshold as well as the product of the <b>spark.sql.adaptive.skewJoin.skewedPartitionFactor</b> value and the median partition size, skew occurs in the partition. Ideally, the value of this parameter should be greater than that of <b>spark.sql.adaptive.advisoryPartitionSizeInBytes</b> .	256MB
spark.sql.adaptive.nonEmptyPartitionRatioForBroadcastJoin	If the ratio of non-null partitions is less than the value of this parameter when two tables are joined, broadcast hash join cannot be properly performed regardless of the partition size. This parameter is valid only when <b>spark.sql.adaptive.enabled</b> is set to <b>true</b> .	0.2

### 24.2.7.25 Configuring Event Log Rollover

#### Scenario

When the event log mode is enabled for Spark, that is, **spark.eventLog.enabled** is set to **true**, events are written to a configured log file to record the program running process. If a program, for example JDBCServer or Spark Streaming, runs

for a long period of time and has run many jobs and tasks during this period, many events are recorded in the log file, significantly increasing the file size.

When log rollover is enabled, metadata events are written into the log file and job events are written into a new log file (whether a job event is written to the new log file depends on the file size). Metadata events include EnvironmentUpdate, BlockManagerAdded, BlockManagerRemoved, UnpersistRDD, ExecutorAdded, ExecutorRemoved, MetricsUpdate, ApplicationStart, ApplicationEnd, and LogStart. Job events include StageSubmitted, StageCompleted, TaskResubmit, TaskStart, TaskEnd, TaskGettingResult, JobStart, and JobEnd. For Spark SQL applications, job events also include ExecutionStart and ExecutionEnd.

The UI for the HistoryServer service of Spark is obtained by reading and parsing these log files. The memory size is preset before the HistoryServer process starts. Therefore, when the size of log files is large, loading and parsing these files may cause problems such as insufficient memory and driver GC.

To load large log files in small memory mode, you need to enable log rollover for large applications. Generally, it is recommended that this function be enabled for long-running applications.

## Parameters

Log in to FusionInsight Manager, choose **Cluster > Services > Spark2x > Configurations**, click **All Configurations**, and search for the following parameters.

Parameter	Description	Default Value
spark.eventLog.rolling.enabled	Whether to enable rollover for event log files. If this parameter is set to <b>true</b> , the size of each event log file is reduced to the configured size.	true
spark.eventLog.rolling.maxFileSize	Maximum size of the event log file to be rolled over when <b>spark.eventlog.rolling.enabled</b> is set to <b>true</b> .	128M
spark.eventLog.compression.codec	Codec used to compress event logs. By default, Spark provides four types of codecs: LZ4, LZF, Snappy, and ZSTD. If this parameter is not specified, <b>spark.io.compression.codec</b> is used.	None
spark.eventLog.logStageExecutorMetrics	Whether to write each stage peak value (for each executor) of executor metrics to the event log.	false

## 24.2.8 Adapting to the Third-party JDK When Ranger Is Used

### Scenarios

When Ranger is used as the permission management service of Spark SQL, the certificate in the cluster is required for accessing RangerAdmin. If you use a third-party JDK instead of the JDK or JRE in the cluster, RangerAdmin fails to be accessed. As a result, the Spark application fails to be started.

In this scenario, you need to perform the following operations to import the certificate in the cluster to the third-party JDK or JRE.

### Configuration Method

**Step 1** Run the following command to export the certificate from the cluster:

1. Install the cluster client. Assume that the installation path is **/opt/client**.
2. Run the following command to go to the client installation directory.  
**cd /opt/client**
3. Run the following command to configure environment variables:  
**source bigdata\_env**
4. Generate the certificate file.  
**keytool -export -alias fusioninsightsubroot -storepass changeit -keystore /opt/client/JRE/jre/lib/security/cacerts -file fusioninsightsubroot.crt**

**Step 2** Import the certificate in the cluster to the third-party JDK or JRE.

Copy the **fusioninsightsubroot.crt** file generated in **Step 1** to the third-party JRE node, set the **JAVA\_HOME** environment variable of the node, and run the following command to import the certificate:

```
keytool -import -trustcacerts -alias fusioninsightsubroot -storepass changeit -file fusioninsightsubroot.crt -keystore MY_JRE/lib/security/cacerts
```

#### NOTE

**MY\_JRE** indicates the installation path of the third-party JRE. Change it based on the site requirements.

----End

## 24.3 Spark2x Logs

### Log Description

Log paths:

- Executor run log: **\${BIGDATA\_DATA\_HOME}/hadoop/data\${i}/nm/containerlogs/application\_\${appid}/container\_\${\$contid}**

 **NOTE**

The logs of running tasks are stored in the preceding path. After the running is complete, the system determines whether to aggregate the logs to an HDFS directory based on the Yarn configuration. For details, see [Common Yarn Parameters](#).

- Other logs: `/var/log/Bigdata/spark2x`

**Log archiving rule:**

- When tasks are submitted in **yarn-client** or **yarn-cluster** mode, executor log files are stored each time when the size of the log files reaches 50 MB. A maximum of 10 log files can be reserved without being compressed.
- The JobHistory2x log file is backed up each time when the size of the log file reaches 100 MB. A maximum of 100 log files can be reserved without being compressed.
- The JDBCServer2x log file is backed up each time when the size of the log file reaches 100 MB. A maximum of 100 log files can be reserved without being compressed.
- The IndexServer2x log file is backed up each time when the size of the log file reaches 100 MB. A maximum of 100 log files can be reserved without being compressed.
- The JDBCServer2x audit log file is backed up each time when the size of the log file reaches 20 MB by default. A maximum of 20 log files can be reserved without being compressed.
- The log file size and the number of compressed files to be reserved can be configured on FusionInsight Manager.

**Table 24-62** Spark2x log list

Log Type	Name	Description
SparkResource2x logs	spark.log	Spark2x service initialization log
	prestart.log	Prestart script log
	cleanup.log	Cleanup log file for instance installation and uninstallation
	spark-availability-check.log	Spark2x service health check log
	spark-service-check.log	Spark2x service check log
JDBCServer2x logs	JDBCServer-start.log	JDBCServer2x startup log
	JDBCServer-stop.log	JDBCServer2x stop log
	JDBCServer.log	JDBCServer2x run log on the server
	jdbc-state-check.log	JDBCServer2x health check log
	jdbcservice-omm-pid***-gc.log.*.current	JDBCServer2x process GC log

Log Type	Name	Description
	spark-omm- org.apache.spark.sql.hive.t hriftserver.HiveThriftProxy Server2-***.out*	JDBCServer2x process startup log. If the process stops, the <b>jstack</b> information is printed.
JobHistory2x logs	jobHistory-start.log	JobHistory2x startup log
	jobHistory-stop.log	JobHistory2x stop log
	JobHistory.log	JobHistory2x running process log
	jobhistory-omm-pid***- gc.log.*.current	JobHistory2x process GC log
	spark-omm- org.apache.spark.deploy.hi story.HistoryServer- ***.out*	JobHistory2x process startup log. If the process stops, the <b>jstack</b> information is printed.
IndexServer2x logs	IndexServer-start.log	IndexServer2x startup log
	IndexServer-stop.log	IndexServer2x stop log
	IndexServer.log	IndexServer2x run log on the server
	indexserver-state- check.log	IndexServer2x health check log
	indexserver-omm-pid***- gc.log.*.current	IndexServer2x process GC log
	spark-omm- org.apache.spark.sql.hive.t hriftserver.IndexServerPro xy-***.out*	IndexServer2x process startup log. If the process stops, the <b>jstack</b> information is printed.
Audit Log	jdbcserver-audit.log	JDBCServer2x audit log
	ranger-audit.log	

## Log levels

**Table 24-63** describes the log levels supported by Spark2x. The priorities of log levels are ERROR, WARN, INFO, and DEBUG in descending order. Logs whose levels are higher than or equal to the specified level are printed. The number of printed logs decreases as the specified log level increases.

**Table 24-63** Log levels

Level	Description
ERROR	Error information about the current event processing
WARN	Exception information about the current event processing
INFO	Logs of this level record normal running status information about the system and events.
DEBUG	Logs of this level record the system information and system debugging information.

To modify log levels, perform the following operations:

 **NOTE**

By default, the service does not need to be restarted after the Spark2x log levels are configured.

- Step 1** Log in to FusionInsight Manager.
- Step 2** Choose **Cluster** > *Name of the desired cluster* > **Service** > **Spark2x** > **Configuration**.
- Step 3** Select **All Configurations**.
- Step 4** On the menu bar on the left, select the log menu of the target role.
- Step 5** Select a desired log level.
- Step 6** Click **Save**. Then, click **OK**.

----End

## Log Format

**Table 24-64** Log Format

Type	Format	Example
Run log	<yyyy-MM-dd HH:mm:ss,SSS> <Log level>  <Name of the thread that generates the log>  <Message in the log>  <Location where the log event occurs>	2014-09-22 11:16:23,980 INFO DAGScheduler: Final stage: Stage 0(reduce at SparkPi.scala:35)



## 24.4 Obtaining Container Logs of a Running Spark Application

Container logs of running Spark applications are distributed on multiple nodes. This section describes how to quickly obtain container logs.

### Scenario Description

You can run the **yarn logs** command to obtain the logs of applications running on Yarn. In different scenarios, you can run the following commands to obtain required logs:

1. Obtain complete logs of the application: **yarn logs --applicationId <appld> -out <outputDir>**.

Example: **yarn logs --applicationId application\_1574856994802\_0016 -out /opt/test**

The following figure shows the command output.

- a. If the application is running, container logs in the **dead** state cannot be obtained.
  - b. If the application is stopped, all archived container logs can be obtained.
2. Obtain logs of a specified container: **yarn logs -applicationId <appld> -containerId <containerId>**.

Example: **yarn logs -applicationId application\_1574856994802\_0018 -containerId container\_e01\_1574856994802\_0018\_01\_000003**

The following figure shows the command output.

- a. If the application is running, container logs in the **dead** state cannot be obtained.
  - b. If the application is stopped, you can obtain logs of any container.
3. Obtain container logs in any state: **yarn logs -applicationId <appld> -containerId <containerId> -nodeAddress <nodeAddress>**

Example: **yarn logs -applicationId application\_1574856994802\_0019 -containerId container\_e01\_1574856994802\_0019\_01\_000003 -nodeAddress 192-168-1-1:8041**

Execution result: Logs of any container can be obtained.

#### NOTE

You need to set *nodeAddress* in the command. You can run the following command to obtain the value:

```
yarn node -list -all
```

## 24.5 Small File Combination Tools

### Tool Overview

In a large-scale Hadoop production cluster, HDFS metadata is stored in the NameNode memory, and the cluster scale is restricted by the memory limitation

of each NameNode. If there are a large number of small files in the HDFS, a large amount of NameNode memory is consumed, which greatly reduces the read and write performance and prolongs the job running time. Based on the preceding information, the small file problem is a key factor that restricts the expansion of the Hadoop cluster.

This tool provides the following functions:

1. Checks the number of small files whose size is less than the threshold configured by the user in tables and returns the average size of all data files in the table directory.
2. Provides the function of combination table files. Users can set the average file size after combination.

## Supported Table Types

Spark: Parquet, ORC, CSV, Text, and Json.

Hive: Parquet, ORC, CSV, Text, RCFile, Sequence and Bucket.

### NOTE

1. After tables with compressed data are merged, Spark uses the default compression format Snappy for data compression. You can configure **spark.sql.parquet.compression.codec** (available values: **uncompressed**, **gzip**, **lzo**, and **snappy**) and **spark.sql.orc.compression.codec** (available values: **uncompressed**, **zlib**, **lzo**, and **snappy**) on the client to select the compression format for the Parquet and ORC tables. Compression formats available for Hive and Spark tables are different, except the preceding compression formats, other compression formats are not supported.
2. To merge bucket table data, you need to add the following configurations to the **hive-site.xml** file on the Spark2x client:

```
<property>
<name>hive.enforce.bucketing</name>
<value>>false</value>
</property>
<property>
<name>hive.enforce.sorting</name>
<value>>false</value>
</property>
```
3. Spark does not support the feature of encrypting data columns in Hive.

## Tool Usage

Download and install the client. For example, the installation directory is **/opt/client**. Go to **/opt/client/Spark2x/spark/bin** and run the **mergetool.sh** script.

### Environment variables loading

```
source /opt/client/bigdata_env
```

```
source /opt/client/Spark2x/component_env
```

### Scanning function

Command: **sh mergetool.sh scan <db.table> <filesize>**

The format of *db.table* is *Database name, Table name*. *filesize* is the user-defined threshold of the small file size (unit: MB). The returned result is the number of

files that is smaller than the threshold and the average size of data files in the table directory.

Example: **sh mergetool.sh scan default.table1 128**

### Combination function

Command: **sh mergetool.sh merge <db.table> <filesize> <shuffle>**

The format of *db.table* is *Database name,Table name*. **filesize** is the user-defined average file size after file combination (unit: MB). **shuffle** is a Boolean value, and the value is **true** or **false**, which is used to configure whether to allow data to be shuffled during the merge.

Example: **sh mergetool.sh merge default.table1 128 false**

If the following information is displayed, the operation is successful:

SUCCESS: Merge succeeded

#### NOTE

1. Ensure that the current user is the owner of the merged table.
2. Before combination, ensure that HDFS has sufficient storage space, greater than the size of the combined table.
3. Table data must be combined separately. If a table is read during table data combination, the file may not be found temporarily. After the combination is complete, this problem is resolved. During the combination, do not write data to the corresponding tables. Otherwise, data inconsistency may occur.
4. If an error occurs indicating that the file does not exist when the query of data in a partitioned table is performed on the session spark-beeline/spark-sql that is always in the connected status. You can run the **refresh table** *Table name* command as prompted to query the data again.
5. Configure **filesize** based on the site requirements. For example, you can set **filesize** to a value greater than the average during file merging after obtaining the average file size by file scan. Otherwise, the number of files may increase after the file merging.
6. During the file merging, data in the original tables is removed to the recycle bin. In the case of any exception occurs on the data after file merging, the data in the original tables is used to replace the damaged data. If an exception occurs during the process, restore the data in the trash directory by using the **mv** command in HDFS.
7. In the HDFS router federation scenario, if the target NameService of the table root path is different from that of the root path **/user**, you need to manually clear the original table files stored in the recycle bin during the second combination. Otherwise, the combination fails.
8. This tool uses the configuration of the client. Performance optimization can be performed modifying required configuration in the client configuration file.

### shuffle configuration

For the combination function, you can roughly estimate the change on the number of partitions before and after the combination.

Generally, if the number of old partitions is greater than the number of new partitions, set **shuffle** to **false**. However, if the number of old partitions is much greater than that of new partitions (for example, more than 100 times), you can set **shuffle** to **true** to increase the degree of parallelism and improve the combination speed.

**NOTICE**

- If **shuffle** is set to **true** (repartition), the performance is improved. However, due to the particularity of the Parquet and ORC storage modes, repartition will reduce the compression ratio and the total size of the table in HDFS increases by 1.3 times.
- If **shuffle** is set to **false** (coalesce), the merged files may have some difference in size, which is close to the value of the configured **filesize**.

**Log storage location**

The default log storage path is `/tmp/SmallFilesLog.log4j`. To customize the log storage path, you can configure `log4j.appender.logfile.File` in `/opt/client/Spark2x/spark/tool/log4j.properties`.

## 24.6 Using CarbonData for First Query

### Tool Overview

The first query of CarbonData is slow, which may cause a delay for nodes that have high requirements on real-time performance.

The tool provides the following functions:

- Preheat the tables that have high requirements on query delay for the first time.

### Tool Usage

Download and install the client. For example, the installation directory is `/opt/client`. Go to the `/opt/client/Spark2x/spark/bin` directory and run `start-prequery.sh`.

Configure `prequeryParams.properties` by referring to [Table 24-65](#).

**Table 24-65** Parameters

Parameter	Description	Example
<code>spark.prequery.period.max.minute</code>	Maximum preheating duration, in minutes.	60
<code>spark.prequery.tables</code>	Table name configuration, <i>database.table:int</i> . The table name supports the wildcard (*). <b>int</b> indicates the duration (unit: day) within which the table is updated before it is preheated.	default.test*:10

Parameter	Description	Example
spark.prequery.maxThreads	Maximum number of concurrent threads during preheating	50
spark.prequery.sslEnable	The value is <b>true</b> in security mode and <b>false</b> in non-security mode.	true
spark.prequery.driver	IP address and port number of JDBCServer. The format is <i>IP address:Port number</i> . If multiple servers need to be preheated, enter multiple <i>IP address:Port number</i> of the servers and separate them with commas (,).	192.168.0.2:22550
spark.prequery.sql	SQL statement for preheating. Different statements are separated by colons (:).	SELECT COUNT(*) FROM %s;SELECT * FROM %s LIMIT 1
spark.security.url	URL required by JDBC in security mode	;sasLQop=auth-conf;auth=KERBEROS;principal=spark2x/hadoop.hadoop.com@HADOOP.COM;

 NOTE

The statement configured in **spark.prequery.sql** is executed in each preheated table. The table name is replaced with **%s**.

**Script Usage**

Command format: **sh start-prequery.sh**

To run this command, place **user.keytab** or **jaas.conf** (either of them) and **krb5.conf** (mandatory) in the **conf** directory.

 NOTE

- Currently, this tool supports only Carbon tables.
- This tool initializes the Carbon environment and pre-reads table metadata to JDBCServer. Therefore, this tool is more suitable for multi-active instances and static allocation mode.

## 24.7 Spark2x Performance Tuning

## 24.7.1 Spark Core Tuning

### 24.7.1.1 Data Serialization

#### Scenario

Spark supports the following types of serialization:

- JsonSerializer
- KryoSerializer

Data serialization affects the Spark application performance. In specific data format, KryoSerializer offers 10X higher performance than JsonSerializer. For Int data, performance optimization can be ignored.

KryoSerializer depends on Chill of Twitter. Not all Java Serializable objects support KryoSerializer. Therefore, class must be manually registered.

Serialization involves task serialization and data serialization. Only JsonSerializer can be used for Spark task serialization. JsonSerializer and KryoSerializer can be used for data serialization.

#### Procedure

When the Spark program is running, a large volume of data needs to be serialized during the shuffle and RDD cache procedures. By default, JsonSerializer is used. You can also configure KryoSerializer as the data serializer to improve serialization performance.

Add the following code to enable KryoSerializer to be used:

- Implement the class registrar and manually register the class.

```
package com.etl.common;

import com.esotericsoftware.kryo.Kryo;
import org.apache.spark.serializer.KryoRegistrator;

public class DemoRegistrator implements KryoRegistrator
{
 @Override
 public void registerClasses(Kryo kryo)
 {
 //Class examples are given below. Register the custom classes.
 kryo.register(AggrateKey.class);
 kryo.register(AggrateValue.class);
 }
}
```

You can configure **spark.kryo.registrationRequired** on Spark client. Whether to require registration with Kryo. If set to 'true', Kryo will throw an exception if an unregistered class is serialized. If set to false (the default), Kryo will write unregistered class names along with each object. Writing class names can cause significant performance overhead. This operation will affect the system performance. If the value of **spark.kryo.registrationRequired** is configured to **true**, you need to manually register the class. For a class that is not serialized, the system will not automatically write the class name, but display an exception. Compare the configuration of **true** with that of **false**, the configuration of **true** has the better performance.

- Configure KryoSerializer as the data serializer and class registrar.

```
val conf = new SparkConf()
conf.set("spark.serializer", "org.apache.spark.serializer.KryoSerializer")
.set("spark.kryo.registrator", "com.etl.common.DemoRegistrator")
```

## 24.7.1.2 Optimizing Memory Configuration

### Scenario

Spark is a memory-based computing frame. If the memory is insufficient during computing, the Spark execution efficiency will be adversely affected. You can determine whether memory becomes the performance bottleneck by monitoring garbage collection (GC) and evaluating the resilient distributed dataset (RDD) size in the memory, and take performance optimization measures.

To monitor GC of node processes, add the `-verbose:gc -XX:+PrintGCDetails -XX:+PrintGCTimeStamps` parameter to the **spark.driver.extraJavaOptions** and **spark.executor.extraJavaOptions** in the client configuration file **conf/spark-default.conf**. If "Full GC" is frequently reported, GC needs to be optimized. Cache the RDD and query the RDD size in the log. If a large value is found, change the RDD storage level.

### Procedure

- To optimize GC, adjust the ratio of the young generation and tenured generation. Add **-XX:NewRatio** parameter to the **spark.driver.extraJavaOptions** and **spark.executor.extraJavaOptions** in the client configuration file **conf/spark-default.conf**. For example, export `SPARK_JAVA_OPTS="-XX:NewRatio=2"`. The new generation accounts for 1/3 of the heap, and the tenured generation accounts for 2/3.
- Optimize the RDD data structure when compiling Spark programs.
  - Use primitive arrays to replace fastutil arrays, for example, use fastutil library.
  - Avoid nested structure.
  - Avoid using String in keys.
- Suggest serializing the RDDs when developing Spark programs.

By default, data is not serialized when RDDs are cached. You can set the storage level to serialize the RDDs and minimize memory usage. For example:

```
testRDD.persist(StorageLevel.MEMORY_ONLY_SER)
```

## 24.7.1.3 Setting the DOP

### Scenario

The degree of parallelism (DOP) specifies the number of tasks to be executed concurrently. It determines the number of data blocks after the shuffle operation. Configure the DOP to improve the processing capability of the system.

Query the CPU and memory usage. If the tasks and data are not evenly distributed among nodes, increase the DOP. Generally, set the DOP to two or three times that of the total CPUs in the cluster.

## Procedure

Configure the DOP parameter using one of the following methods based on the actual memory, CPU, data, and application logic conditions:

- Configure the DOP parameter in the operation function that generates the shuffle. This method has the highest priority.  
`testRDD.groupByKey(24)`
- Configure the DOP using **spark.default.parallelism**. This method has the lower priority than the preceding one.  
`val conf = new SparkConf();  
conf.set("spark.default.parallelism", 24);`
- Configure the value of **spark.default.parallelism** in the **\$SPARK\_HOME/conf/spark-defaults.conf** file. This method has the lowest priority.  
`spark.default.parallelism 24`

### 24.7.1.4 Using Broadcast Variables

#### Scenario

Broadcast distributes data sets to each node. It allows data to be obtained locally when a data set is needed during a Spark task. If broadcast is not used, data serialization will be scheduled to tasks each time when a task requires data sets. It is time-consuming and makes the task get bigger.

1. If a data set will be used by each slice of a task, broadcast the data set to each node.
2. When small and big tables need to be joined, broadcast small tables to each node. This eliminates the shuffle operation, changing the join operation into a common operation.

#### Procedure

Add the following code to broadcast the testArr data to each node:

```
def main(args: Array[String]) {
 ...
 val testArr: Array[Long] = new Array[Long](200)
 val testBroadcast: Broadcast[Array[Long]] = sc.broadcast(testArr)
 val resultRdd: RDD[Long] = inpputRdd.map(input => handleData(testBroadcast, input))
 ...
}

def handleData(broadcast: Broadcast[Array[Long]], input: String) {
 val value = broadcast.value
 ...
}
```

### 24.7.1.5 Using the external shuffle service to improve performance

#### Scenario

When the Spark system runs applications that contain a shuffle process, an executor process also writes shuffle data and provides shuffle data for other executors in addition to running tasks. If the executor is heavily loaded and GC is triggered, the executor cannot provide shuffle data for other executors, affecting task running.



The external shuffle service is an auxiliary service in NodeManager. It captures shuffle data to reduce the load on executors. If GC occurs on an executor, tasks on other executors are not affected.

## Procedure

- Step 1** Log in to FusionInsight Manager.
- Step 2** Choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x** > **Configurations**. Select **All Configurations**.
- Step 3** Choose **SparkResource2x** > **Default** and modify the following parameters.

**Table 24-66** Parameter list

Parameter	Default Value	Changed To
spark.shuffle.service.enabled	false	true

- Step 4** Restart the Spark2x service for the configuration to take effect.

### NOTE

To use the External Shuffle Service function on the Spark2x client, you need to download and install the Spark2x client again.

----End

## 24.7.1.6 Configuring Dynamic Resource Scheduling in Yarn Mode

### Scenario

Resources are a key factor that affects Spark execution efficiency. When a long-running service (such as the JDBCServer) is allocated with multiple executors without tasks but resources of other applications are insufficient, resources are wasted and scheduled improperly.

Dynamic resource scheduling can add or remove executors of applications in real time based on the task load. In this way, resources are dynamically scheduled to applications.

## Procedure

- Step 1** Configure the external shuffle service.
- Step 2** Log in to FusionInsight Manager, and choose **Cluster** > *Name of the desired cluster* > **Service** > **Spark2x** > **Configuration** > **All Configurations**. Enter **spark.dynamicAllocation.enabled** in the search box and set its value to **true** to enable the dynamic resource scheduling function. This function is disabled by default.

----End

[Table 24-67](#) lists some optional configuration items.

**Table 24-67** Parameters for dynamic resource scheduling

Configuration Item	Description	Default Value
spark.dynamicAllocation.minExecutors	Indicates the minimum number of executors.	0
spark.dynamicAllocation.initialExecutors	Indicates the number of initial executors.	0
spark.dynamicAllocation.maxExecutors	Indicates the maximum number of executors.	2048
spark.dynamicAllocation.schedulerBacklogTimeout	Indicates the first timeout period for scheduling.	1s
spark.dynamicAllocation.sustainedSchedulerBacklogTimeout	Indicates the second and later timeout interval for scheduling.	1s
spark.dynamicAllocation.executorIdleTimeout	Indicates the idle timeout interval for common executors.	60s
spark.dynamicAllocation.cachedExecutorIdleTimeout	Indicates the idle timeout interval for executors with cached blocks.	<ul style="list-style-type: none"> <li>• JDBCServer2x: 2147483647s</li> <li>• IndexServer2x: 2147483647s</li> <li>• SparkResource2x: 120</li> </ul>

 **NOTE**

The external shuffle service must be configured before using the dynamic resource scheduling function.

### 24.7.1.7 Configuring Process Parameters

#### Scenario

There are three processes in Spark on Yarn mode: driver, ApplicationMaster, and executor. The Driver and Executor handle the scheduling and running of the task. The ApplicationMaster handles the start and stop of the container.

Therefore, the configuration of the driver and executor is very important to run the Spark application. You can optimize the performance of the Spark cluster according to the following procedure.

#### Procedure

**Step 1** Configure the driver memory.

The driver schedules tasks and communicates with the executor and the ApplicationMaster. Add driver memory when the number and parallelism level of the tasks increases.

You can configure the driver memory based on the number of the tasks.

- Set **spark.driver.memory** in **spark-defaults.conf** to a proper value.
- Add the **--driver-memory MEM** parameter to configure the memory when using the **spark-submit** command.

### Step 2 Configure the number of the executors.

One core in an executor can run one task at the same time. Therefore, more tasks can be processed at the same time if you increase the number of the executors. You can add the number of the executors to increase the efficiency if resources are sufficient.

- Set **spark.executor.instance** in **spark-defaults.conf** or **SPARK\_EXECUTOR\_INSTANCES** in **spark-env.sh** to a proper value.
- Add the **--num-executors NUM** parameter to configure the number of the executors when using the **spark-submit** command.

### Step 3 Configure the number of the executor cores.

Multiple cores in an executor can run multiple tasks at the same time, which increases the task concurrency. However, because all cores share the memory of an executor, you need to balance the memory and the number of cores.

- Set **spark.executor.cores** in **spark-defaults.conf** or **SPARK\_EXECUTOR\_CORES** in **spark-env.sh** to a proper value.
- When you run the **spark-submit** command, add the **--executor-cores NUM** parameter to set the number of executor cores.

### Step 4 Configure the executor memory.

The executor memory is used for task execution and communication. You can increase the memory for a big task that needs more resources, and reduce the memory to increase the concurrency level for a small task that runs fast.

- Set **spark.executor.memory** in **spark-defaults.conf** or **SPARK\_EXECUTOR\_MEMORY** in **spark-env.sh** to a proper value.
- When you run the **spark-submit** command, add the **--executor-memory MEM** parameter to set the memory.

----End

## Example

- During the **spark wordcount** calculation, the amount of data is 1.6 TB and the number of the executors is 250.

The execution fails under the default configuration, and the **Futures timed out** and **OOM** errors occur.

However each task of wordcount is small and runs fast, the amount of the data is big and the tasks are too many. Therefore the objects on the driver end become huge when there are many tasks. Besides the fact that the executor communicates with the driver once each task is finished, the

problem of disconnection between processes caused by insufficient memory occurs.

The application runs successfully when the memory of the Driver is set to 4 GB.

- Many errors still occurred in the default configuration when running TPC-DS test on JDBCServer, such as "Executor Lost". When there is 30 GB of driver memory, 2 executor cores, 125 executors, and 6 GB of executor memory, all tasks can be successfully executed.

### 24.7.1.8 Designing the Direction Acyclic Graph (DAG)

#### Scenario

Optimal program structure helps increase execution efficiency. During application programming, avoid shuffle operations and combine narrow-dependency operations.

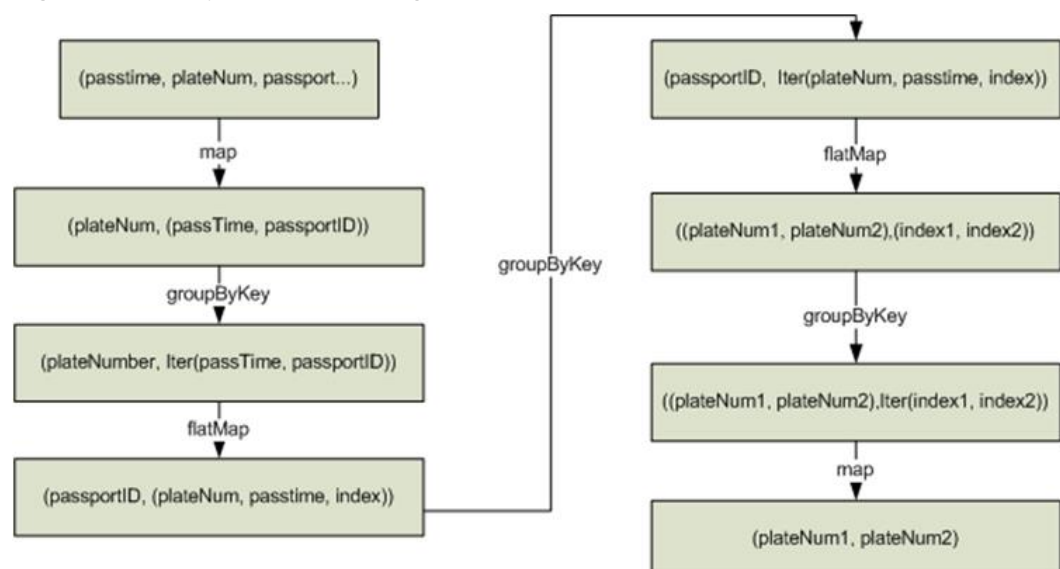
#### Procedure

This topic describes how to design the DAG using the following example:

- **Data format:** Time passing through the toll station, vehicle license plate number, toll station number ...
- **Logic:** Two vehicles are determined to be raveling together if the following conditions are met:
  - The two vehicles pass through tool stations in the same sequence.
  - The time difference that the two vehicles pass through the same toll station is below the specified value.

This example can be implemented in two ways:

**Figure 24-6** Implementation logic 1



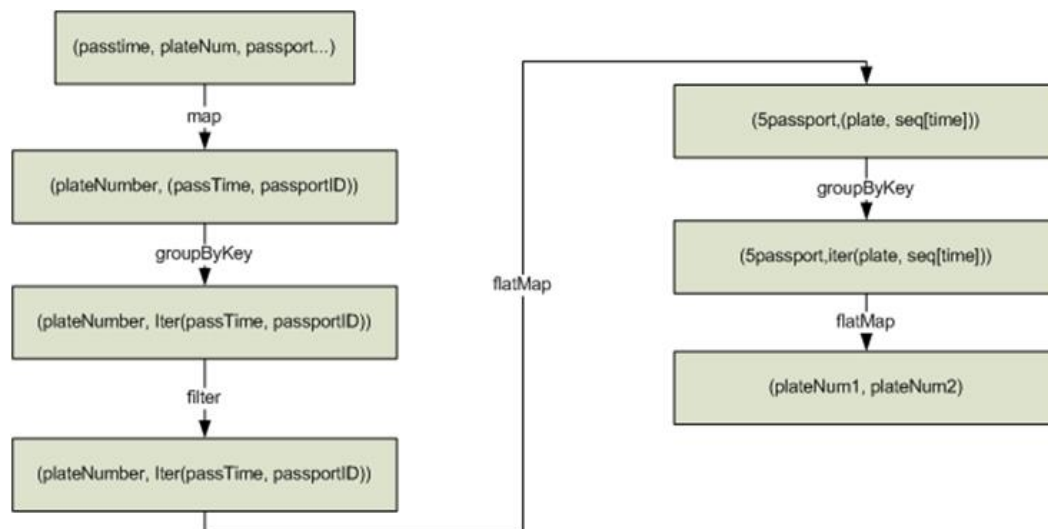
Logic of implementation 1:

1. Collect information about the toll stations passed by each vehicle based on the vehicle license plate number and sort the toll stations. The following data is obtained:  
vehicle license plate number 1, [(time, toll station 3), (time, toll station 2), (time, toll station 4), (time, toll station 5)]
2. Determine the sequence in which the vehicle passed through.  
(toll station 3, (vehicle license plate number 1, time, 1st toll station))  
(toll station 2, (vehicle license plate number 1, time, 2nd toll station))  
(toll station 4, (vehicle license plate number 1, time, 3rd toll station))  
(toll station 5, (vehicle license plate number 1, time, 4th toll station))
3. Aggregate data by toll station.  
toll station 1, [(vehicle license plate number 1, time, 1st toll station), (vehicle license plate number 2, time, 5th toll station), (vehicle license plate number 3, time, 2nd toll station)]
4. Determine whether the time difference that two vehicles passed through the same toll station is below the specified value. If yes, fetch information about the two vehicles.  
(vehicle license plate number 1, vehicle license plate number 2),(1st toll station, 5th toll station)  
(vehicle license plate number 1, vehicle license plate number 3),(1st toll station, 2nd toll station)
5. Aggregate data based on the vehicle license plate numbers that passed through the same toll stations.  
(vehicle license plate number 1, vehicle license plate number 2), [(1st toll station, 5th toll station), (2nd toll station, 6th toll station), (1st toll station, 7th toll station), (3rd toll station, 8th toll station)]
6. If the two vehicles pass through the same toll stations in sequence, for example, toll stations 3, 4, 5 are the first, second, and third toll station passed by vehicle 1 and the 6th, 7th, and 8th toll station passed by vehicle 2, and the number of toll stations meets the specified requirements, the two vehicles are determined to be traveling together.

The logic of implementation 1 has the following disadvantages:

- The logic is complex.
- Too many shuffle operations affect performance.

Figure 24-7 Implementation logic 2



Logic of implementation 2:

1. Collect information about the toll stations passed by each vehicle based on the vehicle license plate number and sort the toll stations. The following data is obtained:  
vehicle license plate number 1, [(time, toll station 3), (time, toll station 2), (time, toll station 4), (time, toll station 5)]
2. Based on the number of toll stations that must be passed by peer vehicles (it is 3 in this example), divide the toll station sequence as follows:  
toll station 3->toll station 2->toll station 4, (vehicle license plate number 1, [time passing through toll station 3, time passing through toll station 2, time passing through toll station 4])  
toll station 2->toll station 4->toll station 5, (vehicle license plate number 1, [time passing through toll station 2, time passing through toll station 4, time passing through toll station 5])
3. Aggregate the vehicles that passed through tool stations in the same sequence.  
toll station 3->toll station 2->toll station 4, [(vehicle license plate number 1, [time passing through toll station 3, time passing through toll station 2, time passing through toll station 4]), (vehicle license plate number 2, [time passing through toll station, time passing through toll station 3,time passing through toll station, time passing through toll station 2, time passing through toll station 4]), (vehicle license plate number 3, [time passing through toll station 3, time passing through toll station 2, time passing through toll station 4])]
4. Determine whether the time difference that these vehicles passed through the same toll station is below the specified value. If yes, the vehicles are determined to be raveling together.

The logic of implementation 2 has the following advantages:

- The logic is simplified.
- One groupByKey is reduced, that is, one less shuffle operation is performed. It helps improve performance.

## 24.7.1.9 Experience

### Use mapPartitions to calculate data by partition.

If the overhead of each record is high, for example:

```
rdd.map{x=>conn=getDBConn;conn.write(x.toString);conn.close}
```

Use mapPartitions to calculate data by partition.

```
rdd.mapPartitions(records => conn.getDBConn;for(item <- records)
write(item.toString); conn.close)
```

Use mapPartitions to flexibly operate data. For example, to calculate the TopN of a large data, mapPartitions can be used to calculate the TopN of each partition and then sort the TopN of all partitions when N is small. Compared with sorting full data for the TopN, this method has the higher efficiency.

### Use coalesce to adjust the number of slices.

Use coalesce to adjust the number of slices. There are two coalesce functions:

```
coalesce(numPartitions: Int, shuffle: Boolean = false)
```

When **shuffle** is set to **true**, the function is the same as `repartition(numPartitions: Int)`. Partitions are recreated using the shuffle. When **shuffle** is set to **false**, partitions of the parent resilient distributed datasets (RDD) are calculated in the same task. In this case, if the value of **numPartitions** is larger than the number of sections of the parent RDD, partitions will not be recreated.

The following scenario is encountered, you can choose the coalesce operator:

- If the previous operation involves a large number of filters, use coalesce to minimize the number of zero-loaded tasks. In `coalesce(numPartitions, false)`, the value of **numPartitions** is smaller than the number of sections of the parent RDD.
- Use coalesce when the number of slices entered is too big to execute.
- Use coalesce when the programs are suspended in the shuffle operation because of a large number of tasks or the Linux resources are limited. In this case, use `coalesce(numPartitions, true)` to recreate partitions.

### Configure a localDir for each disk.

During the shuffle procedure of Spark, data needs to be written into local disks. The performance bottleneck of Spark is shuffle, and the bottleneck of shuffle is the I/O. To improve the I/O performance, you can configure multiple disks to implement concurrent data writing. If a node is mounted with multiple disks, configure a Spark local Dir for each disk. This can effectively distribute shuffle files in multiple locations, improving disk I/O efficiency. The performance cannot be improved effectively if a disk is configured with multiple directories.

### Collect small data sets.

The collect operation does not apply to a large data volume.

When the collect operation is performed, the Executor data will be sent to the Driver. Before performing this operation, ensure that the memory of Driver is sufficient. Otherwise, the Driver process may encounter an OutOfMemory error. If the data volume is unknown, perform the saveAsTextFile operation to write data into the HDFS. If the data volume is known and the Driver has sufficient memory, perform the collect operation.

## Use reduceByKey

reduceByKey causes local aggregation on the Map side, which offers a smooth shuffle procedure. The shuffle operations, like groupByKey, will not perform aggregation on the Map side. Therefore, use reduceByKey as possible as you can, and avoid groupByKey().map(x=>(x.\_1,x.\_2.size)).

## Broadcast map instead of array.

If table query is required for each record of the data transmitted from the Driver side, broadcast the data in the set/map instead of Iterator. The query speed of Set/Map is approximately  $O(1)$ , while the query speed of Iterator is  $O(n)$ .

## Avoid data skew.

If data skew occurs (certain data volume is extremely large), the execution time of tasks is inconsistent even if there is no Garbage Collection (GC).

- Redefine the keys. Use keys of smaller granularity to optimize the task size.
- Modify the degree of parallelism (DOP).

## Optimize the data structure.

- Store data by column. As a result, only the required columns are scanned when data is read.
- When using the Hash Shuffle, set **spark.shuffle consolidateFiles** to **true** to combine the intermediate files of shuffle, minimize the number of shuffle files and file I/O operations, and improve performance. The number of final files is the number of reduce tasks.

## 24.7.2 Spark SQL and DataFrame Tuning

### 24.7.2.1 Optimizing the Spark SQL Join Operation

#### Scenario

When two tables are joined in Spark SQL, the broadcast function (see section "Using Broadcast Variables") can be used to broadcast tables to each node. This minimizes shuffle operations and improves task execution efficiency.

#### NOTE

The join operation refers to the inner join operation only.



## Procedure

The following describes how to optimize the join operation in Spark SQL. Assume that both tables A and B have the **name** column. Join tables A and B as follows:

1. Estimate the table sizes.

Estimate the table size based on the size of data loaded each time.

You can also check the table size in the directory of the Hive database. In the **hive-site.xml** configuration file of Spark, view the Hive database directory, which is **/user/hive/warehouse** by default. The default Hive database directory for multi-instance Spark is **/user/hive/warehouse**, for example, **/user/hive1/warehouse**.

```
<property>
 <name>hive.metastore.warehouse.dir</name>
 <value>${test.warehouse.dir}</value>
 <description></description>
</property>
```

Run the **hadoop** command to check the size of the table. For example, run the following command to view the size of table **A**:

```
hadoop fs -du -s -h ${test.warehouse.dir}/a
```

### NOTE

To perform the broadcast operation, ensure that at least one table is not empty.

2. Configure a threshold for automatic broadcast.

The threshold for triggering broadcast for a table is 10485760 (that is, 10 MB) in Spark. If either of the table sizes is smaller than 10 MB, skip this step.

**Table 24-68** lists configuration parameters of the threshold for automatic broadcasting.

**Table 24-68** Parameter description

Parameter	Default Value	Description
spark.sql.autoBroadcastJoinThreshold	10485760	<p>Indicates the maximum value for the broadcast configuration when two tables are joined.</p> <ul style="list-style-type: none"> <li>• When the size of a field in a table involved in an SQL statement is less than the value of this parameter, the system broadcasts the SQL statement.</li> <li>• If the value is set to <b>-1</b>, broadcast is not performed.</li> </ul> <p>For details, visit <a href="https://spark.apache.org/docs/3.1.1/sql-programming-guide.html">https://spark.apache.org/docs/3.1.1/sql-programming-guide.html</a>.</p>

Methods for configuring the threshold for automatic broadcasting:

- Set **spark.sql.autoBroadcastJoinThreshold** in the **spark-defaults.conf** configuration file of Spark.  
`spark.sql.autoBroadcastJoinThreshold = <size>`
  - Run the Hive command to set the threshold. Before joining the tables, run the following command:  
`SET spark.sql.autoBroadcastJoinThreshold=<size>;`
3. Join the tables.
- The size of each table is smaller than the threshold.
    - If the size of table A is smaller than that of table B, run the following command:  
`SELECT A.name FROM B JOIN A ON A.name = B.name;`
    - If the size of table B is smaller than that of table A, run the following command:  
`SELECT A.name FROM A JOIN B ON A.name = B.name;`
  - One table size is smaller than the threshold, while the other table size is greater than the threshold.  
Broadcast the smaller table.
  - The size of each table is greater than the threshold.  
Compare the size of the field involved in the query with the threshold.
    - If the values of the fields in a table are smaller than the threshold, the corresponding data in the table is broadcast.
    - If the values of the fields in the two tables are greater than the threshold, do not broadcast either of the table.
4. (Optional) In the following scenarios, you need to run the Analyze command (***ANALYZE TABLE tableName COMPUTE STATISTICS noscan;***) to update metadata before performing the broadcast operation:
- The table to be broadcasted is a newly created partitioned table and the file type is non-Parquet.
  - The table to be broadcasted is a newly updated partitioned table.

## Reference

A task is ended if a timeout occurs during the execution of the to-be-broadcasted table.

By default, BroadcastJoin allows only 5 minutes for the to-be-broadcasted table calculation. If the time is exceeded, a timeout will occur. However, the broadcast task of the to-be-broadcasted table calculation is still being executed, resulting in resource waste.

The following methods can be used to address this issue:

- Modify the value of **spark.sql.broadcastTimeout** to increase the timeout duration.
- Reduce the value of **spark.sql.autoBroadcastJoinThreshold** to disable the optimization of BroadcastJoin.

## 24.7.2.2 Improving Spark SQL Calculation Performance Under Data Skew

### Scenario

When multiple tables are joined in Spark SQL, skew occurs in join keys and the data volume in some Hash buckets is much higher than that in other buckets. As a result, some tasks with a large amount of data run slowly, resulting low computing performance. Other tasks with a small amount of data are quickly completed, which frees many CPUs and results in a waste of CPU resources.

If the automatic data skew function is enabled, data that exceeds the bucketing threshold is bucketed. Multiple tasks proceed data in one bucket. Therefore, CPU usage is enhanced and the system performance is improved.

#### NOTE

Data that has no skew is bucketed and run in the original way.

Restrictions:

- Only the join between two tables is supported.
- FULL OUTER JOIN data does not support data skew.  
For example, the following SQL statement indicates that the skew of table **a** or table **b** cannot trigger the optimization.  
***select aid FROM a FULL OUTER JOIN b ON aid=bid;***
- LEFT OUTER JOIN data does not support the data skew of the right table.  
For example, the following SQL statement indicates that the skew of table **b** cannot trigger the optimization.  
***select aid FROM a LEFT OUTER JOIN b ON aid=bid;***
- RIGHT OUTER JOIN does not support the data skew of the left table.  
For example, the following SQL statement indicates that the skew of table **a** cannot trigger the optimization.  
***select aid FROM a RIGHT OUTER JOIN b ON aid=bid;***

### Configuration Description

Add the following parameters in the following table to the **spark-defaults.conf** configuration file on the Spark driver.

**Table 24-69** Parameter description

Parameter	Description	Default Value
spark.sql.adaptive.enabled	The switch to enable the adaptive execution feature.  Note: If AQE and Static Partition Pruning (DPP) are enabled at the same time, DPP takes precedence over AQE during SparkSQL task execution. As a result, AQE does not take effect. The DPP in the cluster is enabled by default. Therefore, you need to disable it when enabling the AQE.	false
spark.sql.optimizer.dynamicPartitionPruning.enabled	The switch to enable DPP.	true
spark.sql.adaptive.skewJoin.enabled	Specifies whether to enable the function of automatic processing of the data skew in join operations. The function is enabled when this parameter is set to <b>true</b> and <b>spark.sql.adaptive.enabled</b> is set to <b>true</b> .	true
spark.sql.adaptive.skewJoin.skewedPartitionFactor	This parameter is a multiplier used to determine whether a partition is a data skew partition. If the data size of a partition exceeds the value of this parameter multiplied by the median of the all partition sizes except this partition and exceeds the value of <b>spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes</b> , this partition is considered as a data skew partition.	5
spark.sql.adaptive.skewjoin.skewedPartitionThresholdInBytes	If the partition size (unit: byte) is greater than the threshold as well as the product of the <b>spark.sql.adaptive.skewJoin.skewedPartitionFactor</b> value and the median partition size, skew occurs in the partition. Ideally, the value of this parameter should be greater than that of <b>spark.sql.adaptive.advisoryPartitionSizeInBytes</b> .	256MB
spark.sql.adaptive.shuffle.targetPostShuffleInputSize	Minimum amount of shuffle data processed by each task. The unit is byte.	67108864

### 24.7.2.3 Optimizing Spark SQL Performance in the Small File Scenario

#### Scenario

A Spark SQL table may have many small files (far smaller than an HDFS block), each of which maps to a partition on the Spark by default. In other words, each

small file is a task. If the small files are great in number, Spark must initiate a large number of tasks. If shuffle operations exist in Spark SQL, the number of hash buckets increases, affecting performance.

In this scenario, you can manually specify the split size of each task to avoid an excessive number of tasks and improve performance.

 **NOTE**

If the SQL logic does not involve shuffle operations, this optimization does not improve performance.

## Configuration

If you want to enable small file optimization, configure the **spark-defaults.conf** file on the Spark client.

**Table 24-70** Parameter description

Parameter	Description	Default Value
spark.sql.files.maxPartitionBytes	The maximum number of bytes that can be packed into a single partition when a file is read. Unit: byte	134217728 (128 MB)
spark.files.openCostInBytes	The estimated cost to open a file, measured by the number of bytes that can be scanned in the same time. This is used when putting multiple files into a partition. It is better to over estimate, then the partitions with small files will be faster than partitions with larger files.	4 MB

### 24.7.2.4 Optimizing the INSERT...SELECT Operation

#### Scenario

The INSERT...SELECT operation needs to be optimized if any of the following conditions is true:

- Many small files need to be queried.
- A few large files need to be queried.
- The INSERT...SELECT operation is performed by a non-spark user in Beeline/JDBCServer mode.

#### Procedure

Optimize the INSERT...SELECT operation as follows:

- If the table to be created is the Hive table, set the storage type to Parquet. This enables INSERT...SELECT statements to be run faster.

- Perform the INSERT...SELECT operation as a spark-sql user or spark user (if in Beeline/JDBCServer mode). In this way, it is no longer necessary to change the file owner repeatedly, accelerating the execution of INSERT...SELECT statements.

 **NOTE**

In Beeline/JDBCServer mode, the executor user is the same as the driver user. The driver user is a spark user because the driver is a part of JDBCServer service and started by a spark user. If the Beeline user is not a spark user, the file owner must be changed to the Beeline user (actual user) because the executor is unaware of the Beeline user.

- If many small files need to be queried, set spark.sql.files.maxPartitionBytes and spark.files.openCostInBytes to set the maximum size in bytes of partition and combine multiple small files in a partition to reduce file amount. This accelerates file renaming, ultimately enabling INSERT...SELECT statements to be run faster.

 **NOTE**

The preceding optimizations are not a one-size-fits-all solution. In the following scenario, it still takes long to perform the INSERT...SELECT operation:

The dynamic partitioned table contains many partitions.

## 24.7.2.5 Multiple JDBC Clients Concurrently Connecting to JDBCServer

### Scenario

Multiple clients can be connected to JDBCServer at the same time. However, if the number of concurrent tasks is too large, the default configuration of JDBCServer must be optimized to adapt to the scenario.

### Procedure

1. Set the fair scheduling policy of JDBCServer.  
The default scheduling policy of Spark is **FIFO**, which may cause a failure of short tasks in multi-task scenarios. Therefore, the fair scheduling policy must be used in multi-task scenarios to prevent task failure.
  - a. For details about how to configure Fair Scheduler in Spark, visit <http://spark.apache.org/docs/3.1.1/job-scheduling.html#scheduling-within-an-application>.
  - b. Configure Fair Scheduler on the JDBC client.
    - i. In the Beeline command line client or the code defined by JDBC, run the following statement:  
**PoolName** is a scheduling pool for Fair Scheduler.

```
SET spark.sql.thriftserver.scheduler.pool=PoolName;
```
    - ii. Run the SQL command. The Spark task will be executed in the preceding scheduling pool.
2. Set the **BroadCastHashJoin** timeout interval.  
There is a timeout parameter of **BroadCastHashJoin**. The task query fails if the query period exceeds the preset timeout interval. In multi-task scenarios,

the Spark task of BroadcastHashJoin may fail due to resource preemption. Therefore, it is necessary to modify the timeout interval in the **spark-defaults.conf** file of JDBCServer.

**Table 24-71** Parameter description

Parameter	Description	Default Value
spark.sql.broadcastTimeout	The timeout interval in the broadcast table of <b>BroadcastHashJoin</b> . If there are many concurrent tasks, set the parameter to a larger value or a negative number.	-1 (Numeral type. The actual value is 5 minutes.)

### 24.7.2.6 Optimizing Memory when Data Is Inserted into Dynamic Partitioned Tables

#### Scenario

When SparkSQL inserts data to dynamic partitioned tables, the more partitions there are, the more HDFS files a single task generates and the more memory metadata occupies. In this case, Garbage Collection (GC) is severe and Out of Memory (OOM) may occur.

Assume there are 10240 tasks and 2000 partitioned. Before the rename operation of HDFS files from a temporary directory to the target directory, there is about 29 GB FileStatus metadata.

#### Procedure

Insert **distribute by** followed by partition fields into dynamic partition statements.

For example:

```
insert into table store_returns partition (sr_returned_date_sk) select
sr_return_time_sk,sr_item_sk,sr_customer_sk,sr_demo_sk,sr_hdemo_sk,sr_addr_sk,
sr_store_sk,sr_reason_sk,sr_ticket_number,sr_return_quantity,sr_return_amt,sr_return_tax,sr_return_amt_inc_tax,sr_fee,sr_return_ship_cost,sr_refunded_cash,sr_reversed_charge,sr_store_credit,sr_net_loss,sr_returned_date_sk from $
{SOURCE}.store_returns distribute by sr_returned_date_sk;
```

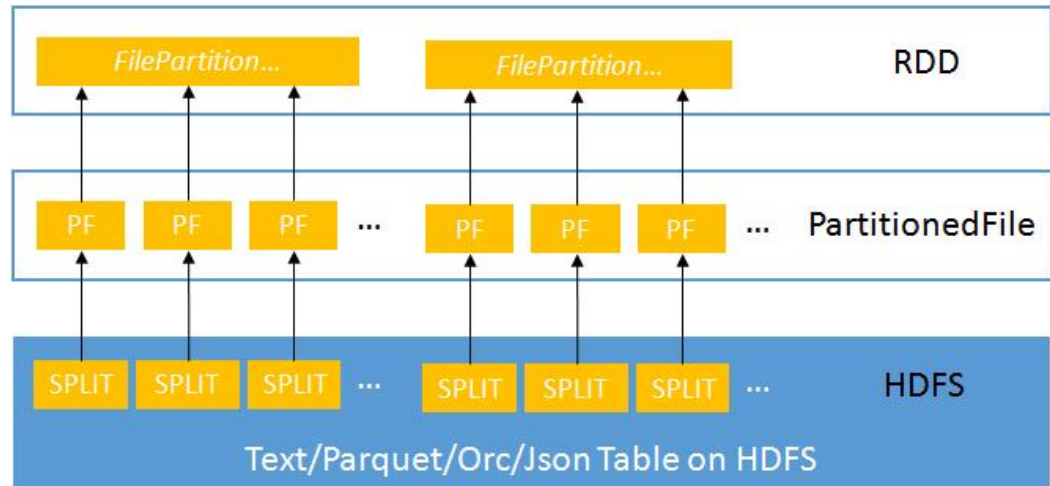
### 24.7.2.7 Optimizing Small Files

#### Scenario

A Spark SQL table may have many small files (far smaller than an HDFS block), each of which maps to a partition on the Spark by default. In other words, each small file is a task. In this way, Spark has to start many such tasks. If a shuffle operation is involved in the SQL logic, the number of hash buckets soars, severely hindering system performance.

In case of massive number of small files, when DataSource creates an RDD, it splits small files in the Spark SQL table to PartitionedFiles and then merges the PartitionedFiles to a partition to avoid generating too many hash buckets during the shuffle operation. See [Figure 24-8](#).

**Figure 24-8** Merging small files



## Procedure

If you want to enable small file optimization, configure the **spark-defaults.conf** file on the Spark client.

**Table 24-72** Parameter description

Parameter	Description	Default Value
spark.sql.files.maxPartitionBytes	The maximum number of bytes that can be packed into a single partition when a file is read. Unit: byte	134217728 (128 MB)
spark.files.openCostInBytes	The estimated cost to open a file, measured by the number of bytes that can be scanned in the same time. This is used when putting multiple files into a partition. It is better to over estimate, then the partitions with small files will be faster than partitions with larger files.	4 MB

### 24.7.2.8 Optimizing the Aggregate Algorithms

#### Scenario

Spark SQL supports hash aggregate algorithm. Namely, use fast aggregate hashmap as cache to improve aggregate performance. The hashmap replaces the



previous ColumnarBatch to avoid performance problems caused by the wide mode (multiple key or value fields) of an aggregate table.

## Procedure

If you want to enable optimization of aggregate algorithm, configure following parameters in the **spark-defaults.conf** file on the Spark client.

**Table 24-73** Parameter description

Parameter	Description	Default Value
spark.sql.codegen.aggregate.map.twolevel.enabled	Specifies whether to enable aggregation algorithm optimization. <ul style="list-style-type: none"> <li>• <b>true</b>: Enable</li> <li>• <b>false</b>: Disable</li> </ul>	true

### 24.7.2.9 Optimizing Datasource Tables

#### Scenario

Save the partition information about the datasource table to the Metastore and process partition information in the Metastore.

- Optimize the datasource tables, support syntax such as adding, deletion, and modification in the table based on partitions, improving compatibility with Hive.
- Support statements of partition tailoring and push down to the Metastore to filter unmatched partitions.

Example:

```
select count(*) from table where partCol=1; //partCol (partition column)
```

You need only to process data corresponding to partCol=1 when performing the TableScan operation in the physical plan.

#### Procedure

If you want to enable Datasource table optimization, configure the **spark-defaults.conf** file on the Spark client.

**Table 24-74** Parameter description

Parameter	Description	Default Value
spark.sql.hive.manageFilesourcePartitions	<p>Specifies whether to enable Metastore partition management (including datasource tables and converted Hive).</p> <ul style="list-style-type: none"> <li>• <b>true</b> indicates enabling Metastore partition management. In this case, datasource tables are stored in Hive and Metastore is used to tailor partitions in query statements.</li> <li>• <b>false</b> indicates disabling Metastore partition management.</li> </ul>	true
spark.sql.hive.metastorePartitionPruning	<p>Specifies whether to support pushing down predicate to Hive Metastore.</p> <ul style="list-style-type: none"> <li>• <b>true</b> indicates supporting pushing down predicate to Hive Metastore. Only the predicate of Hive tables is supported.</li> <li>• <b>false</b> indicates not supporting pushing down predicate to Hive Metastore.</li> </ul>	true
spark.sql.hive.filesourcePartitionFileCacheSize	<p>The cache size of the partition file metadata in the memory.</p> <p>All tables share a cache that can use up to specified num bytes for file metadata.</p> <p>This parameter is valid only when <b>spark.sql.hive.manageFilesourcePartitions</b> is set to <b>true</b>.</p>	250 * 1024 * 1024
spark.sql.hive.convertMetastoreOrc	<p>The processing approach of ORC tables.</p> <ul style="list-style-type: none"> <li>• <b>false</b>: Spark SQL uses Hive SerDe to process ORC tables.</li> <li>• <b>true</b>: Spark SQL uses the Spark built-in mechanism to process ORC tables.</li> </ul>	true

### 24.7.2.10 Merging CBO

#### Scenario

Spark SQL supports rule-based optimization by default. However, the rule-based optimization cannot ensure that Spark selects the optimal query plan. Cost-Based Optimizer (CBO) is a technology that intelligently selects query plans for SQL statements. After CBO is enabled, the CBO optimizer performs a series of

estimations based on the table and column statistics to select the optimal query plan.

## Procedure

Perform the following steps to enable CBO:

1. You need to run corresponding SQL commands to collect required table and column statistics.

SQL commands are as follows (to be chosen as required):

- Generate table-level statistics (table scanning):

***ANALYZE TABLE src COMPUTE STATISTICS***

This command generates **sizeInBytes** and **rowCount**.

When you use the ANALYZE statement to collect statistics, sizes of tables not from HDFS cannot be calculated.

- Generate table-level statistics (no table scanning):

***ANALYZE TABLE src COMPUTE STATISTICS NOSCAN***

This command generates only **sizeInBytes**. Compared with the originally generated **sizeInBytes** and **rowCount** if the **sizeInBytes** remains unchanged, **rowCount** (if any) reserves. Otherwise, **rowCount** is cleared.

- Generate column-level statistics:

***ANALYZE TABLE src COMPUTE STATISTICS FOR COLUMNS a, b, c***

This command generates column statistics and updates table statistics for consistency. Statistics of complicated data types (such as Seq and Map) and HiveStringType cannot be generated.

- Display statistics:

***DESC FORMATTED src***

This command displays *xxx* bytes and *xxx* rows in **Statistics** to indicate table-level statistics. You can also run the following command to display column statistics:

***DESC FORMATTED src a***

**Limitation:** The current statistics collection does not support statistics for partition levels for partitioned tables.

2. Configure parameters in [Table 24-75](#) in the **spark-defaults.conf** file on the Spark client.

**Table 24-75** Parameter description

Parameter	Description	Default Value
spark.sql.cbo.enabled	<p>The switch to enable or disable CBO.</p> <ul style="list-style-type: none"> <li>• <b>true:</b> Enable</li> <li>• <b>false:</b> Disable</li> </ul> <p>To enable this function, ensure that statistics of related tables and columns are generated.</p>	false

Parameter	Description	Default Value
spark.sql.cbo.joinReorder.enabled	<p>Specifies whether to automatically adjust the sequence of consecutive inner joins by using CBO.</p> <ul style="list-style-type: none"> <li>• <b>true</b>: Enable</li> <li>• <b>false</b>: Disable</li> </ul> <p>To enable this function, ensure that statistics of related tables and columns are generated and CBO is enabled.</p>	false
spark.sql.cbo.joinReorder.dp.threshold	<p>Specifies the threshold of the number of tables that the sequence of consecutive inner joins is automatically adjusted by CBO.</p> <p>If the threshold is exceeded, the sequence of joins is not adjusted.</p>	12

### 24.7.2.11 Optimizing SQL Query of Data of Multiple Sources

#### Scenario

This section describes how to enable or disable the query optimization for inter-source complex SQL.

#### Procedure

- (Optional) Prepare for connecting to the MPPDB data source.  
If the data source to be connected is MPPDB, a class name conflict occurs because the MPPDB Driver file **gsjdbc4.jar** and the Spark JAR package **gsjdbc4-VXXXRXXXCXXSPCXXX.jar** contain the same class name. Therefore, before connecting to the MPPDB data source, perform the following steps:
  - a. Move **gsjdbc4-VXXXRXXXCXXSPCXXX.jar** from Spark. Spark running does not depend on this JAR file. Therefore, moving this JAR file to another directory (for example, the **/tmp** directory) will not affect Spark running.
    - i. Log in to the Spark server and move **gsjdbc4-VXXXRXXXCXXSPCXXX.jar** from the **`\${BIGDATA\_HOME}/FusionInsight\_Spark2x\_8.0.2.1/install/FusionInsight-Spark2x-2.4.5/spark/jars** directory.
    - ii. Log in to the Spark client host and move **gsjdbc4-VXXXRXXXCXXSPCXXX.jar** from the **/opt/client/Spark2x/spark/jars** directory.
  - b. Obtain the MPPDB Driver file **gsjdbc4.jar** from the MPPDB installation package and upload the file to the following directories:

- `/${BIGDATA_HOME}/FusionInsight_Spark2x_8.0.2.1/install/FusionInsight-Spark2x-2.4.5/spark/jars` on the Spark server.
- `/opt/client/Spark2x/spark/jars` on the Spark client.
- c. Update the `/user/spark2x/jars/8.0.2.1/spark-archive-2x.zip` package stored in the HDFS.

 NOTE

The version 8.0.2.1 is used as an example. Replace it with the actual version number.

- i. Log in to the node where the client is installed as a client installation user. Run the following command to switch to the client installation directory, for example, `/opt/client`:  
**cd /opt/client**
- ii. Run the following command to configure environment variables:  
**source bigdata\_env**
- iii. If the cluster is in security mode, run the following command to get authenticated:  
**kinit Component service user**
- iv. Run the following commands to create the temporary file `./tmp`, obtain `spark-archive-2x.zip` from HDFS, and decompress it to the `tmp` directory:  
**mkdir tmp**  
**hdfs dfs -get /user/spark2x/jars/8.0.2.1/spark-archive-2x.zip ./**  
**unzip spark-archive-2x.zip -d ./tmp**
- v. Switch to the `tmp` directory, delete the `gsjdbc4-VXXXRXXXCXXSPCXXX.jar` file, upload the MPPDB Driver file `gsjdbc4.jar` to the `tmp` directory, and run the following command to compress the file again:  
**zip -r spark-archive-2x.zip \*.jar**
- vi. Delete `spark-archive-2x.zip` from the HDFS and update the `spark-archive-2x.zip` package generated in `c.v` to the `/user/spark2x/jars/8.0.2.1/` directory in the HDFS.  
**hdfs dfs -rm /user/spark2x/jars/8.0.2.1/spark-archive-2x.zip**  
**hdfs dfs -put ./spark-archive-2x.zip /user/spark2x/jars/8.0.2.1**
- d. Restart the Spark service. After the Spark service is restarted, restart the Spark client.
- Enable the optimization function.  
For all modules that support query pushdown, you can run the **SET** command on the `spark-beeline` client to enable the cross-source query optimization function. By default, the function is disabled.  
Pushdown configurations can be performed in dimensions of global, data sources, and tables. Commands are as follows:
  - Global (valid for all data sources):  
**SET spark.sql.datasource.jdbc = project,aggregate,orderby-limit**

- Data sources:  
**SET spark.sql.datasource.\${url} = project,aggregate,orderby-limit**
- Tables:  
**SET spark.sql.datasource.\${url}.\${table} = project,aggregate,orderby-limit**

When you run the **SET** command to configure preceding parameters, you are allowed to specify multiple pushdown modules and separate them by commas. The following table lists parameters of corresponding pushdown modules.

**Table 24-76** Parameters of modules

Module	Parameter Value in the SET Command
project	project
aggregate	aggregate
order by, limit over project or aggregate	orderby-limit

The following is a statement for creating an external table of MySQL:

```
create table if not exists pdmysql using org.apache.spark.sql.jdbc
options(driver "com.mysql.jdbc.Driver", url "jdbc:mysql://ip2:3306/test",
user "hive", password "123456", dbtable "mysqldata");
```

In the preceding statement:

- `${url}` = `jdbc:mysql://ip2:3306/test`
- `${table}` = `mysqldata`

 **NOTE**

- On the right of the equal sign (=) is the operators (separated by commas) to be enabled by pushdown.
- Priority: table > data source > global. If the table switch is set, the global switch of the data source is invalid for the table. If a data source switch is set, the global switch is invalid for the data source.
- The equal sign (=) is not allowed in URL. Equal signs (=) are automatically deleted in the SET clause.
- After multiple SET operations, results with different keys will not overwrite each other.

- Add functions that support query pushdown.

In addition to query pushdown of mathematical, time, and string functions such as `abs()`, `month()`, and `length()`, you can run the **SET** command to add a data source that supports query pushdown. Run the following command on the Spark-beeline client:

```
SET park.sql.datasource.${datasource}.functions = fun1,fun2
```

- Reset the configuration set by the **SET** command.

Currently, you can only run the **RESET** command on the **spark-beeline** client to cancel all **SET** content. After running the **RESET** command, all values in the

**SET** command will be cleared. Exercise caution when performing this operation.

The **SET** command is valid in the current session on the client. After the client is shut down, the content in the **SET** command turns invalid.

Alternatively, change the value of **spark.sql.locale.support** in the **spark-defaults.conf** file to **true**.

- Support pushdown of Hive data sources in Portuguese.  
Currently, the table names and field names of Spark and Hive support Portuguese. Users can run the **SET** command to add the function of pushdown in Portuguese. Run the following command on the Spark-SQL client:

```
set spark.sql.locale.support = true
```

Alternatively, change the value of **spark.sql.locale.support** in the **spark-defaults.conf** file to **true**.

## Precautions

Only MySQL, MPPDB, Hive, oracle, and PostgreSQL data sources are supported.

### 24.7.2.12 SQL Optimization for Multi-level Nesting and Hybrid Join

#### Scenario

This section describes the optimization suggestions for SQL statements in multi-level nesting and hybrid join scenarios.

#### Prerequisites

The following provides an example of complex query statements:

```
elect
s_name,
count(1) as numwait
from (
select s_name from (
select
s_name,
t2.l_orderkey,
l_suppkey,
count_suppkey,
max_suppkey
from
test2 t2 right outer join (
select
s_name,
l_orderkey,
l_suppkey from (
select
s_name,
t1.l_orderkey,
l_suppkey,
count_suppkey,
max_suppkey
from
test1 t1 join (
select
s_name,
l_orderkey,
```

```

L_suppkey
from
orders o join (
select
s_name,
L_orderkey,
L_suppkey
from
nation n join supplier s
on
s.s_nationkey = n.n_nationkey
and n.n_name = 'SAUDI ARABIA'
join lineitem l
on
s.s_suppkey = l.l_suppkey
where
l.l_receiptdate > l.l_commitdate
and l.l_orderkey is not null
) l1 on o.o_orderkey = l1.l_orderkey and o.o_orderstatus = 'F'
) l2 on l2.l_orderkey = t1.l_orderkey
) a
where
(count_suppkey > 1)
or ((count_suppkey=1)
and (L_suppkey <> max_suppkey))
) l3 on l3.l_orderkey = t2.l_orderkey
) b
where
(count_suppkey is null)
or ((count_suppkey=1)
and (L_suppkey = max_suppkey))
) c
group by
s_name
order by
numwait desc,
s_name
limit 100;

```

## Procedure

### Step 1 Analyze business.

Analyze business to determine whether SQL statements can be simplified through measures, for example, by combining tables to reduce the number of nesting levels and join times.

### Step 2 If the SQL statements cannot be simplified, configure the driver memory.

- If SQL statements are executed through spark-submit or spark-sql, go to [Step 3](#).
- If SQL statements are executed through spark-beeline, go to [Step 4](#).

### Step 3 During execution of SQL statements, specify the **driver-memory** parameter. An example of SQL statements is as follows:

```
/spark-sql --master=local[4] --driver-memory=512M -f /tpch.sql
```

### Step 4 Before running SQL statements, change the memory size as the MRS cluster administrator.

1. Log in to FusionInsight Manager and choose **Cluster** > *Name of the desired cluster* > **Services** > **Spark2x** > **Configurations**.
2. On the displayed page, click **All Configurations** and search for **SPARK\_DRIVER\_MEMORY**.



3. Modify the **SPARK\_DRIVER\_MEMORY** parameter value to increase the memory size. The parameter value consists of two parts: memory size (an integer) and the unit (M or G), for example, **512M**.

----End

## Reference

In the event of insufficient driver memory, the following error may be displayed during the query:

```
2018-02-11 09:13:14,683 | WARN | Executor task launch worker for task 5 | Calling spill() on
RowBasedKeyValueBatch. Will not spill but return 0. |
org.apache.spark.sql.catalyst.expressions.RowBasedKeyValueBatch.spill(RowBasedKeyValueBatch.java:173)
2018-02-11 09:13:14,682 | WARN | Executor task launch worker for task 3 | Calling spill() on
RowBasedKeyValueBatch. Will not spill but return 0. |
org.apache.spark.sql.catalyst.expressions.RowBasedKeyValueBatch.spill(RowBasedKeyValueBatch.java:173)
2018-02-11 09:13:14,704 | ERROR | Executor task launch worker for task 2 | Exception in task 2.0 in stage
1.0 (TID 2) | org.apache.spark.internal.Logging$class.logError(Logging.scala:91)
java.lang.OutOfMemoryError: Unable to acquire 262144 bytes of memory, got 0
 at org.apache.spark.memory.MemoryConsumer.allocateArray(MemoryConsumer.java:100)
 at org.apache.spark.unsafe.map.BytesToBytesMap.allocate(BytesToBytesMap.java:791)
 at org.apache.spark.unsafe.map.BytesToBytesMap.<init>(BytesToBytesMap.java:208)
 at org.apache.spark.unsafe.map.BytesToBytesMap.<init>(BytesToBytesMap.java:223)
 at
org.apache.spark.sql.execution.UnsafeFixedWidthAggregationMap.<init>(UnsafeFixedWidthAggregationMap.j
ava:104)
 at
org.apache.spark.sql.execution.aggregate.HashAggregateExec.createHashMap(HashAggregateExec.scala:307)
 at org.apache.spark.sql.catalyst.expressions.GeneratedClass
$GeneratedIterator.agg_doAggregateWithKeys$(Unknown Source)
 at org.apache.spark.sql.catalyst.expressions.GeneratedClass$GeneratedIterator.processNext(Unknown
Source)
 at org.apache.spark.sql.execution.BufferedRowIterator.hasNext(BufferedRowIterator.java:43)
 at org.apache.spark.sql.execution.WholeStageCodegenExec$$anonfun$8$$anon
$1.hasNext(WholeStageCodegenExec.scala:381)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:408)
 at
org.apache.spark.shuffle.sort.BypassMergeSortShuffleWriter.write(BypassMergeSortShuffleWriter.java:126)
 at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:96)
 at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:53)
 at org.apache.spark.scheduler.Task.run(Task.scala:99)
 at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:325)
 at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
 at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
 at java.lang.Thread.run(Thread.java:748)
```

## 24.7.3 Spark Streaming Tuning

### Scenario

Streaming is a mini-batch streaming processing framework that features second-level delay and high throughput. To optimize Streaming is to improve its throughput while maintaining second-level delay so that more data can be processed per unit time.

#### NOTE

This section applies to the scenario where the input data source is Kafka.

## Procedure

A simple streaming processing system consists of a data source, a receiver, and a processor. The data source is Kafka, the receiver is the Kafka data source receiver of Streaming, and the processor is Streaming.

Streaming optimization is to optimize the performance of the three components.

- **Data source optimization**

In actual application scenarios, the data source stores the data in the local disks to ensure the error tolerance of the data. However, the calculation results of the Streaming are stored in the memory, and the data source may become the largest bottleneck of the streaming system.

Kafka can be optimized from the following aspects:

- Use Kafka-0.8.2 or later version that allows you to use new Producer APIs in asynchronous mode.
- Configure multiple Broker directories, multiple I/O threads, and a proper number of partitions for a topic.

For details, see section **Performance Tuning** in the Kafka open source documentation at <http://kafka.apache.org/documentation.html>.

- **Receiver optimization**

Streaming has multiple data source receivers, such as Kafka, Flume, MQTT, and ZeroMQ. Kafka has the most receiver types and is the most mature receiver.

Kafka provides three types of receiver APIs:

- `KafkaReceiver` directly receives Kafka data. If the process is abnormal, data may be lost.
- `ReliableKafkaReceiver` receives data displacement through ZooKeeper records.
- `DirectKafka` reads data from each partition of Kafka through the RDD, ensuring high reliability.

According to the implementation mechanism and test results, `DirectKafka` provides better performance than the other two APIs. Therefore, the `DirectKafka` API is recommended to implement the receiver.

For details about the Kafka receivers and their optimization methods, see the Kafka open source documentation at <http://kafka.apache.org/documentation.html>.

- **Processor optimization**

The bottom layer of Spark Streaming is executed by Spark. Therefore, most optimization measures for Spark can also be applied to Spark Streaming. The following is an example:

- Data serialization
- Memory configuration
- Configuring DOP
- Using the external shuffle service to improve performance

 NOTE

Higher performance of Spark Streaming indicates lower overall reliability. Examples:  
If `spark.streaming.receiver.writeAheadLog.enable` is set to `false`, disk I/Os are reduced and performance is improved. However, because WAL is disabled, data is lost during fault recovery.

Therefore, do not disable configuration items that ensure data reliability in production environments during Spark Streaming tuning.

- **Log archive optimization**

The `spark.eventLog.group.size` parameter is used to group **JobHistory** logs of an application based on the specified number of jobs. Each group creates a file recording log to prevent **JobHistory** reading failures caused by an oversized log generated during the long-term running of the application. If this parameter is set to `0`, logs are not grouped.

Most Spark Streaming jobs are small jobs and are generated at a high speed. As a result, frequent grouping is performed and a large number of small log files are generated, consuming disk I/O resources. You are advised to increase the parameter value to, for example, `1000` or greater.

## 24.8 Common Issues About Spark2x

### 24.8.1 Spark Core

#### 24.8.1.1 How Do I View Aggregated Spark Application Logs?

##### Question

How do I view the aggregated container logs on the page when the log aggregation function is enabled on YARN?

##### Answer

For details, see [Viewing Aggregated Container Logs on the Web UI](#).

#### 24.8.1.2 Why Is the Return Code of Driver Inconsistent with Application State Displayed on ResourceManager WebUI?

##### Question

Communication between ApplicationMaster and ResourceManager remains abnormal for a long time. Why is the driver return code inconsistent with application status on ResourceManager WebUI?

##### Answer

In yarn-client mode, Spark Driver and ApplicationMaster run as two independent processes. When Driver exits, it notifies ApplicationMaster to call the unregister API to deregister itself with ResourceManager.

This is a remote call and susceptible to network faults. If there exists a network fault, ApplicationMaster uses the retry mechanism of the Yarn client to try again. If the network is recovered before the maximum number of retries is reached, ApplicationMaster exits gracefully.

If the number and duration of retries are reached, ApplicationMaster fails to deregister itself, and ResourceManager declares ApplicationMaster to have exited forcibly and tries to restart ApplicationMaster. After the restart, if ApplicationMaster fails to connect to the exited Driver, ResourceManager flags the Application being failed.

This problem rarely occurs and it does not impact the display of application states by SparkSQL. You can also increase the number of Yarn client connections and the connection duration to reduce the probability of this event. For details about the configuration, see <http://hadoop.apache.org/docs/r3.1.1/hadoop-yarn/hadoop-yarn-common/yarn-default.xml>.

### 24.8.1.3 Why Cannot Exit the Driver Process?

#### Question

Why cannot exit the Driver process after running the **yarn application -kill applicationID** command to stop the Spark Streaming application?

#### Answer

Running the **yarn application -kill applicationID** command can only stop the SparkContext corresponding to Spark Streaming application, but cannot exit the current Driver process. If there are other permanent threads in the Driver process (for example, the spark shell is continually checking command input or Spark Streaming is continually reading data form data source), the Driver process will not be killed when the SparkContext is stopped. To exit the Driver process, you are advised to run the **kill -9 pid** command to kill the current Driver process by hand.

### 24.8.1.4 Why Does FetchFailedException Occur When the Network Connection Is Timed out

#### Question

On a large cluster of 380 nodes, run the ScalaSort test case in the HiBench test that runs the 29T data, and configure Executor as **--executor-cores 4**. The following abnormality is displayed:

```
org.apache.spark.shuffle.FetchFailedException: Failed to connect to /192.168.114.12:23242
 at
 org.apache.spark.storage.ShuffleBlockFetcherIterator.throwFetchFailedException(ShuffleBlockFetcherIterator.scala:321)
 at org.apache.spark.storage.ShuffleBlockFetcherIterator.next(ShuffleBlockFetcherIterator.scala:306)
 at org.apache.spark.storage.ShuffleBlockFetcherIterator.next(ShuffleBlockFetcherIterator.scala:51)
 at scala.collection.Iterator$$anon$11.next(Iterator.scala:328)
 at scala.collection.Iterator$$anon$13.hasNext(Iterator.scala:371)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:327)
 at org.apache.spark.util.CompletionIterator.hasNext(CompletionIterator.scala:32)
 at org.apache.spark.InterruptibleIterator.hasNext(InterruptibleIterator.scala:39)
 at org.apache.spark.util.collection.ExternalSorter.insertAll(ExternalSorter.scala:217)
 at org.apache.spark.shuffle.hash.HashShuffleReader.read(HashShuffleReader.scala:102)
```

```

at org.apache.spark.rdd.ShuffledRDD.compute(ShuffledRDD.scala:90)
at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:301)
at org.apache.spark.rdd.RDD.iterator(RDD.scala:265)
at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:38)
at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:301)
at org.apache.spark.rdd.RDD.iterator(RDD.scala:265)
at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:38)
at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:301)
at org.apache.spark.rdd.RDD.iterator(RDD.scala:265)
at org.apache.spark.rdd.UnionRDD.compute(UnionRDD.scala:87)
at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:301)
at org.apache.spark.rdd.RDD.iterator(RDD.scala:265)
at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:73)
at org.apache.spark.scheduler.ShuffleMapTask.runTask(ShuffleMapTask.scala:41)
at org.apache.spark.scheduler.Task.run(Task.scala:87)
at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:213)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
at java.lang.Thread.run(Thread.java:745)
Caused by: java.io.IOException: Failed to connect to /192.168.114.12:23242
at org.apache.spark.network.client.TransportClientFactory.createClient(TransportClientFactory.java:214)
at org.apache.spark.network.client.TransportClientFactory.createClient(TransportClientFactory.java:167)
at org.apache.spark.network.netty.NettyBlockTransferService$$anon
$1.createAndStart(NettyBlockTransferService.scala:91)
at org.apache.spark.network.shuffle.RetryingBlockFetcher.fetchAllOutstanding(RetryingBlockFetcher.java:
140)
at org.apache.spark.network.shuffle.RetryingBlockFetcher.access$200(RetryingBlockFetcher.java:43)
at org.apache.spark.network.shuffle.RetryingBlockFetcher$1.run(RetryingBlockFetcher.java:170)
at java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)
at java.util.concurrent.FutureTask.run(FutureTask.java:266)
... 3 more
Caused by: java.net.ConnectException: Connection timed out: /192.168.114.12:23242
at sun.nio.ch.SocketChannelImpl.checkConnect(Native Method)
at sun.nio.ch.SocketChannelImpl.finishConnect(SocketChannelImpl.java:717)
at io.netty.channel.socket.nio.NioSocketChannel.doFinishConnect(NioSocketChannel.java:224)
at io.netty.channel.nio.AbstractNioChannel$AbstractNioUnsafe.finishConnect(AbstractNioChannel.java:
289)
at io.netty.channel.nio.NioEventLoop.processSelectedKey(NioEventLoop.java:528)
at io.netty.channel.nio.NioEventLoop.processSelectedKeysOptimized(NioEventLoop.java:468)
at io.netty.channel.nio.NioEventLoop.processSelectedKeys(NioEventLoop.java:382)
at io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:354)
at io.netty.util.concurrent.SingleThreadEventExecutor$2.run(SingleThreadEventExecutor.java:111)
... 1 more

```

## Answer

When an application is run, configure the Executor parameter as **--executor-cores 4**. The degree of parallelism (DOP) is high in a single process, resulting in that the IO is highly occupied and the task works slowly.

```
16/02/26 10:04:53 INFO TaskSetManager: Finished task 2139.0 in stage 1.0 (TID 151149) in 376455 ms on
10-196-115-2 (694/153378)
```

Because running a single task takes more than 6 minutes. The network connection is timed out and the running task fails.

Set the number of cores as 1, which is **--executor-cores 1**. A task is executed smoothly in proper time (within 15s).

```
16/02/29 02:24:46 INFO TaskSetManager: Finished task 59564.0 in stage 1.0 (TID 208574) in 15088 ms on
10-196-115-6 (59515/153378)
```

Therefore, to process the task of network connection timed out and avoid such error, you can reduce the core number of a single Executor.

### 24.8.1.5 How to Configure Event Queue Size If Event Queue Overflows?

#### Question

How to configure the event queue size if the following Driver log information is displayed indicating that the event queue overflows?

- **Common applications**  
Dropping SparkListenerEvent because no remaining room in event queue.  
This likely means one of the SparkListeners is too slow and cannot keep up with the rate at which tasks are being started by the scheduler.
- **Spark Streaming applications**  
Dropping StreamingListenerEvent because no remaining room in event queue.  
This likely means one of the StreamingListeners is too slow and cannot keep up with the rate at which events are being started by the scheduler.

#### Answer

1. Stop the application. Set the configuration option **spark.event.listener.logEnable** in the Spark configuration file **spark-defaults.conf** to **true**. And set the configuration option **spark.eventQueue.size** to **1000W**. If you need to control the logging rate (in milliseconds), also change the value of the configuration option **spark.event.listener.logRate**.  
By default, the logging rate is 1000 ms, which means that one log is printed out every 1000 ms.
2. Start the application.  
The following log information is displayed, including the event consumption rate, event production rate, and **MaxSize** (maximum size of messages in the queue).  
INFO LiveListenerBus: [SparkListenerBus]:16044 events are consumed in 5000 ms.  
INFO LiveListenerBus: [SparkListenerBus]:51381 events are produced in 5000 ms, eventQueue still has 86417 events, MaxSize: 171764.
3. Change the value of the configuration option **spark.eventQueue.size** in the Spark configuration file **spark-defaults.conf** based on the **MaxSize** in the log information.  
For example, if **MaxSize** is 250000, the appropriate message queue size is 300000.

### 24.8.1.6 What Can I Do If the `getApplicationReport` Exception Is Recorded in Logs During Spark Application Execution and the Application Does Not Exit for a Long Time?

#### Question

During Spark application execution, if the driver fails to connect to ResourceManager, the following error is reported and it does not exit for a long time. What can I do?

```
16/04/23 15:31:44 INFO RetryInvocationHandler: Exception while invoking getApplicationReport of class ApplicationClientProtocolPBClientImpl over 37 after 1 fail over attempts. Trying to fail over after sleeping for 44160ms.
java.net.ConnectException: Call From vm1/192.168.39.30 to vm1:8032 failed on connection exception: java.net.ConnectException: Connection refused; For more details see: http://wiki.apache.org/hadoop/ConnectionRefused
```

## Answer

In Spark, there is a scheduled thread that listens to the status of ApplicationMaster by connecting to ResourceManager. The connection to the ResourceManager times out. As a result, the preceding error is reported and the system keeps trying to connect to the ResourceManager. In the ResourceManager, the number of retry times is limited. By default, the number of retry times is 30 and the retry interval is about 30 seconds. The preceding error is reported during each retry. The driver exits only after the number of times is exceeded.

**Table 24-77** describes the retry-related configuration items in the ResourceManager.

**Table 24-77** Parameter description

Parameter	Description	Default Value
yarn.resourcemanager.connect.max-wait.ms	Maximum waiting time for connecting to the ResourceManager.	900000
yarn.resourcemanager.connect.retry-interval.ms	Interval for reconnecting to the ResourceManager.	30000

Number of retries (**yarn.resourcemanager.connect.max-wait.ms/ yarn.resourcemanager.connect.retry-interval.ms**) = Maximum waiting time for connecting to the ResourceManager/Interval for reconnecting to the ResourceManager

On the Spark client, modify the **conf/yarn-site.xml** file to add and configure **yarn.resourcemanager.connect.max-wait.ms** and **yarn.resourcemanager.connect.retry-interval.ms**. In this way, the number of retry times can be changed, and the Spark application can exit in advance.

### 24.8.1.7 What Can I Do If "Connection to ip:port has been quiet for xxx ms while there are outstanding requests" Is Reported When Spark Executes an Application and the Application Ends?

#### Question

When Spark executes an application, an error similar to the following is reported and the application ends. What can I do?

```
2016-04-20 10:42:00,557 | ERROR | [shuffle-server-2] | Connection to 10-91-8-208/10.18.0.115:57959 has
been quiet for 180000 ms while there are outstanding requests. Assuming connection is dead; please adju
st spark.network.timeout if this is wrong. |
org.apache.spark.network.server.TransportChannelHandler.userEventTriggered(TransportChannelHandler.java:
128)
2016-04-20 10:42:00,558 | ERROR | [shuffle-server-2] | Still have 1 requests outstanding when connection
from 10-91-8-208/10.18.0.115:57959 is closed | org.apache.spark.network.client.TransportResponseHandl
er.channelUnregistered(TransportResponseHandler.java:102)
2016-04-20 10:42:00,562 | WARN | [yarn-scheduler-ask-am-thread-pool-160] | Error sending message
[message = DoShuffleClean(application_1459995017785_0108,319)] in 1 attempts |
org.apache.spark.Logging$clas
```

```
s.logWarning(Logging.scala:92)
java.io.IOException: Connection from 10-91-8-208/10.18.0.115:57959 closed
 at
 org.apache.spark.network.client.TransportResponseHandler.channelUnregistered(TransportResponseHandler.java:104)
 at
 org.apache.spark.network.server.TransportChannelHandler.channelUnregistered(TransportChannelHandler.java:94)
 at
 io.netty.channel.AbstractChannelHandlerContext.invokeChannelUnregistered(AbstractChannelHandlerContext.java:158)
 at
 io.netty.channel.AbstractChannelHandlerContext.fireChannelUnregistered(AbstractChannelHandlerContext.java:144)
 at
 io.netty.channel.ChannelInboundHandlerAdapter.channelUnregistered(ChannelInboundHandlerAdapter.java:53)
 at
 io.netty.channel.AbstractChannelHandlerContext.invokeChannelUnregistered(AbstractChannelHandlerContext.java:158)
 at
 io.netty.channel.AbstractChannelHandlerContext.fireChannelUnregistered(AbstractChannelHandlerContext.java:144)
 at
 io.netty.channel.ChannelInboundHandlerAdapter.channelUnregistered(ChannelInboundHandlerAdapter.java:53)
 at
 io.netty.channel.AbstractChannelHandlerContext.invokeChannelUnregistered(AbstractChannelHandlerContext.java:158)
 at
 io.netty.channel.AbstractChannelHandlerContext.fireChannelUnregistered(AbstractChannelHandlerContext.java:144)
 at
 io.netty.channel.ChannelInboundHandlerAdapter.channelUnregistered(ChannelInboundHandlerAdapter.java:53)
 at
 io.netty.channel.AbstractChannelHandlerContext.invokeChannelUnregistered(AbstractChannelHandlerContext.java:158)
 at
 io.netty.channel.AbstractChannelHandlerContext.fireChannelUnregistered(AbstractChannelHandlerContext.java:144)
 at
 io.netty.channel.ChannelInboundHandlerAdapter.channelUnregistered(ChannelInboundHandlerAdapter.java:53)
 at
 io.netty.channel.AbstractChannelHandlerContext.invokeChannelUnregistered(AbstractChannelHandlerContext.java:158)
 at
 io.netty.channel.AbstractChannelHandlerContext.fireChannelUnregistered(AbstractChannelHandlerContext.java:144)
 at io.netty.channel.DefaultChannelPipeline.fireChannelUnregistered(DefaultChannelPipeline.java:739)
 at io.netty.channel.AbstractChannel$AbstractUnsafe$8.run(AbstractChannel.java:659)
 at io.netty.util.concurrent.SingleThreadEventExecutor.runAllTasks(SingleThreadEventExecutor.java:357)
 at io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:357)
 at io.netty.util.concurrent.SingleThreadEventExecutor$2.run(SingleThreadEventExecutor.java:111)
 at java.lang.Thread.run(Thread.java:745)
2016-04-20 10:42:00,573 | INFO | [dispatcher-event-loop-14] | Starting task 177.0 in stage 1492.0 (TID 1996351, linux-254, PROCESS_LOCAL, 2106 bytes) | org.apache.spark.Logging$class.logInfo(Logging.scala:59)
2016-04-20 10:42:00,574 | INFO | [task-result-getter-0] | Finished task 85.0 in stage 1492.0 (TID 1996259) in 191336 ms on linux-254 (106/3000) | org.apache.spark.Logging$class.logInfo(Logging.scala:59)
2016-04-20 10:42:00,811 | ERROR | [Yarn application state monitor] | Yarn application has already exited with state FINISHED! | org.apache.spark.Logging$class.logError(Logging.scala:75)
```

## Answer

Symptom: The value of **spark.rpc.io.connectionTimeout** is less than the value of **spark.rpc.askTimeout**. In full GC or network delay scenarios, when the channel reaches the expiration time and still receives no response, the channel is terminated. When detecting that the channel is terminated, the AM considers the driver as disconnected, and the entire application is stopped.

Solution: Set the parameter in the **spark-defaults.conf** file on the Spark client by running the **set** command. During parameter configuration, ensure that the channel expiration time (**spark.rpc.io.connectionTimeout**) is greater than or equal to the RPC response timeout (**spark.rpc.askTimeout**).



**Table 24-78** Parameter description

Parameter	Description	Default Value
spark.rpc.askTimeout	RPC response timeout. If this parameter is not set, the value of <b>spark.network.timeout</b> is used by default.	120s

### 24.8.1.8 Why Do Executors Fail to be Removed After the NodeManager Is Shut Down?

#### Question

If the NodeManager is shut down with the Executor dynamic allocation enabled, the Executors on the node where the NodeManager is shut down fail to be removed from the driver page after the idle time expires.

#### Answer

When the ResourceManager detects that the NodeManager is shut down, the driver has requested to kill Executors due to idle time expiry. However, the Executors cannot actually be killed because the NodeManager is shut down. The driver cannot detect the LOST events of these Executors and does not remove Executors from its Executor list. Therefore, the Executors are not removed from the driver page. This phenomenon is normal after the YARN NodeManager is shut down. The Executors will be removed after the NodeManager restarts.

### 24.8.1.9 What Can I Do If the Message "Password cannot be null if SASL is enabled" Is Displayed?

#### Question

ExternalShuffle is enabled for the application that runs Spark. Task loss occurs in the application because the message "java.lang.NullPointerException: Password cannot be null if SASL is enabled" is displayed. The following shows some key logs:

```
2016-05-13 12:05:27.093 | WARN | [task-result-getter-2] | Lost task 98.0 in stage 22.1 (TID 193603, linux-173, 2): FetchFailed(BlockManagerId(13, 172.168.100.13, 27337)),
org.apache.spark.shuffle.FetchFailedException: java.lang.NullPointerException: Password cannot be null if SASL is enabled
 at org.spark-project.guava.base.Preconditions.checkNotNull(Preconditions.java:208)
 at org.apache.spark.network.sasl.SparkSaslServer.encodePassword(SparkSaslServer.java:196)
 at org.apache.spark.network.sasl.SparkSaslServer$DigestCallbackHandler.handle(SparkSaslServer.java:166)
 at com.sun.security.sasl.digest.DigestMD5Server.validateClientResponse(DigestMD5Server.java:589)
 at com.sun.security.sasl.digest.DigestMD5Server.evaluateResponse(DigestMD5Server.java:244)
 at org.apache.spark.network.sasl.SparkSaslServer.response(SparkSaslServer.java:119)
 at org.apache.spark.network.sasl.SaslRpcHandler.receive(SaslRpcHandler.java:100)
 at org.apache.spark.network.server.TransportRequestHandler.processRpcRequest(TransportRequestHandler.java:128)
 at org.apache.spark.network.server.TransportRequestHandler.handle(TransportRequestHandler.java:99)
 at org.apache.spark.network.server.TransportChannelHandler.channelRead0(TransportChannelHandler.java:104)
```

#### Answer

The cause is that NodeManager restarts. When ExternalShuffle is used, Spark uses NodeManager to transmit shuffle data. Therefore, the memory of NodeManager may be seriously insufficient.

In the FusionInsight of the current version, the default memory of NodeManager is only 1 GB. When the data volume of Spark tasks is large (greater than 1 TB),

the memory is severely insufficient and the message response is slow. As a result, the FusionInsight health check determines that the NodeManager process exits and forcibly restarts the NodeManager, causing the preceding problem.

Solution

Adjust the memory of the NodeManager. If the data volume is large (greater than 1 TB), the memory of NodeManager must be greater than 4 GB.

### 24.8.1.10 What Should I Do If the Message "Failed to CREATE\_FILE" Is Displayed in the Restarted Tasks When Data Is Inserted Into the Dynamic Partition Table?

#### Question

When inserting data into the dynamic partition table, a large number of shuffle files are damaged due to the disk disconnection, node error, and the like. In this case, why the message **Failed to CREATE\_FILE** is displayed in the restarted tasks?

```
2016-06-25 15:11:31,323 | ERROR | [Executor task launch worker-0] | Exception in task 15.0 in stage 10.1 (TID 1258) | org.apache.spark.Logging$class.logError(Logging.scala:96)
org.apache.hadoop.hive.ql.metadata.HiveException:
org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.hdfs.protocol.AlreadyBeingCreatedException):
Failed to CREATE_FILE /user/hive/warehouse/testdb.db/web_sales/.hive-staging_hive_2016-06-25_15-09-16_999_8137121701603617850-1/-ext-10000/_temporary/0/_temporary/attempt_201606251509_0010_m_000015_0/ws_sold_date=1999-12-17/part-00015 for
DFSClient_attempt_2016
06251509_0010_m_000015_0_353134803_151 on 10.1.1.5 because this file lease is currently owned by
DFSClient_attempt_201606251509_0010_m_000015_0_-848353830_156 on 10.1.1.6
```

#### Answer

The last step of inserting data into the dynamic partition table is to read shuffle files and then write the data to the mapped partition files.

After a large number of shuffle files are damaged, a large number of tasks fail, causing the restart of jobs. Before the restart of jobs, Spark closes the handles that write table partition files. However, the HDFS cannot process the scenario of batch tasks closing handles. After tasks restart next time, the handles are not released in a timely manner on the NameNode. As a result, the message **Failed to CREATE\_FILE** is displayed.

This error only occurs when a large number of shuffle files are damaged. The tasks will restart after the error occurs and the restart can be completed within milliseconds.

### 24.8.1.11 Why Tasks Fail When Hash Shuffle Is Used?

#### Question

When Hash shuffle is used to run a job that consists of 1000000 map tasks x 100000 reduce tasks, run logs report many message failures and Executor heartbeat timeout, leading to task failures. Why does this happen?

## Answer

During the shuffle process, Hash shuffle just writes the data of different reduce partitions to their respective disk files according to hash results without sorting the data.

If there are many reduce partitions, a large number of disk files will be generated. In your case,  $10^{11}$  shuffle files, that is,  $1000000 * 100000$  shuffle files, will be generated. The sheer number of disk files will have a great impact on the file read and write performance. In addition, the operations such as sorting and compressing will consume a large amount of temporary memory space because a large number of file handles are open, presenting great challenges to memory management and garbage collection and incurring the possibility that the Executor fails to respond to Driver.

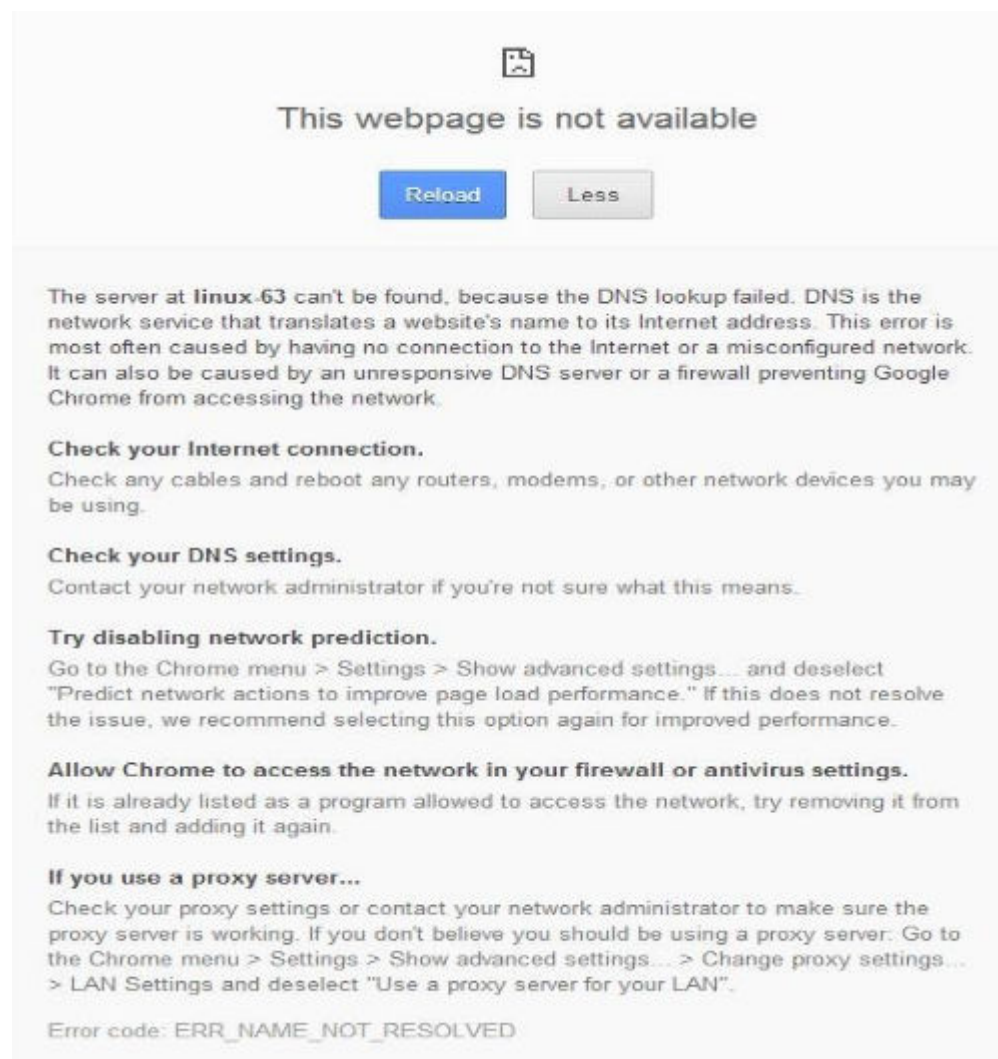
Sort shuffle, instead of Hash shuffle, is recommended to run a job.

### 24.8.1.12 What Can I Do If the Error Message "DNS query failed" Is Displayed When I Access the Aggregated Logs Page of Spark Applications?

#### Question

When the `http(s)://<spark ip>:<spark port>` mode is used to access the Spark JobHistory page, if the displayed Spark JobHistory page is not the page of FusionInsight Manager (the URL of FusionInsight Manager is similar to `https://<oms ip>:20026/Spark2x/JobHistory2x/xx/`), click an application and click **AggregatedLogs**, click the logs of an executor to be viewed. An error message in [Figure 24-9](#) is displayed.

Figure 24-9 DNS query failure



## Answer

**Cause:** The domain name is not added to the **hosts** file of the Windows OS in the pop-up URL (for example, **https://<hostname>:20026/Spark2x/JobHistory2x/xx/history/application\_xxx/jobs/**). As a result, the DNS query fails and the web page cannot be displayed.

### Solution:

- You are advised to visit **Spark JobHistory** page using the FusionInsight Manager.
- If you do not want to access the **Spark JobHistory** page using the FusionInsight Manager, change **<hostname>** in the URL to the IP address or add the domain name to the **hosts** file of the Windows OS.

### 24.8.1.13 What Can I Do If Shuffle Fetch Fails Due to the "Timeout Waiting for Task" Exception?

#### Question

When I execute a 100 TB TPC-DS test suite in the JDBCServer mode, the "Timeout waiting for task" is displayed. As a result, shuffle fetch fails, the stage keeps retrying, and the task cannot be completed properly. What can I do?

#### Answer

The ShuffleService function is used in JDBCServer mode. In the reduce phase, all executors obtain data from NodeManager. When the data volume reaches a level (more than 10 TB), the NodeManager may reach the bottleneck (ShuffleService is in the NodeManager process). As a result, some tasks for obtaining data time out. Therefore, the problem occurs.

You are advised to disable ShuffleService for Spark tasks whose data volume is greater than 10 TB. That is, set `spark.shuffle.service.enabled` in the `Spark-defaults.conf` configuration file to `false`.

### 24.8.1.14 Why Does the Stage Retry due to the Crash of the Executor?

#### Question

When I run Spark tasks with a large data volume, for example, 100 TB TPCDS test suite, why does the Stage retry due to Executor loss sometimes? The message "Executor 532 is lost rpc with driver, but is still alive, going to kill it" is displayed, indicating that the loss of the Executor is caused by a JVM crash.

The log of the key JVM crash is as follows:

```

A fatal error has been detected by the Java Runtime Environment:

Internal Error (sharedRuntime.cpp:834), pid=241075, tid=140476258551552
fatal error: exception happened outside interpreter, nmethods and vtable stubs at pc
0x00007fcda9eb8eb1
```

#### Answer

This error does not affect services. This error is caused by defects of the Oracle JVM, but not the platform code. There is the fault tolerance mechanism for Executors in Spark: the Stage retries in case of an Executor crash to ensure the success execution of tasks.

### 24.8.1.15 Why Do the Executors Fail to Register Shuffle Services During the Shuffle of a Large Amount of Data?

#### Question

When more than 50 terabytes of data is shuffled, some executors fail to register shuffle services due to timeout. The shuffle tasks then fail. Why? The error log is as follows:

```

2016-10-19 01:33:34,030 | WARN | ContainersLauncher #14 | Exception from container-launch with
container ID: container_e1452_1476801295027_2003_01_004512 and exit code: 1 |
LinuxContainerExecutor.java:397
ExitCodeException exitCode=1:
at org.apache.hadoop.util.Shell.runCommand(Shell.java:561)
at org.apache.hadoop.util.Shell.run(Shell.java:472)
at org.apache.hadoop.util.Shell$ShellCommandExecutor.execute(Shell.java:738)
at
org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor.launchContainer(LinuxContainerExecuto
r.java:381)
at
org.apache.hadoop.yarn.server.nodemanager.containermanager.launcher.ContainerLaunch.call(ContainerLau
ch.java:312)
at
org.apache.hadoop.yarn.server.nodemanager.containermanager.launcher.ContainerLaunch.call(ContainerLau
ch.java:88)
at java.util.concurrent.FutureTask.run(FutureTask.java:266)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
at java.lang.Thread.run(Thread.java:745)
2016-10-19 01:33:34,031 | INFO | ContainersLauncher #14 | Exception from container-launch. |
ContainerExecutor.java:300
2016-10-19 01:33:34,031 | INFO | ContainersLauncher #14 | Container id:
container_e1452_1476801295027_2003_01_004512 | ContainerExecutor.java:300
2016-10-19 01:33:34,031 | INFO | ContainersLauncher #14 | Exit code: 1 | ContainerExecutor.java:300
2016-10-19 01:33:34,031 | INFO | ContainersLauncher #14 | Stack trace: ExitCodeException exitCode=1: |
ContainerExecutor.java:300

```

## Answer

The imported data exceeds 50 TB, which exceeds the shuffle processing capability. The shuffle may fail to respond to the registration request of an executor in a timely manner due to the heavy load.

The timeout interval for an executor to register the shuffle service is 5 seconds. The maximum number of retries is 3. This parameter is not configurable.

You are advised to increase the number of task retry times and the number of allowed executor failure times.

Configure the following parameters in the **spark-defaults.conf** file on the client: If **spark.yarn.max.executor.failures** does not exist, manually add it.

**Table 24-79** Parameter Description

Parameter	Description	Default Value
spark.task.maxFailures	Specifies task retry times.	4
spark.yarn.max.executor.failures	Specifies executor failure attempt times. Set <b>spark.dynamicAllocation.enabled</b> to <b>false</b> , to disable the dynamic allocation of executors.	numExecutors * 2, with minimum of 3

Parameter	Description	Default Value
	Specifies executor failure attempt times.  Set <b>spark.dynamicAllocation.enabled to true</b> , to enable the dynamic allocation of executors.	3

### 24.8.1.16 Why Does the Out of Memory Error Occur in NodeManager During the Execution of Spark Applications

#### Question

During the execution of Spark applications, if the YARN External Shuffle service is enabled and there are too many shuffle tasks, the **java.lang.OutOfMemoryError: Direct buffer Memory** error occurs, indicating insufficient memory. The error log is as follows:

```
2016-12-06 02:01:00,768 | WARN | shuffle-server-38 | Exception in connection from /192.168.101.95:53680
| TransportChannelHandler.java:79
io.netty.handler.codec.DecoderException: java.lang.OutOfMemoryError: Direct buffer memory
 at io.netty.handler.codec.ByteToMessageDecoder.channelRead(ByteToMessageDecoder.java:153)
 at
io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChannelHandlerContext.java:
333)
 at
io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelHandlerContext.java:319)
 at io.netty.channel.DefaultChannelPipeline.fireChannelRead(DefaultChannelPipeline.java:787)
 at io.netty.channel.nio.AbstractNioByteChannel$NioByteUnsafe.read(AbstractNioByteChannel.java:130)
 at io.netty.channel.nio.NioEventLoop.processSelectedKey(NioEventLoop.java:511)
 at io.netty.channel.nio.NioEventLoop.processSelectedKeysOptimized(NioEventLoop.java:468)
 at io.netty.channel.nio.NioEventLoop.processSelectedKeys(NioEventLoop.java:382)
 at io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:354)
 at io.netty.util.concurrent.SingleThreadEventExecutor$2.run(SingleThreadEventExecutor.java:116)
 at java.lang.Thread.run(Thread.java:745)
Caused by: java.lang.OutOfMemoryError: Direct buffer memory
 at java.nio.Bits.reserveMemory(Bits.java:693)
 at java.nio.DirectByteBuffer.<init>(DirectByteBuffer.java:123)
 at java.nio.ByteBuffer.allocateDirect(ByteBuffer.java:311)
 at io.netty.buffer.PoolArena$DirectArena.newChunk(PoolArena.java:434)
 at io.netty.buffer.PoolArena.allocateNormal(PoolArena.java:179)
 at io.netty.buffer.PoolArena.allocate(PoolArena.java:168)
 at io.netty.buffer.PoolArena.reallocate(PoolArena.java:277)
 at io.netty.buffer.PooledByteBuf.capacity(PooledByteBuf.java:108)
 at io.netty.buffer.AbstractByteBuf.ensureWritable(AbstractByteBuf.java:251)
 at io.netty.buffer.AbstractByteBuf.writeBytes(AbstractByteBuf.java:849)
 at io.netty.buffer.AbstractByteBuf.writeBytes(AbstractByteBuf.java:841)
 at io.netty.buffer.AbstractByteBuf.writeBytes(AbstractByteBuf.java:831)
 at io.netty.handler.codec.ByteToMessageDecoder.channelRead(ByteToMessageDecoder.java:146)
 ... 10 more
```

#### Answer

In the Shuffle Service of YARN, the number of started threads are twice of the number of available CPU cores. The default size of direct buffer memory is 128

MB. If there are too many shuffle tasks connected at the same time, the direct buffer memory allocated to each thread service is insufficient. For example, if there are 40 CPU cores and there are 80 threads started by the Shuffle Service of YARN, the direct buffer memory allocated to each thread is less than 2 MB.

To solve this problem, increase the directory buffer memory based on the number of CPU cores in NodeManager. For example, if there are 40 of CPU cores, increase the direct buffer memory to 512 MB, that is, configure the **GC\_OPTS** parameter of NodeManager as follows:

`-XX:MaxDirectMemorySize=512M`

 **NOTE**

By default, `-XX:MaxDirectMemorySize` is not configured in the **GC\_OPTS** parameter. To configure it, you need to add it to the **GC\_OPTS** parameter as a custom option.

To configure the **GC\_OPTS** parameter, log in to FusionInsight Manager, choose **Cluster > Name of the desired cluster > Services > Yarn > Configurations**, click **All Configurations**, and choose **NodeManager > System**, and then modify the **GC\_OPTS** parameter.

**Table 24-80** Parameter description

Parameter	Description	Default Value
GC_OPTS	The GC parameter of YARN NodeManger.	128M

### 24.8.1.17 Why Does the Realm Information Fail to Be Obtained When SparkBench is Run on HiBench for the Cluster in Security Mode?

#### Question

Execution of the sparkbench task (for example, Wordcount) of HiBench6 fails. The bench.log indicates that the Yarn task fails to be executed. The failure information displayed on the Yarn UI is as follows:

```
Exception in thread "main" org.apache.spark.SparkException: Unable to load YARN support
 at org.apache.spark.deploy.SparkHadoopUtil$.liftedTree1$1(SparkHadoopUtil.scala:390)
 at org.apache.spark.deploy.SparkHadoopUtil$.yarn$lzycompute(SparkHadoopUtil.scala:385)
 at org.apache.spark.deploy.SparkHadoopUtil$.yarn(SparkHadoopUtil.scala:385)
 at org.apache.spark.deploy.SparkHadoopUtil$.get(SparkHadoopUtil.scala:410)
 at org.apache.spark.deploy.yarn.ApplicationMaster$.main(ApplicationMaster.scala:796)
 at org.apache.spark.deploy.yarn.ExecutorLauncher$.main(ApplicationMaster.scala:821)
 at org.apache.spark.deploy.yarn.ExecutorLauncher.main(ApplicationMaster.scala)
Caused by: java.lang.IllegalArgumentException: Can't get Kerberos realm
 at org.apache.hadoop.security.HadoopKerberosName.setConfiguration(HadoopKerberosName.java:65)
 at org.apache.hadoop.security.UserGroupInformation.initialize(UserGroupInformation.java:288)
 at org.apache.hadoop.security.UserGroupInformation.setConfiguration(UserGroupInformation.java:336)
 at org.apache.spark.deploy.SparkHadoopUtil.<init>(SparkHadoopUtil.scala:51)
 at org.apache.spark.deploy.yarn.YarnSparkHadoopUtil.<init>(YarnSparkHadoopUtil.scala:49)
 at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
 at sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl.java:62)
 at sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAccessorImpl.java:45)
 at java.lang.reflect.Constructor.newInstance(Constructor.java:423)
 at java.lang.Class.newInstance(Class.java:442)
 at org.apache.spark.deploy.SparkHadoopUtil$.liftedTree1$1(SparkHadoopUtil.scala:387)
 ... 6 more
```



```
Caused by: java.lang.reflect.InvocationTargetException
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.apache.hadoop.security.authentication.util.KerberosUtil.getDefaultRealm(KerberosUtil.java:88)
at org.apache.hadoop.security.HadoopKerberosName.setConfiguration(HadoopKerberosName.java:63)
... 16 more
Caused by: KrbException: Cannot locate default realm
at sun.security.krb5.Config.getDefaultRealm(Config.java:1029)
... 22 more
```

## Answer

In C80SPC200 and later, the file stored in the `/etc/krb5.conf` directory is no longer replaced during cluster installation. Instead, the file is stored in the corresponding path on the client through parameter configurations, and HiBench does not reference the client configuration file. Solution: Use the file stored in the `/opt/client/KrbClient/kerberos/var/krb5kdc/krb5.conf` directory on the client to overwrite that in the `/etc/krb5.conf` directories of all nodes. Make a backup before the overwriting.

## 24.8.2 Spark SQL and DataFrame

### 24.8.2.1 What Do I have to Note When Using Spark SQL ROLLUP and CUBE?

#### Question

Suppose that there is a table `src(d1, d2, m)` with the following data:

```
1 a 1
1 b 1
2 b 2
```

The results for statement "select d1, sum(d1) from src group by d1, d2 with rollup" are shown as below:

```
NULL 0
1 2
2 2
1 1
1 1
2 2
```

Why the first line of the above results is (NULL,0), rather than (NULL,4)?

#### Answer

When conducting the rollup and cube operation, we usually perform the dimension-based analysis and what we need is the measurement result, so we would not conduct aggregation operation on the dimension.

Suppose that there is a table `src(d1, d2, m)`, so the statement 1 "select d1, sum(m) from src group by d1, d2 with rollup" conducts the rollup operation on the dimension d1 and d2 to compute the result of m. It has actual business meaning, and its results are in line with the expectation. However, the statement 2 "select d1, sum(d1) from src group by d1, d2 with rollup" cannot be explained from the business perspective. For the statement 2, the result for all aggregations (sum/avg/max/min) is 0.

**NOTE**

Only when there is an aggregation operation for fields in "group by" in the rollup and cube operation, the result is 0. For non-rollup and non-cube operations, the result will be in line with the expectation.

### 24.8.2.2 Why Spark SQL Is Displayed as a Temporary Table in Different Databases?

#### Question

Why temporary tables of the previous database are displayed after the database is switched?

1. Create a temporary DataSource table, for example:  

```
create temporary table ds_parquet
using org.apache.spark.sql.parquet
options(path '/tmp/users.parquet');
```
2. Switch to another database, and run **show tables**. The temporary table created in the previous table is displayed.

```
0: jdbc:hive2://192.168.169.84:22550/default> show tables;
+-----+-----+
| tableName | isTemporary |
+-----+-----+
| ds_parquet | true |
| cmb_tbl_carbon | false |
+-----+-----+
2 rows selected (0.109 seconds)
0: jdbc:hive2://192.168.169.84:22550/default>
```

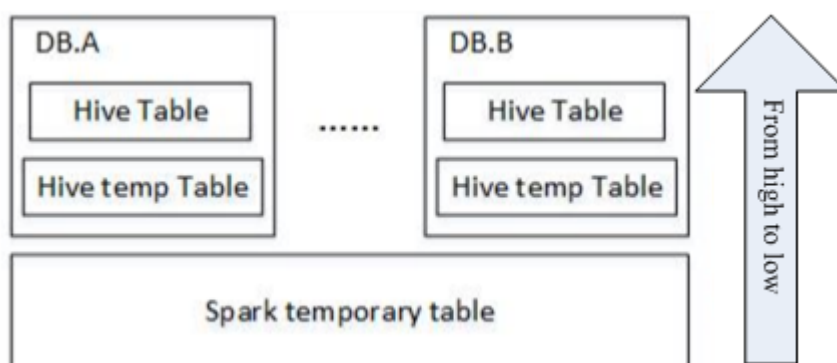
#### Answer

The table management hierarchy of Spark is shown in [Figure 24-10](#). The lowest layer stores all temporary DataSource tables. There is no such concept as database at this layer. DataSource tables are visible in various databases.

The MetaStore of Hive is located at the upper layer. This layer distinguishes among databases. In each database, there are two types of Hive table, permanent and temporary. Therefore, Spark supports data tables of the same name at three layers.

During query, SparksSQL first checks for temporary Spark tables, then temporary Hive tables in the current database, and at last the permanent tables in the current database.

**Figure 24-10** Spark table management hierarchy



When a session quits, temporary tables related to the user operation are automatically deleted. Manual deletion of temporary files is not recommended.

When deleting temporary files, use the same priority as that for query. The priorities are temporary Spark table, temporary Hive table, and permanent Hive table ranging from high to low. If you want to directly delete Hive tables but not temporary Spark tables, you can directly use the ***drop table dbName.TableName*** command.

### 24.8.2.3 How to Assign a Parameter Value in a Spark Command?

#### Question

Is it possible to assign parameter values through Spark commands, in addition to through a user interface or a configuration file?

#### Answer

Spark configuration options can be defined either in a configuration file or in Spark commands.

To assign a parameter value, run the `--conf` command on a Spark client. The parameter value takes effect immediately after the command is run.

The command format is `--conf + parameter name + parameter value`. Example command:

```
--conf spark.eventQueue.size=50000
```

### 24.8.2.4 What Directory Permissions Do I Need to Create a Table Using SparkSQL?

#### Question

The following error information is displayed when a new user creates a table using SparkSQL:

```
0: jdbc:hive2://192.168.169.84:22550/default> create table testACL(c string);
Error: org.apache.spark.sql.execution.QueryExecutionException: FAILED: Execution Error, return code 1 from
org.apache.hadoop.hive.ql.exec.DDLTask. MetaException(message:Got exception:
org.apache.hadoop.security.AccessControlException
Permission denied: user=testACL, access=EXECUTE, inode="/user/hive/warehouse/
testacl":spark:hadoop:drwxrwx---
 at
org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkAccessAcl(FSPermissionChecker.java:
403)
 at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:306)
 at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkTraverse(FSPermissionChecker.java:
259)
 at
org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:
205)
 at
org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:
190)
 at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermission(FSDirectory.java:1710)
 at
org.apache.hadoop.hdfs.server.namenode.FSDirStatAndListingOp.getFileInfo(FSDirStatAndListingOp.java:109)
 at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getFileInfo(FSNamesystem.java:3762)
```

```
at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.getFileInfo(NameNodeRpcServer.java:1014)
at
org.apache.hadoop.hdfs.protocolPB.ClientNamenodeProtocolServerSideTranslatorPB.getFileInfo(ClientNamenodeProtocolServerSideTranslatorPB.java:853)
at org.apache.hadoop.hdfs.protocol.proto.ClientNamenodeProtocolProtos$ClientNamenodeProtocol$2.callBlockingMethod(ClientNamenodeProtocolProtos.java)
at org.apache.hadoop.ipc.ProtobufRpcEngine$Server$ProtoBufRpcInvoker.call(ProtobufRpcEngine.java:616)
at org.apache.hadoop.ipc.RPC$Server.call(RPC.java:973)
at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2089)
at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2085)
at java.security.AccessController.doPrivileged(Native Method)
at javax.security.auth.Subject.doAs(Subject.java:422)
at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1675)
at org.apache.hadoop.ipc.Server$Handler.run(Server.java:2083)
) (state=,code=0)
```

## Answer

When you create a table using Spark SQL, the interface of Hive is called by the underlying system and a directory named after the table will be created in the **/user/hive/warehouse** directory. Therefore, you must have the permissions to read, write, and execute the **/user/hive/warehouse** directory or the group permission of Hive.

The **/user/hive/warehouse** is specified by the `hive.metastore.warehouse.dir` parameter.

### 24.8.2.5 Why Do I Fail to Delete the UDF Using Another Service?

## Question

Why do I fail to delete the UDF using another service, for example, delete the UDF created by Hive using Spark SQL.

## Answer

The UDF can be created using any of the following services:

1. Hive client.
2. JDBCServer API. You can connect JDBCServer to Spark Beeline or JDBC client code, and run SQL statements to create the UDF.
3. spark-sql.

The scenarios in which the UDF failed to be deleted may be as follows:

- If you use Spark Beeline to delete the UDF created by other services, you must restart the JDBCServer before the deletion. Otherwise, the deletion fails. If you use spark-sql to delete the UDF created by other services, you must restart the spark-sql before the deletion. Otherwise, the deletion fails.

Cause: After the UDF is created, if the JDBCServer or the spark-sql has not been restarted, the newly created UDF will not be saved by the FunctionRegistry object in the thread where Spark locates. As a result, the UDF failed to be deleted.

Solution: Restart the JDBCServer and spark-sql of the Spark client and delete the UDF.

- When creating UDF on the Hive client, the **add jar** command (e.g. **add jar /opt/test/two\_udfs.jar**) is used to add the **.jar** package instead of specifying the path of **.jar** package in creating UDF statement. As a result, the **ClassNotFound** error occurs when you use other services to delete the UDF.  
Cause: When you use a service to delete the UDF, the service will load the class that corresponds to the UDF to obtain the UDF. However, the **.jar** package is added by the **add jar** command and jar package does not exist in the classpath of other services. As a result, the **ClassNotFound** error occurs and the UDF failed to be deleted.  
Solution: The UDF created using the preceding approach must be deleted using the same approach. No other approaches are allowed.

### 24.8.2.6 Why Cannot I Query Newly Inserted Data in a Parquet Hive Table Using SparkSQL?

#### Question

Why cannot I query newly inserted data in a parquet Hive table using SparkSQL? This problem occurs in the following scenarios:

1. For partitioned tables and non-partitioned tables, after data is inserted on the Hive client, the latest inserted data cannot be queried using SparkSQL.
2. After data is inserted into a partitioned table using SparkSQL, if the partition information remains unchanged, the newly inserted data cannot be queried using SparkSQL.

#### Answer

To improve Spark performance, parquet metadata is cached. When the parquet table is updated by Hive or another means, the cached metadata remains unchanged, resulting in SparkSQL failing to query the newly inserted data.

For a parquet Hive partition table, if the partition information remains unchanged after data is inserted, the cached metadata is not updated. As a result, the newly inserted data cannot be queried by SparkSQL.

To solve the query problem, update metadata before starting a Spark SQL query.

***REFRESH TABLE table\_name;***

*table\_name* indicates the name of the table to be updated. The table must exist. Otherwise, an error is reported.

When the query statement is executed, the latest inserted data can be obtained.

For details, visit <https://spark.apache.org/docs/3.1.1/sql-programming-guide.html#metadata-refreshing>.

### 24.8.2.7 How to Use Cache Table?

#### Question

What is cache table used for? Which point should I pay attention to while using cache table?

## Answer

Spark SQL caches tables into memory so that data can be directly read from memory instead of disks, reducing memory overhead due to disk reads.

Note that cached tables consume Executor's memory. This means that caching large or many tables compromises Executor's stability even if compressed storage has been used to reduce memory overhead as much as possible.

If it is no longer necessary to accelerate data query by means of cache table, run the following command to uncache tables to free up memory:

```
uncache table table_name
```

### NOTE

The Storage tab page of the Spark Driver user interface displays the cached tables.

## 24.8.2.8 Why Are Some Partitions Empty During Repartition?

### Question

During the repartition operation, the number of blocks (**spark.sql.shuffle.partitions**) is set to 4,500, and the number of keys used by repartition exceeds 4,000. It is expected that data corresponding to different keys can be allocated to different partitions. However, only 2,000 partitions have data, and data corresponding to different keys is allocated to the same partition.

### Answer

This is normal.

The partition to which data is distributed is obtained by performing a modulo operation on hashcode of a key. Different hashcodes may have the same modulo result. In this case, data is distributed to the same partition, as a result, some partitions do not have data, and some partitions have data corresponding to multiple keys.

You can adjust the value of **spark.sql.shuffle.partitions** to adjust the cardinality during modulo operation and improve the unevenness of data blocks. After multiple verifications, it is found that the effect is good when the parameter is set to a prime number or an odd number.

Configure the following parameters in the **spark-defaults.conf** file on the Driver client.

**Table 24-81** Parameter Description

Parameter	Description	Default Value
spark.sql.shuffle.partitions	Number of shuffle data blocks during the shuffle operation.	200

## 24.8.2.9 Why Does 16 Terabytes of Text Data Fails to Be Converted into 4 Terabytes of Parquet Data?

### Question

When the default configuration is used, 16 terabytes of text data fails to be converted into 4 terabytes of parquet data, and the error information below is displayed. Why?

```
Job aborted due to stage failure: Task 2866 in stage 11.0 failed 4 times, most recent failure: Lost task 2866.6 in stage 11.0 (TID 54863, linux-161, 2): java.io.IOException: Failed to connect to /10.16.1.11:23124 at org.apache.spark.network.client.TransportClientFactory.createClient(TransportClientFactory.java:214) at org.apache.spark.network.client.TransportClientFactory.createClient(TransportClientFactory.java:167) at org.apache.spark.network.netty.NettyBlockTransferService$$anon$1.createAndStart(NettyBlockTransferService.scala:92)
```

[Table 24-82](#) lists the default configuration.

**Table 24-82** Parameter Description

Parameter	Description	Default Value
spark.sql.shuffle.partitions	Number of shuffle data blocks during the shuffle operation.	200
spark.shuffle.sasl.timeout	Timeout interval of SASL authentication for the shuffle operation. Unit: second	120s
spark.shuffle.io.connectionTimeout	Timeout interval for connecting to a remote node during the shuffle operation. Unit: second	120s
spark.network.timeout	Timeout interval for all network connection operations. Unit: second	360s

### Answer

The current data volume is 16 TB, but the number of partitions is only 200. As a result, each task is overloaded and the preceding problem occurs.

To solve the preceding problem, you need to adjust the parameters.

- Increase the number of partitions to divide the task into smaller ones.
- Increase the timeout interval during task execution.

Configure the following parameters in the **spark-defaults.conf** file on the client:

**Table 24-83** Parameter Description

Parameter	Description	Recommended Value
spark.sql.shuffle.partitions	Number of shuffle data blocks during the shuffle operation.	4501
spark.shuffle.sasl.timeout	Timeout interval of SASL authentication for the shuffle operation. Unit: second	2000s
spark.shuffle.io.connectionTimeout	Timeout interval for connecting to a remote node during the shuffle operation. Unit: second	3000s
spark.network.timeout	Timeout interval for all network connection operations. Unit: second	360s

### 24.8.2.10 Why the Operation Fails When the Table Name Is TABLE?

#### Question

When the table name is set to **table**, why the error information similar to the following is displayed after the **drop table table** command or other command is run?

```
16/07/12 18:56:29 ERROR SparkSQLDriver: Failed in [drop table table]
java.lang.RuntimeException: [1.1] failure: identifier expected
table
^
at scala.sys.package$.error(package.scala:27)
at org.apache.spark.sql.catalyst.SqlParserTrait$class.parseTableIdentifier(SqlParser.scala:56)
at org.apache.spark.sql.catalyst.SqlParser$.parseTableIdentifier(SqlParser.scala:485)
```

#### Answer

The word table is a keyword of Spark SQL statements and must not be used as a table name.

### 24.8.2.11 Why Is a Task Suspended When the ANALYZE TABLE Statement Is Executed and Resources Are Insufficient?

#### Question

When the **analyze table** statement is executed using spark-sql, the task is suspended and the information below is displayed. Why?

```
spark-sql> analyze table hivetable2 compute statistics;
Query ID = root_20160716174218_90f55869-000a-40b4-a908-533f63866fed
Total jobs = 1
Launching Job 1 out of 1
```



```
Number of reduce tasks is set to 0 since there's no reduce operator
16/07/20 17:40:56 WARN JobResourceUploader: Hadoop command-line option parsing not performed.
Implement the Tool interface and execute your application with ToolRunner to remedy this.
Starting Job = job_1468982600676_0002, Tracking URL = http://10-120-175-107:8088/proxy/
application_1468982600676_0002/
Kill Command = /opt/hadoopclient/HDFS/hadoop/bin/hadoop job -kill job_1468982600676_0002
```

## Answer

When the statement is executed, the SQL statement starts the ***analyze table hivetable2 compute statistics*** MapReduce tasks. On the ResourceManager Web UI of Yarn, the task is not executed due to insufficient resources. As a result, the task is suspended.

Figure 24-11 ResourceManager web UI

Application ID	User	Name	Type	Priority	Created	Finished	State	Reason	Progress	VCores	Nodes
application_1468982600676_0002	root	analyze table hivetable3 compute statistics(Stage=0)	MAPREDUCE	default	Wed Jul 20 17:40:56 +0800 2016	N/A	ACCEPTED	UNDEFINED	0	0	0
application_1468982600676_0001	root	SparkSQL::192.168.169.84	SPARK	default	Wed Jul 20 17:39:21	N/A	RUNNING	UNDEFINED	3	3	4096

Figure 24-12 ResourceManager Web UI

Application ID	User	Name	Type	Priority	Created	Finished	State	Reason	Progress	VCores	Nodes
application_1468982600676_0002	root	analyze table hivetable3 compute statistics(Stage=0)	MAPREDUCE	default	Wed Jul 20 17:40:56 +0800 2016	N/A	ACCEPTED	UNDEFINED	0	0	0
application_1468982600676_0001	root	SparkSQL::192.168.169.84	SPARK	default	Wed Jul 20 17:39:21	N/A	RUNNING	UNDEFINED	3	3	4096

You are advised to add **noscan** when running the ***analyze table*** statement. The function of this statement is the same as that of the ***analyze table hivetable2 compute statistics*** statement. The command is as follows:

```
spark-sql> analyze table hivetable2 compute statistics noscan
```

This command does not start MapReduce tasks and does not occupy Yarn resources. Therefore, the tasks can be executed.

### 24.8.2.12 If I Access a parquet Table on Which I Do not Have Permission, Why a Job Is Run Before "Missing Privileges" Is Displayed?

#### Question

If I access a parquet table on which I do not have permission, why a job is run before "Missing Privileges" is displayed?

#### Answer

The execution sequence of Spark SQL statement parse the table in the statement first, then obtain the metadata in the table, and finally check the permission.

The metadata of a parquet table contains the Split information (which is read by HDFS API) about files. If the table contains many files, the HDFS API reads data in serial mode, in which degrades the performance. If the number of files in the table exceeds the threshold `spark.sql.sources.parallelSplitDiscovery.threshold`, a job will be generated to use Executor to read the data in parallel mode.

The permission authentication is executed after the metadata is obtained. Therefore, when the number of files in the table exceeds the threshold, a job is run before the permission authentication error message **Missing Privileges**.

### 24.8.2.13 Why Do I Fail to Modify MetaData by Running the Hive Command?

#### Question

When do I fail to modify the metadata in the datasource and Spark on HBase table by running the Hive command?

#### Answer

The current Spark version does not support modifying the metadata in the datasource and Spark on HBase tables by running the Hive command.

### 24.8.2.14 Why Is "RejectedExecutionException" Displayed When I Exit Spark SQL?

#### Question

After successfully running Spark tasks with large data volume, for example, 2-TB TPCDS test suite, why is the abnormal stack information **"RejectedExecutionException"** displayed sometimes? The log is as follows:

```
16/07/16 10:19:56 ERROR TransportResponseHandler: Still have 2 requests outstanding when connection from linux-192/10.1.1.5:59250 is closed
java.util.concurrent.RejectedExecutionException: Task scala.concurrent.impl.CallbackRunnable@5fc1ab rejected from java.util.concurrent.ThreadPoolExecutor@52fa7e19[Terminated, pool size = 0, active threads = 0, queued tasks = 0, completed tasks = 3025]
```

#### Answer

When Spark SQL is closed, the application and the message channel are closed. If there are unprocessed messages, the connection should be closed to rectify the exception. If the thread pool inside Scala is closed, the abnormal stack information **"RejectedExecutionException"** is displayed. This abnormal stack information will not be displayed if the thread pool inside Scala is not closed.

The error occurs when the application is successfully run and closed. Therefore, the error will not affect the services.

### 24.8.2.15 What Should I Do If the JDBCServer Process is Mistakenly Killed During a Health Check?

#### Question

During a health check, if the concurrent statements exceed the threshold of the thread pool, the health check statements fail to be executed, the health check program times out, and the Spark JDBCServer process is killed.

#### Answer

There are two thread pools `HiveServer2-Handler-Pool` and `HiveServer2-Background-Pool` in the current JDBCServer. The `HiveServer2-Handler-Pool` is used

to connect sessions and the HiveServer2-Background-Pool is used to run Spark SQL statements.

The current health check mechanism establishes a session connection and runs the health check command **HEALTHCHECK** in the thread of the session to check the health condition of the Spark JDBCServer. Therefore, one thread must be reserved for the HiveServer2-Handler-Pool respectively to connect sessions and run statements for the health check. Otherwise, the session connection and statement running will fail and the Spark JDBCServer will be killed because it is mistakenly considered unhealthy. For example, if there are 100 threads in the HiveServer2-Handler-Pool respectively, a maximum of 99 sessions can be connected.

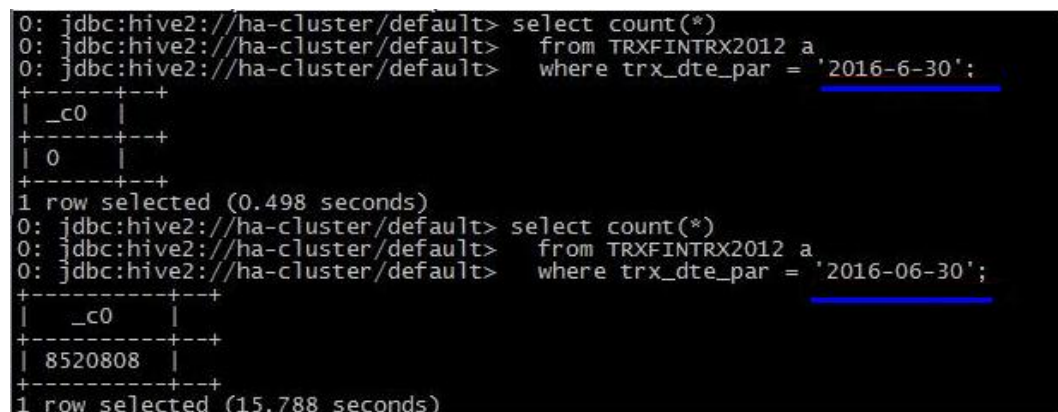
### 24.8.2.16 Why No Result Is found When 2016-6-30 Is Set in the Date Field as the Filter Condition?

#### Question

Why no result is found when 2016-6-30 is set in the date field as the filter condition?

As shown in the following figure, `trx_dte_par` in the `select count (*) from trxfintrx2012 a where trx_dte_par='2016-6-30'` statement is a date field. However, no search result is found when the filter condition is where `trx_dte_par='2016-6-30'`. Search results are found only when the filter condition is where `trx_dte_par='2016-06-30'`.

Figure 24-13 Example



```
0: jdbc:hive2://ha-cluster/default> select count(*)
0: jdbc:hive2://ha-cluster/default> from TRXFINTRX2012 a
0: jdbc:hive2://ha-cluster/default> where trx_dte_par = '2016-6-30';
+-----+
| _c0 |
+-----+
| 0 |
+-----+
1 row selected (0.498 seconds)
0: jdbc:hive2://ha-cluster/default> select count(*)
0: jdbc:hive2://ha-cluster/default> from TRXFINTRX2012 a
0: jdbc:hive2://ha-cluster/default> where trx_dte_par = '2016-06-30';
+-----+
| _c0 |
+-----+
| 8520808 |
+-----+
1 row selected (15.788 seconds)
```

#### Answer

If a data string of the date type is present in Spark SQL statements, the Spark SQL will search the matching character string without checking the date format. In this case, if the date format in the SQL statement is incorrect, the query will fail. For example, if the data format is `yyyy-mm-dd`, then no search results matching `'2016-6-30'` will be found.

### 24.8.2.17 Why Does the "--hivevar" Option I Specified in the Command for Starting spark-beeline Fail to Take Effect?

#### Question

Why does the `--hivevar` option I specified in the command for starting spark-beeline fail to take effect?

In the V100R002C60 version, if I use the `--hivevar <VAR_NAME>=<var_value>` option to define a variable in the command for starting spark-beeline, no error is reported in spark-beeline. However, if the variable `<VAR_NAME>` is used in SQL, the variable cannot be parsed and the `<VAR_NAME>` exception is reported.

For example:

1. Run the following command to start the spark-beeline:  
`spark-beeline --hivevar <VAR_NAME>=<var_value>`
2. After spark-beeline is started successfully, I run the SQL statements `DROP TABLE ${VAR_NAME}` in spark-beeline. The `VAR_NAME` exception occurs.

#### Answer

In the V100R002C60 version, the `--hivevar <VAR_NAME>=<var_value>` feature of Hive is not supported in Spark because multi-session management function is added. Therefore, the `--hivevar` option in the command for starting spark-beeline is invalid.

### 24.8.2.18 Why Does the "Permission denied" Exception Occur When I Create a Temporary Table or View in Spark-beeline?

#### Question

In normal mode, when I create a temporary table or view in spark-beeline, the error message "Permission denied" is displayed, indicating that I have no permissions on the HDFS directory. The error log information is as follows:

```
org.apache.hadoop.security.AccessControlException Permission denied: user=root, access=EXECUTE,
inode="/tmp/spark/sparkhive-scratch/omm/e579a76f-43ed-4014-8a54-1072c07ceeff/_tmp_space.db/
52db1561-60b0-4e7d-8a25-c2eaa44850a9":omm:hadoop:drwx-----
```

#### Answer

In normal mode, if you run the spark-beeline command as a non-omm user, **root** user for example, without specifying the `-n` parameter, your account is still the root user. After spark-beeline is started, a new HDFS directory is created by JDBCServer. In the current version of DataSight, the user that starts the JDBCServer is **omm**. In versions earlier than DataSight V100R002C30, the user is **root**. Therefore, the owner of the HDFS directory is **omm** and the group is **hadoop**. The HDFS directory is used when you create a temporary table or view in spark-beeline and the user **root** is a common user in HDFS and has no permissions on the directory of user **omm**. As a result, the "Permission denied" exception occurs.

In normal mode, only user **omm** can create a temporary table or view. To solve this problem, you can specify the `-n omm` option for user **omm** when starting

spark-beeline. In this way, you have the permissions to perform operations on the HDFS directory.

### 24.8.2.19 Why Is the "Code of method ... grows beyond 64 KB" Error Message Displayed When I Run Complex SQL Statements?

#### Question

When I run a complex SQL statement, for example, SQL statements with multiple layers of nesting statements and a single layer statement contains a large number of logic clauses such as case when, an error message indicating that the code of a certain method exceeds 64 KB is displayed. The log is as follows:

```
java.util.concurrent.ExecutionException: java.lang.Exception: failed to compile:
org.codehaus.janino.JaninoRuntimeException: Code of method "(Lorg/apache/spark/sql/catalyst/expressions/
GeneratedClass$SpecificUnsafeProjection;Lorg/apache/spark/sql/catalyst/InternalRow;)V" of class
"org.apache.spark.sql.catalyst.expressions.GeneratedClass$SpecificUnsafeProjection" grows beyond 64 KB
```

#### Answer

If Project Tungsten is enabled, Spark will use codegen method to generate Java code for part of execution plan. However, each function in Java code to be compiled by JDK must be less than 64 KB. If complex SQL statements are run, the function in the Java code generated by codegen may exceed 64 KB, causing compilation failure.

To solve the problem, go to the **spark-defaults.conf** file on the client and set the **spark.sql.codegen.wholeStage** parameter to **false** to disable Project Tungsten.

### 24.8.2.20 Why Is Memory Insufficient if 10 Terabytes of TPCDS Test Suites Are Consecutively Run in Beeline/JDBCServer Mode?

#### Question

When the driver memory is set to 10 GB and the 10 TB TPCDS test suites are continuously run in Beeline/JDBCServer mode, SQL statements fail to be executed due to insufficient driver memory. Why?

#### Answer

By default, 1000 UI data records of jobs and stages are reserved in the memory.

The function of overflowing UI data to disks has been added to optimize large clusters. The overflow condition is that the size of UI data in each stage reaches the minimum threshold 5 MB. If the number of tasks in each stage is small, the size of UI data in the stage may not reach the threshold. As a result, the UI data in the stage is cached in the memory until the number of UI data records reaches the upper limit (1000 by default). Only then the old UI data is cleared from the memory.

Therefore, before the old UI data is cleared, the UI data occupies a large amount of memory. As a result, the driver memory is insufficient when 10 terabytes of TPCDS test suites are executed.

Workaround:

- Set `spark.ui.retainedJobs` and `spark.ui.retainedStages` based on service requirements to specify the number of UI data records of jobs and stages to be reserved. For details, see [Table 24-15](#) in [Common Parameters](#).
- If a large amount of UI data of jobs and stages needs to be reserved, increase the memory of the driver by setting the `spark.driver.memory` parameter. For details, see [Table 24-12](#) in [Common Parameters](#).

## 24.8.2.21 Why Are Some Functions Not Available when Another JDBCServer Is Connected?

### Question

Scenario 1

I set up permanent functions using the `add jar` statement. After Beeline connects to different JDBCServer or JDBCServer is restarted, I have to run the `add jar` statement again.

Figure 24-14 Error information in scenario 1

```
0: jdbc:hive2://192.168.91.247:23040/default> create function al as '
-----+-----+
| result |
-----+-----+
NO rows selected (0.222 seconds)
0: jdbc:hive2://192.168.91.247:23040/default> SELECT test.al(array(1, 2, 3), array(2));
-----+-----+
| _c0 |
-----+-----+
| true |
-----+-----+
1 row selected (8.282 seconds)
0: jdbc:hive2://192.168.91.247:23040/default> closing: 0: jdbc:hive2://192.168.91.247:24002,192.168.154.81:24002,192.168.8.27:24002;serviceDiscoveryMode=zooKeeper
-beeline-cort;auth=KERBEROS;principal=spark/hadoop,hadoop.com@HADOOP.COM;
100-106-121-140:/opt/hadoopclient # ./spark-beeline
It's running the fl spark-beeline, it calls /opt/hadoopclient/spark/spark/bin/beeline
and helps to connect to the JDBCServer automatically
connecting to jdbc:hive2://192.168.91.247:24002,192.168.154.81:24002,192.168.8.27:24002;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=sparkthriftserver;sa
dooop,hadoop.com@HADOOP.COM;
2017-06-15 08:17:55,495 | WARN | Thread-2 | TGT refresh thread time adjusted from : Thu Jun 15 05:59:42 GMT+08:00 2017 to : Thu Jun 15 08:18:55 GMT+08:00 2017
fresh interval (60 seconds) from now. | org.apache.zookeeper.Login$.run(Login.java:177)
2017-06-15 08:17:56,743 | WARN | main | unable to load native-hadoop library for your platform... using builtin-java classes where applicable | org.apache.had
ader.java:62)
2017-06-15 08:17:56,773 | WARN | TGT Renewer for sparkuser@HADOOP.COM | Exception encountered while running the renewal command. Aborting renew thread. ExitCo
d requested option while renewing credentials
| org.apache.hadoop.security.UserGroupInformation$.run(UserGroupInformation.java:946)
Connected to: Spark SQL (version)
Driver: Hive JDBC (version 1.2.1,spark)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 1.2.1,spark by Apache Hive
[INFO] unable to bind key for unsupported operation: backward-delete-word
[INFO] unable to bind key for unsupported operation: backward-delete-word
[INFO] unable to bind key for unsupported operation: down-history
[INFO] unable to bind key for unsupported operation: up-history
[INFO] unable to bind key for unsupported operation: up-history
[INFO] unable to bind key for unsupported operation: down-history
[INFO] unable to bind key for unsupported operation: up-history
[INFO] unable to bind key for unsupported operation: up-history
[INFO] unable to bind key for unsupported operation: down-history
[INFO] unable to bind key for unsupported operation: up-history
[INFO] unable to bind key for unsupported operation: down-history
0: jdbc:hive2://192.168.8.27:23040/default> SELECT test.al(array(1, 2, 3), array(2));
Error: org.apache.spark.SparkException: unable to load UDF class (state=,code=0)
0: jdbc:hive2://192.168.8.27:23040/default> set role admin;
-----+-----+
| key | value |
-----+-----+
| role admin |
-----+-----+
1 row selected (0.465 seconds)
0: jdbc:hive2://192.168.8.27:23040/default> add jar /home/smartcare-udf-0.0.1-SNAPSHOT.jar;
-----+-----+
| result |
-----+-----+
| 0 |
-----+-----+
```

Scenario 2

The `show functions` statement can be used to query functions, but not obtain functions. The reason is that connected JDBC node does not contain jar packages of the corresponding path. However, after I add corresponding `.jar` packages, the `show functions` statement can be used to obtain functions.

Figure 24-15 Error information in scenario 2

```

-----+-----+
| function |
-----+-----+
stddev_pop
stddev_samp
str_to_map
string
struct
substr
substr_index
sum
tan
tanh
test.a1
timestamp
tinyint
to_date
to_unix_timestamp
to_utc_timestamp
translate
trim
trunc
ucase
unbase64
unhex
unix_timestamp
upper
var_pop
var_samp
variance
weekofyear
when
window
xpath
0: jdbc:hive2://192.168.8.27:22550/default> use test;
-----+-----+
| Result |
-----+-----+
No rows selected (0.038 seconds)
0: jdbc:hive2://192.168.8.27:22550/default> SELECT test.a1(array(1, 2, 3), array(2));
Error: org.apache.spark.sql.AnalysisException: undefined function: 'test.a1'. This function is neither a registered temporary function nor a permaner
7 (state=,code=0)
0: jdbc:hive2://192.168.8.27:22550/default> show functions;
-----+-----+
| function |
-----+-----+

```

## Answer

### Scenario 1

The **add jar** statement is used to load jars to the jarClassLoader of the JDBCServer connected currently. The **add jar** statement is not shared by different JDBCServer. After the JDBCServer restarts, new jarClassLoader is created. So the add jar statement needs to be run again.

There are two methods to add jar packages: You can run the **spark-sql --jars /opt/test/two\_udfs.jar** statement to add the jar package during the startup of the Spark SQL process; or run the **add jar /opt/test/two\_udfs.jar** statement to add the jar package after the Spark SQL process is started. Note that the path following the add jar statement can be a local path or an HDFS path.

### Scenario 2

The show functions statement is used to obtain all functions in the current database from the external catalog. If functions are used in SQL, thriftJDBC-server loads .jar files related to the function.

If .jar files do not exist, the function cannot obtain corresponding .jar files. Therefore, the corresponding .jar files need to be added.

## 24.8.2.22 Why Does Spark2x Have No Access to DataSource Tables Created by Spark1.5?

### Question

When Spark2x accesses the DataSource table created by Spark1.5, a message is displayed indicating that schema information cannot be obtained. As a result, the table cannot be accessed. Why?

## Answer

- Cause analysis:  
This is because the formats of the DataSource table information stored in Spark2x and Spark1.5 are inconsistent. Spark 1.5 divides schema information into multiple parts and uses **path.park.0** as the key for storage. Spark 1.5 reads information from each part and reassembles the information into complete one. Spark2x directly uses the corresponding key to obtain the corresponding information. In this case, when Spark2x reads the DataSource table created by Spark1.5, the information corresponding to the key cannot be read. As a result, the DataSource table information fails to be parsed.  
When processing Hive tables, Spark2x and Spark1.5 use the same storage mode. Therefore, Spark2x can directly read tables created by Spark1.5.
- Workaround:  
In Spark2x, create a foreign table to point to the actual data in the Spark1.5 table. In this way, the DataSource table created by Spark1.5 can be read in Spark2x. In addition, after Spark1.5 updates data, Spark2x can detect the change. The reverse is also true. In this way, Spark2x can access the DataSource table created by Spark1.5.

### 24.8.2.23 Why Does Spark-beeline Fail to Run and Error Message "Failed to create ThriftService instance" Is Displayed?

#### Question

Why does "Failed to create ThriftService instance" occur when spark beeline fails to run?

Beeline logs are as follows:

```
Error: Failed to create ThriftService instance (state=,code=0)
Beeline version 1.2.1.spark by Apache Hive
[INFO] Unable to bind key for unsupported operation: backward-delete-word
[INFO] Unable to bind key for unsupported operation: backward-delete-word
[INFO] Unable to bind key for unsupported operation: down-history
[INFO] Unable to bind key for unsupported operation: up-history
[INFO] Unable to bind key for unsupported operation: up-history
[INFO] Unable to bind key for unsupported operation: down-history
[INFO] Unable to bind key for unsupported operation: up-history
[INFO] Unable to bind key for unsupported operation: down-history
[INFO] Unable to bind key for unsupported operation: up-history
[INFO] Unable to bind key for unsupported operation: down-history
[INFO] Unable to bind key for unsupported operation: up-history
[INFO] Unable to bind key for unsupported operation: down-history
beeline>
```

In addition, the "Timed out waiting for client to connect" error log is generated on the JDBCServer. The details are as follows:

```
2017-07-12 17:35:11,284 | INFO | [main] | Will try to open client transport with JDBC Uri:
jdbc:hive2://192.168.101.97:23040/default;principal=spark/hadoop.<System domain name>@<System
domain name>;healthcheck=true;saslQop=auth-conf;auth=KERBEROS;user.principal=spark/hadoop.<System
domain name>@<System domain name>;user.keytab=${BIGDATA_HOME}/FusionInsight_HD_8.0.2.1/install/
FusionInsight-Spark-2.4.5/keytab/spark/JDBCServer/spark.keytab |
org.apache.hive.jdbc.HiveConnection.openTransport(HiveConnection.java:317)
2017-07-12 17:35:11,326 | INFO | [HiveServer2-Handler-Pool: Thread-92] | Client protocol version:
HIVE_CLI_SERVICE_PROTOCOL_V8 |
org.apache.proxy.service.ThriftCLIProxyService.OpenSession(ThriftCLIProxyService.java:554)
2017-07-12 17:35:49,790 | ERROR | [HiveServer2-Handler-Pool: Thread-113] | Timed out waiting for client
```



**to connect.**

Possible reasons include network issues, errors in remote driver or the cluster has no available resources, etc. Please check YARN or Spark driver's logs for further information. |

```
org.apache.proxy.service.client.SparkClientImpl.<init>(SparkClientImpl.java:90)
java.util.concurrent.ExecutionException: java.util.concurrent.TimeoutException: Timed out waiting for client connection.
```

```
at io.netty.util.concurrent.AbstractFuture.get(AbstractFuture.java:37)
at org.apache.proxy.service.client.SparkClientImpl.<init>(SparkClientImpl.java:87)
at org.apache.proxy.service.client.SparkClientFactory.createClient(SparkClientFactory.java:79)
at org.apache.proxy.service.SparkClientManager.createSparkClient(SparkClientManager.java:145)
at org.apache.proxy.service.SparkClientManager.createThriftServerInstance(SparkClientManager.java:160)
at org.apache.proxy.service.ThriftServiceManager.getOrCreateThriftServer(ThriftServiceManager.java:182)
at org.apache.proxy.service.ThriftCLIProxyService.OpenSession(ThriftCLIProxyService.java:596)
at org.apache.hive.service.cli.thrift.TCLIService$Processor$OpenSession.getResult(TCLIService.java:1257)
at org.apache.hive.service.cli.thrift.TCLIService$Processor$OpenSession.getResult(TCLIService.java:1242)
at org.apache.thrift.ProcessFunction.process(ProcessFunction.java:39)
at org.apache.thrift.TBaseProcessor.process(TBaseProcessor.java:39)
at org.apache.hadoop.hive.thrift.HadoopThriftAuthBridge$Server
$TUGIAssumingProcessor.process(HadoopThriftAuthBridge.java:696)
at org.apache.thrift.server.TThreadPoolServer$WorkerProcess.run(TThreadPoolServer.java:286)
at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:617)
at java.lang.Thread.run(Thread.java:748)
```

Caused by: java.util.concurrent.TimeoutException: Timed out waiting for client connection.

**Answer**

This problem occurs when the network is unstable. When a timed-out exception occurs in beeline, Spark does not attempt to reconnect to beeline. Therefore, you need to restart spark-beeline for reconnection.

## 24.8.3 Spark Streaming

### 24.8.3.1 What Can I Do If Spark Streaming Tasks Are Blocked?

**Question**

After a Spark Streaming task is run and data is input, no processing result is displayed. Open the web page to view the Spark job execution status. The following figure shows that two jobs are waiting to be executed but cannot be executed successfully.

**Figure 24-16 Active Jobs**

Active Jobs (2)

Job Id	Description ▾	Submitted	Duration	Stages: Succeeded/Total
3	<a href="#">print at test2StreamFromKafka.scala:31</a>	2015/05/25 18:28:55	63.7 h	0/3
2	<a href="#">start at test2StreamFromKafka.scala:34</a>	2015/05/25 18:28:55	63.7 h	0/1

Check the completed jobs. Only two jobs are found, indicating that Spark Streaming does not trigger data computing tasks. (By default, Spark Streaming has two jobs that attempt to run. See the figure below.)

**Figure 24-17 Completed Jobs**

Completed Jobs (2)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total
1	<a href="#">print at test2StreamFromKafka.scala:31</a>	2015/05/25 18:28:55	0.7 s	2/2 (1 skipped)
0	<a href="#">start at test2StreamFromKafka.scala:34</a>	2015/05/25 18:28:54	1 s	2/2

## Answer

After fault locating, it is found that the number of computing cores of Spark Streaming is less than the number of receivers. As a result, after some receivers are started, no resource is available to run computing tasks. Therefore, the first task keeps waiting and subsequent tasks keep queuing. [Figure 24-16](#) is an example of two queuing tasks.

To address this problem, it is advised to check whether the number of Spark cores is greater than the number of receivers when two tasks are queuing.

### NOTE

Receiver is a permanent Spark job in Spark Streaming. It is common for Spark, but its life cycle is the same as that of a Spark Streaming task and occupies one computing core.

Pay attention to the relationship between the number of cores and the number of receivers in scenarios where default configurations are often used, such as debugging and testing.

## 24.8.3.2 What Should I Pay Attention to When Optimizing Spark Streaming Task Parameters?

### Question

When Spark Streaming tasks are running, the data processing performance does not improve significantly as the number of executors increases. What should I pay attention to if I perform parameter optimization?

### Answer

When the number of executor cores is 1, comply with the following rules to optimize Spark Streaming running parameters:

- The Spark task processing speed is related to the number of partitions in Kafka. When the number of partitions is less than the specified number of executors, the number of actually used executors is the same as the number of partitions, and other executors will be idle. Therefore, the number of executors must be less than or equal to the number of partitions.
- When data skew occurs on different partitions of Kafka, the executor corresponding to the partition with a large amount of data touches the glass ceiling of data processing. Therefore, when the Producer program is executed, data is sent to each partition on average to improve the processing speed.
- When partition data is evenly distributed, increasing the number of partitions and executors will improve the Spark processing speed. (When the number of partitions is the same as that of executors, the processing speed is the fastest.)

- When partition data is evenly distributed, ensure that the number of partitions is an integer multiple of the number of executors for proper allocation of resources.

### 24.8.3.3 Why Does the Spark Streaming Application Fail to Be Submitted After the Token Validity Period Expires?

#### Question

Change the validity period of the Kerberos ticket and HDFS token to 5 minutes, set **dfs.namenode.delegation.token.renew-interval** to a value less than 60 seconds, and submit the Spark Streaming application. If the token expires, the error message below is displayed, and the application exits. Why?

```
token (HDFS_DELEGATION_TOKEN token 17410 for spark2x) is expired
```

#### Answer

- Possible causes:

The credential refresh thread of the ApplicationMaster process uploads the updated credential file to the HDFS based on the *token renew period multiplied by 0.75*.

In the executor process, the credential refresh thread obtains the updated credential file from the HDFS based on the time ratio of the *token renewal period multiplied by 0.8* to update the token in UserGroupInformation, preventing the token from being invalid.

When the credential refresh thread of the executor process detects that the current time is later than the credential file update time (*token renew period  $\times$  0.8*), it waits for 1 minute and then obtains the latest credential file from the HDFS to ensure that the AM has stored the updated credential file in the HDFS.

When the value of **dfs.namenode.delegation.token.renew-interval** is less than 60 seconds, the started executor detects that the current time is later than the time when the credential file is updated. One minute later, the executor obtains the latest credential file from the HDFS. However, the token is already invalid, and the task fails to be executed. Then, other executor processes retry within 1 minute. The task also fails to run on other executors. As a result, the executors that fail to run are added to the blacklist. If no executors are available, the application exits.

- Solution:

In the Spark application scenario, set **dfs.namenode.delegation.token.renew-interval** to a value greater than 80 seconds. For details about the **dfs.namenode.delegation.token.renew-interval** parameter, see [Table 24-84](#).

**Table 24-84** Parameter description

Parameter	Description	Default Value
dfs.namenode.delegation.token.renew-interval	This parameter is a server parameter. It specifies the maximum lifetime to renew a token. Unit: milliseconds.	86400000

### 24.8.3.4 Why does Spark Streaming Application Fail to Restart from Checkpoint When It Creates an Input Stream Without Output Logic?

#### Question

Spark Streaming application creates one input stream without output logic. The application fails to restart from checkpoint and an error will be shown like below:

```
17/04/24 10:13:57 ERROR Utils: Exception encountered
java.lang.NullPointerException
at org.apache.spark.streaming.dstream.DStreamCheckpointData$$anonfun$writeObject$1.applymcVsp(DStreamCheckpointData.scala:125)
at org.apache.spark.streaming.dstream.DStreamCheckpointData$$anonfun$writeObject$1.apply(DStreamCheckpointData.scala:123)
at org.apache.spark.streaming.dstream.DStreamCheckpointData$$anonfun$writeObject$1.apply(DStreamCheckpointData.scala:123)
at org.apache.spark.util.Utils$.tryOrIOException(Utils.scala:1195)
at org.apache.spark.streaming.dstream.DStreamCheckpointData.writeObject(DStreamCheckpointData.scala:123)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at java.io.ObjectStreamClass.invokeWriteObject(ObjectStreamClass.java:1028)
at java.io.ObjectOutputStream.writeSerialData(ObjectOutputStream.java:1496)
at java.io.ObjectOutputStream.writeOrdinaryObject(ObjectOutputStream.java:1432)
at java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1178)
at java.io.ObjectOutputStream.defaultWriteFields(ObjectOutputStream.java:1548)
at java.io.ObjectOutputStream.defaultWriteObject(ObjectOutputStream.java:441)
at org.apache.spark.streaming.dstream.DStream$$anonfun$writeObject$1.applymcVsp(DStream.scala:515)
at org.apache.spark.streaming.dstream.DStream$$anonfun$writeObject$1.apply(DStream.scala:510)
at org.apache.spark.streaming.dstream.DStream$$anonfun$writeObject$1.apply(DStream.scala:510)
at org.apache.spark.util.Utils$.tryOrIOException(Utils.scala:1195)
at org.apache.spark.streaming.dstream.DStream.writeObject(DStream.scala:510)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at java.io.ObjectStreamClass.invokeWriteObject(ObjectStreamClass.java:1028)
at java.io.ObjectOutputStream.writeSerialData(ObjectOutputStream.java:1496)
at java.io.ObjectOutputStream.writeOrdinaryObject(ObjectOutputStream.java:1432)
at java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1178)
at java.io.ObjectOutputStream.writeArray(ObjectOutputStream.java:1378)
at java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1174)
at java.io.ObjectOutputStream.defaultWriteFields(ObjectOutputStream.java:1548)
at java.io.ObjectOutputStream.writeSerialData(ObjectOutputStream.java:1509)
at java.io.ObjectOutputStream.writeOrdinaryObject(ObjectOutputStream.java:1432)
at java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1178)
at java.io.ObjectOutputStream.defaultWriteFields(ObjectOutputStream.java:1548)
at java.io.ObjectOutputStream.defaultWriteObject(ObjectOutputStream.java:441)
at org.apache.spark.streaming.DStreamGraph$$anonfun$writeObject$1.applymcVsp(DStreamGraph.scala:
```

```

191)
at org.apache.spark.streaming.DStreamGraph$$anonfun$writeObject$1.apply(DStreamGraph.scala:186)
at org.apache.spark.streaming.DStreamGraph$$anonfun$writeObject$1.apply(DStreamGraph.scala:186)
at org.apache.spark.util.Utils$.tryOrIOException(Utils.scala:1195)
at org.apache.spark.streaming.DStreamGraph.writeObject(DStreamGraph.scala:186)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at java.io.ObjectStreamClass.invokeWriteObject(ObjectStreamClass.java:1028)
at java.io.ObjectOutputStream.writeSerialData(ObjectOutputStream.java:1496)
at java.io.ObjectOutputStream.writeOrdinaryObject(ObjectOutputStream.java:1432)
at java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1178)
at java.io.ObjectOutputStream.defaultWriteFields(ObjectOutputStream.java:1548)
at java.io.ObjectOutputStream.writeSerialData(ObjectOutputStream.java:1509)
at java.io.ObjectOutputStream.writeOrdinaryObject(ObjectOutputStream.java:1432)
at java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1178)
at java.io.ObjectOutputStream.writeObject(ObjectOutputStream.java:348)
at org.apache.spark.streaming.Checkpoint$$anonfun$serialize$1.applymcVsp(Checkpoint.scala:142)
at org.apache.spark.streaming.Checkpoint$$anonfun$serialize$1.apply(Checkpoint.scala:142)
at org.apache.spark.streaming.Checkpoint$$anonfun$serialize$1.apply(Checkpoint.scala:142)
at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1230)
at org.apache.spark.streaming.Checkpoint$.serialize(Checkpoint.scala:143)
at org.apache.spark.streaming.StreamingContext.validate(StreamingContext.scala:566)
at org.apache.spark.streaming.StreamingContext.liftedTree1$1(StreamingContext.scala:612)
at org.apache.spark.streaming.StreamingContext.start(StreamingContext.scala:611)
at com.spark.test.kafka08LifoTwoInkfk$.main(kafka08LifoTwoInkfk.scala:21)
at com.spark.test.kafka08LifoTwoInkfk.main(kafka08LifoTwoInkfk.scala)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.apache.spark.deploy.SparkSubmit$.org$apache$spark$deploy$SparkSubmit$
$runMain(SparkSubmit.scala:772)
at org.apache.spark.deploy.SparkSubmit$.doRunMain$1(SparkSubmit.scala:183)
at org.apache.spark.deploy.SparkSubmit$.submit(SparkSubmit.scala:208)
at org.apache.spark.deploy.SparkSubmit$.main(SparkSubmit.scala:123)
at org.apache.spark.deploy.SparkSubmit.main(SparkSubmit.scala)

```

## Answer

When Streaming Context starts, DStream checkpoint object of application should be serialized with application set to checkpoint and Dstream context will be used during this serialization.

Dstream.context is the Dstream which Streaming Context relies on to check reversely from output Stream, set the context one by one. If Spark Streaming application creates one input stream which does not have output logic, there will be no context set for the input stream. 'NullPointerException' will be reported during serialization.

Solution: If there is no input logic for the output stream in the application, delete the input stream in the code or add the relevant output logic for that input stream.

### 24.8.3.5 Why Is the Input Size Corresponding to Batch Time on the Web UI Set to 0 Records When Kafka Is Restarted During Spark Streaming Running?

#### Question

When the Kafka is restarted during the execution of the Spark Streaming application, the application cannot obtain the topic offset from the Kafka. As a

result, the job fails to be generated. As shown in **Figure 24-18**, **2017/05/11 10:57:00-2017/05/11 10:58:00** indicates the Kafka restart time. After the restart is successful at 10:58:00 on May,11,2017, the value of **Input Size** is **0 records**.

**Figure 24-18** On the Web UI, the **input size** corresponding to the **batch time** is **0 records**.

Completed Batches (last 9 out of 9)

Batch Time	Input Size	Scheduling Delay (?)	Processing Time (?)	Total Delay (?)	Output Ops: Succeeded/Total
2017/05/11 10:58:50	18 records	0 ms	0.4 s	0.4 s	1/1
2017/05/11 10:58:40	20 records	4 s	0.3 s	4 s	1/1
2017/05/11 10:58:30	20 records	14 s	0.5 s	14 s	1/1
2017/05/11 10:58:20	20 records	23 s	0.4 s	24 s	1/1
2017/05/11 10:58:10	20 records	33 s	0.5 s	33 s	1/1
2017/05/11 10:58:00	0 records	6 ms	43 s	43 s	1/1
2017/05/11 10:57:00	19 records	1 ms	0.9 s	0.9 s	1/1
2017/05/11 10:56:50	20 records	1 ms	0.6 s	0.6 s	1/1
2017/05/11 10:56:40	28 records	13 ms	5 s	5 s	1/1

## Answer

After Kafka is restarted, the application supplements the missing RDD between 10:57:00 on May 11, 2017 and 10:58:00 on May 11, 2017 based on the batch time. Although the number of read data records displayed on the UI is **0**, the missing data is processed in the supplemented RDD. Therefore, no data loss occurs.

The data processing mechanism during the Kafka restart period is as follows:

The Spark Streaming application uses the **state** function (for example, **updateStateByKey**). After Kafka is restarted, the Spark Streaming application generates a batch task at 10:58:00 on May 11, 2017. The missing RDD between 10:57:00 on May 11, 2017 and 10:58:00 on May 11, 2017 is supplemented based on the batch time (data that is not read in Kafka before Kafka restart, which belongs to the batch before 10:57:00 on May 11, 2017).

## 24.8.4 Why the Job Information Obtained from the restful Interface of an Ended Spark Application Is Incorrect?

### Question

The job information obtained from the restful interface of an ended Spark application is incorrect: the value of **numActiveTasks** is negative, as shown in **Figure 24-19**:

Figure 24-19 job information

```
[{
 "jobId" : 0,
 "name" : "reduce at SparkPi.scala:36",
 "submissionTime" : "2016-05-28T09:35:34.415GMT",
 "completionTime" : "2016-05-28T09:35:35.686GMT",
 "stageIds" : [0],
 "status" : "SUCCEEDED",
 "numTasks" : 2,
 "numActiveTasks" : -1,
 "numCompletedTasks" : 2,
 "numSkippedTasks" : 2,
 "numFailedTasks" : 0,
 "numActiveStages" : 0,
 "numCompletedStages" : 1,
 "numSkippedStages" : 0,
 "numFailedStages" : 0
}]
```

 NOTE

numActiveTasks indicates the number of active tasks.

## Answer

The job information can be obtained in either of the following methods:

- Set **spark.history.briefInfo.gather=true** and then view the brief JobHistory information.
- Visit the JobHistory2x page of Spark (URL: <https://IP:port/api/v1/<appid>/jobs/>).

The value of **numActiveTasks** in the job information is calculated from the difference between the number of SparkListenerTaskStart events and the number of SparkListenerTaskEnd events in the **eventLog** file. If some events are not recorded in the **eventLog** file, the job information obtained from the restful interface is incorrect.

## 24.8.5 Why Cannot I Switch from the Yarn Web UI to the Spark Web UI?

### Question

In FusionInsight, the Spark application is run in yarn-client mode on the client. The following error occurs during the switch from the Yarn web UI to the application web UI:

Error Occurred.

Problem accessing /proxy/application\_1468986660719\_0045/

Powered by Jetty://

The YARN ResourceManager log shows the following information:

```
2016-07-21 16:35:27,099 | INFO | Socket Reader #1 for port 8032 | Auth successful for mapred/
hadoop.<System domain name>@<System domain name> (auth:KERBEROS) | Server.java:1388
2016-07-21 16:35:27,105 | INFO | 1526016381@qtp-1178290888-1015 | admin is accessing unchecked
http://10.120.169.53:23011 which is the app master GUI of
application_1468986660719_0045 owned by spark | WebAppProxyServlet.java:393
2016-07-21 16:36:02,843 | INFO | Socket Reader #1 for port 8032 | Auth successful for hive/
hadoop.<System domain name>@<System domain name> (auth:KERBEROS) | Server.java:1388
2016-07-21 16:36:02,851 | INFO | Socket Reader #1 for port 8032 | Auth successful for hive/
hadoop.<System domain name>@<System domain name> (auth:KERBEROS) | Server.java:1388
2016-07-21 16:36:12,163 | WARN | 1526016381@qtp-1178290888-1015 | /proxy/
application_1468986660719_0045/: java.net.ConnectException: Connection timed out |
Slf4jLog.java:76
2016-07-21 16:37:03,918 | INFO | Socket Reader #1 for port 8032 | Auth successful for hive/
hadoop.<System domain name>@<System domain name> (auth:KERBEROS) | Server.java:1388
2016-07-21 16:37:03,926 | INFO | Socket Reader #1 for port 8032 | Auth successful for hive/
hadoop.<System domain name>@<System domain name> (auth:KERBEROS) | Server.java:1388
2016-07-21 16:37:11,956 | INFO | AsyncDispatcher event handler | Updating application attempt
appattempt_1468986660719_0045_000001 with final state: FINISHING,
and exit status: -1000 | RMAAppAttemptImpl.java:1253
```

## Answer

On FusionInsight Manager, the IP address of the Yarn service is in the 192 network segment.

In Yarn logs, the IP address of Spark web UI read by Yarn is `http://10.120.169.53:23011`, which is in the 10 network segment. The IP addresses in the 192 network segment cannot communicate with those in the 10 network segment. As a result, the Spark web UI fails to be accessed.

Solution:

Log in to the client whose IP address is **10.120.169.53** and change the IP address in the `/etc/hosts` file to the IP address in the 192 network segment. Run the Spark application again. The Spark web UI is displayed.

## 24.8.6 What Can I Do If an Error Occurs when I Access the Application Page Because the Application Cached by HistoryServer Is Recycled?

### Question

An error occurs when I access a Spark application page on the HistoryServer page.

Check the HistoryServer logs. The "FileNotFoundException" exception is found. The related logs are as follows:

```
2016-11-22 23:58:03,694 | WARN | [qtp55429210-232] | /history/application_1479662594976_0001/stages/
stage/ | org.sparkproject.jetty.servlet.ServletHandler.doHandle(ServletHandler.java:628)
java.io.FileNotFoundException: ${BIGDATA_HOME}/tmp/spark/jobHistoryTemp/
blockmgr-5f1f6aca-2303-4290-9845-88fa94d78480/09/temp_shuffle_11f82aaf-e226-46dc-
b1f0-002751557694 (No such file or directory)
```

### Answer

If a Spark application with a large number of tasks is run on the HistoryServer page, the memory overflows to disk and files with the `temp_shuffle` prefix are generated.

By default, HistoryServer caches 50 Spark applications (determined by the `spark.history.retainedApplications` configuration item). When the number of



Spark applications in the memory exceeds 50, HistoryServer reclaims the first cached Spark application and clears the corresponding **temp\_shuffle** file.

When a user is viewing Spark applications to be recycled, the **temp\_shuffle** file may not be found. As a result, the current page cannot be accessed.

If the preceding problem occurs, use either of the following methods to solve the problem:

- Access the HistoryServer page of the Spark application again. The correct page information is displayed.
- If more than 50 Spark applications need to be accessed at the same time, increase the value of **spark.history.retainedApplications**.

Log in to FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Service** > **Spark2x** > **Configuration**, and click **All Configurations**. In the navigation tree on the left, choose **JobHistory2x** > **GUI**, and set parameters.

**Table 24-85** Parameter description

Parameter	Description	Default Value
spark.history.retainedApplications	Number of Spark applications cached by HistoryServer. When the number of applications to be cached exceeds the value of this parameter, HistoryServer reclaims the first cached Spark application.	50

## 24.8.7 Why Is not an Application Displayed When I Run the Application with the Empty Part File?

### Question

When I run an application with an empty part file in HDFS with the log grouping function enabled, why is not the application displayed on the homepage of JobHistory?

### Answer

On the JobHistory page, information about applications is updated only with changed sizes of part files in HDFS. If a file is read for the first time, its size is compared with 0. The file is read only when the file size is greater than 0.

When the log grouping function is enabled, if the application you run does not have jobs in running status, the part file is empty. As a result, JobHistory does not read the part file and the application information is not displayed on the JobHistory page. However, if the size of part file is changed later, the application will be displayed on JobHistory.

## 24.8.8 Why Does Spark2x Fail to Export a Table with the Same Field Name?

### Question

The following code fails to be executed on spark-shell of Spark2x:

```
val acctId = List(("49562", "Amal", "Derry"), ("00000", "Fred", "Xanadu"))
val rddLeft = sc.makeRDD(acctId)
val dfLeft = rddLeft.toDF("Id", "Name", "City")
//dfLeft.show
val acctCustId = List(("Amal", "49562", "CO"), ("Dave", "99999", "ZZ"))
val rddRight = sc.makeRDD(acctCustId)
val dfRight = rddRight.toDF("Name", "CustId", "State")
//dfRight.show
val dfJoin = dfLeft.join(dfRight, dfLeft("Id") === dfRight("CustId"), "outer")
dfJoin.show
dfJoin.repartition(1).write.format("com.databricks.spark.csv").option("delimiter", "\t").option("header", "true").option("treatEmptyValuesAsNulls", "true").option("nullValue", "").save("/tmp/outputDir")
```

### Answer

In Spark2x, the duplicate field name of the **join** statement is checked. You need to modify the code to ensure that no duplicate field exists in the saved data.

## 24.8.9 Why JRE fatal error after running Spark application multiple times?

### Question

Why JRE fatal error after running Spark application multiple times?

### Answer

When you run Spark application multiple times, JRE fatal error occurs and this is due to the problem with the Linux Kernel.

To resolve this issue, upgrade the **kernel version to 4.13.9-2.ge7d7106-default**.

## 24.8.10 "This page can't be displayed" Is Displayed When Internet Explorer Fails to Access the Native Spark2x UI

### Question

Occasionally, Internet Explorer 9, Explorer 10, or Explorer 11 fails to access the native Spark2x UI.

### Symptom

Internet Explorer 9, Explorer 10, or Explorer 11 fails to access the native Spark UI, as shown in the following figure.

Turn on TLS 1.0, TLS 1.1, and TLS 1.2 in Advanced settings and try connecting to

## Cause

Some Internet Explorer 9, Explorer 10, or Explorer 11 versions fail to handle SSL handshake issues, causing access failure.

## Solution

Google Chrome 71 and later versions and Firefox browsers 62 and later versions are recommended.

## 24.8.11 How Does Spark2x Access External Cluster Components?

### Question

There are two clusters, cluster 1 and cluster 2. How do I use Spark2x in cluster 1 to access HDFS, Hive, HBase, and Kafka components in cluster 2?

### Answer

1. Components in two clusters can access each other. However, there are the following restrictions:
  - Only one Hive MetaStore can be accessed. Specifically, Hive MetaStore in cluster 1 and Hive MetaStore in cluster 2 cannot be accessed at the same time.
  - User systems in different clusters are not synchronized. When users access components in another cluster, user permission is determined by the user configuration of the peer cluster. For example, if user A of cluster 1 does not have the permissions to access the HBase meta table in cluster 1 but user A of cluster 2 can access the HBase meta table in cluster 2, user A of cluster 1 can access the HBase meta table in cluster 2.
  - To enable components in a security cluster to communicate with each other across Manager, you need to configure mutual trust.
2. The following describes how to access Hive, HBase, and Kafka components in cluster 2 as user A.

#### NOTE

The following operations are based on the scenario where a user uses the FusionInsight client to submit the Spark2x application. If the user uses the configuration file directory, the user needs to modify the corresponding file in the configuration directory of the application and upload the configuration file to the executor.

When the HDFS and HBase clients access the server, **hostname** is used to configure the server address. Therefore, the hosts configuration of all nodes to be accessed must be saved in the **/etc/hosts** file on the client. You can add the host of the peer cluster node to the **/etc/hosts** file of the client node in advance.

- Access Hive metastore: Replace the **hive-site.xml** file in the **conf** directory of the Spark2x client in cluster 1 with the **hive-site.xml** file in the **conf** directory of the Spark2x client in cluster 2.

After the preceding operations are performed, you can use Spark SQL to access Hive MetaStore. To access Hive table data, you need to perform

the operations in **Access HDFS of two clusters at the same time**: and set **nameservice** of the peer cluster to **LOCATION**.

- Access HBase of the peer cluster.
  - i. Configure the IP addresses and host names of all ZooKeeper nodes and HBase nodes in cluster 2 in the **/etc/hosts** file on the client node of cluster 1.
  - ii. Replace the **hbase-site.xml** file in the **conf** directory of the Spark2x client in cluster 1 with the **hbase-site.xml** file in the **conf** directory of the Spark2x client in cluster 2.
- Access Kafka: Set the address of the Kafka Broker to be accessed to the Kafka Broker address in cluster 2.
- Access HDFS of two clusters at the same time:
  - Two tokens with the same NameService cannot be obtained at the same time. Therefore, the NameServices of the HDFS in two clusters must be different. For example, one is **hacluster**, and the other is **test**.

- 1) Obtain the following configurations from the **hdfs-site.xml** file of cluster2 and add them to the **hdfs-site.xml** file in the **conf** directory of the Spark2x client in cluster1:

**dfs.nameservices.mappings**, **dfs.nameservices**, **dfs.namenode.rpc-address.test.\***, **dfs.ha.namenodes.test**, and **dfs.client.failover.proxy.provider.test**

The following is an example:

```
<property>
<name>dfs.nameservices.mappings</name>
<value>[{"name":"hacluster","roleInstances":["14","15"]},
{"name":"test","roleInstances":["16","17"]}]</value>
</property>
<property>
<name>dfs.nameservices</name>
<value>hacluster,test</value>
</property>
<property>
<name>dfs.namenode.rpc-address.test.16</name>
<value>192.168.0.1:8020</value>
</property>
<property>
<name>dfs.namenode.rpc-address.test.17</name>
<value>192.168.0.2:8020</value>
</property>
<property>
<name>dfs.ha.namenodes.test</name>
<value>16,17</value>
</property>
<property>
<name>dfs.client.failover.proxy.provider.test</name>
<value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider
</value>
</property>
```

- 2) Modify **spark.yarn.extra.hadoopFileSystems = hdfs://test** and **spark.hadoop.hdfs.externalToken.enable = true** in the **spark-defaults.conf** configuration file under the **conf** directory on the Spark client of cluster 1.

```
spark.yarn.extra.hadoopFileSystems = hdfs://test
spark.hadoop.hdfs.externalToken.enable = true
```

- 3) In the application submission command, add the **--keytab** and **--principal** parameters and set them to the user who submits the task in cluster1.
  - 4) Use the Spark client of cluster1 to submit the application. Then, the two HDFS services can be accessed at the same time.
- Access HBase of two clusters at the same time:
    - i. Modify **spark.hadoop.hbase.externalToken.enable = true** in the **spark-defaults.conf** configuration file under the **conf** directory on the Spark client of cluster 1.

```
spark.hadoop.hbase.externalToken.enable = true
```
    - ii. When accessing HBase, you need to use the configuration file of the corresponding cluster to create a **Configuration** object for creating a **Connection** object.
    - iii. In an MRS cluster, tokens of multiple HBase services can be obtained at the same time to solve the problem that the executor cannot access HBase. The method is as follows:

Assume that you need to access HBase of the current cluster and HBase of cluster2. Save the **hbase-site.xml** file of cluster2 in a compressed package named **external\_hbase\_conf\*\*\***, and use **--archives** to specify the compressed package when submitting the command.

## 24.8.12 Why Does the Foreign Table Query Fail When Multiple Foreign Tables Are Created in the Same Directory?

### Question

Assume there is a data file path named **/test\_data\_path**. User A creates a foreign table named **tableA** for the directory, and user B creates a foreign table named **tableB** for the directory. When user B performs the insert operation on **tableB**, user A fails to query data using **tableA** and the error "Permission denied" is displayed.

### Answer

After user B performs the insert operation on **tableB**, a new data file is generated in the foreign table path and the file belongs to user B. When user A queries data using **tableA**, all files in the foreign table directory are read. In this case, the query fails because user A does not have the read permissions on the file generated by user B.

This problem also occurs in other scenarios. For example, the **inset overwrite** operation will also duplicate other table files in this directory.

Due to the Spark SQL implementation mechanism, check restrictions in this scenario will lead to inconsistency and performance deterioration. Therefore, no restriction is added in this scenario, and this method is not recommended.

## 24.8.13 What Should I Do If the Native Page of an Application of Spark2x JobHistory Fails to Display During Access to the Page

### Question

After a Spark application that contains a job with millions of tasks. After the application creation is complete, if you access the native page of the application in JobHistory, the native page of the application can be displayed after a long time. If the native page cannot be displayed within 10 minutes, Error information will be generated for the Proxy.

**Figure 24-20** Error information example

#### Proxy Error

```
The proxy server received an invalid response from an upstream server.
The proxy server could not handle the request GET /Spark2x/JobHistory/77/history/application_1558518306528_0048/1/job/
Reason: Error reading from remote server
```

### Answer

When you switch to the native page of an application on the JobHistory page, JobHistory needs to play back the event log of the application. If the application contains a large number of event logs, the playback takes a long time and the browser takes a long time to navigate you to the native page.

The current browser uses the HTTPd as the proxy to access the JobHistory native page. The proxy timeout duration is 10 minutes. Therefore, if the JobHistory cannot parse the event log and return the result within 10 minutes, the HTTPd automatically returns the proxy error information to the browser.

### Solution

The local disk cache function is enabled on the JobHistory. When a user accesses an application, the event log of the application is cached on the local disk. In this case, the response speed can be greatly accelerated for the second access. Therefore, in this case, you only need to wait for a while and then access the link again. For the second time, you do not need to wait for a long time.

## 24.8.14 Why Do I Fail to Create a Table in the Specified Location on OBS After Logging to spark-beeline?

### Question

When the OBS ECS/BMS image cluster is connected, after spark-beeline is logged in, an error is reported when a location is specified to create a table on OBS.

Figure 24-21 Error message

```
de-master2qCKJ:22550/> create database sparkdb location 'obs://800mrs/sparktest/sparkdb';
0.626 seconds)
de-master2qCKJ:22550/> use sparkdb;
0.072 seconds)
de-master2qCKJ:22550/> create table orc (id int,name string) using orc;
Exception: Configuration problem with provider path. (state=,code=0)
```

## Answer

The permission on the `ssl.jceks` file in HDFS is insufficient. As a result, the table fails to be created.

```
Caused by: org.apache.hadoop.security.AccessControlException: Permission denied: user=root, access=READ, inode="/user/spark2x/jars/8.0.2/ssl.jceks":spark2xhadoop:-rw-----
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:410)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:264)
at com.hawabi.hadoop.adapter.hdfs.plugin.HWAccessControlEnforcer.checkPermission(HWAccessControlEnforcer.java:54)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.checkPermission(FSPermissionChecker.java:194)
at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermission(FSDirectory.java:1957)
at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermission(FSDirectory.java:1941)
at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPathAccess(FSDirectory.java:1891)
at org.apache.hadoop.hdfs.server.namenode.FSFileBlockAccessOp.getBlockLocations(FSFileBlockAccessOp.java:175)
at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getBlockLocations(FSNamesystem.java:1950)
at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.getBlockLocations(NameNodeRpcServer.java:762)
at org.apache.hadoop.hdfs.protocolPB.ClientNameNodeProtocolServerSideTranslatorPB.getBlockLocations(ClientNameNodeProtocolServerSideTranslatorPB.java:445)
at org.apache.hadoop.hdfs.protocol.proto.ClientNameNodeProtocolProtosClientNameNodeProtocol2.callBlockingMethod(ClientNameNodeProtocolProtos.java)
at org.apache.hadoop.ipc.ProtocolHandlerEngine$Server$ProtocolHandlerEngine.call(ProtocolHandlerEngine.java:528)
at org.apache.hadoop.ipc.RPCServer.call(RPC.java:1036)
at org.apache.hadoop.ipc.Server$RPCCall.run(Server.java:985)
at org.apache.hadoop.ipc.Server$RPCCall.run(Server.java:913)
at java.security.AccessController.doPrivileged(Native Method)
at javax.security.auth.Subject.doAs(Subject.java:422)
at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1737)
at org.apache.hadoop.ipc.ServerHandler.run(Server.java:2876)
```

## Solution

1. Log in to the node where Spark2x resides as user `omm` and run the following command:  
`vi ${BIGDATA_HOME}/FusionInsight_Spark2x_8.0.2.1/install/FusionInsight-Spark2x-2.4.5/spark/sbin/fake_prestart.sh`
2. Change `eval "${hdfsCmd}" -chmod 600 "${InnerHdfsDir}"/ssl.jceks >> "${PRESTART_LOG}" 2>&1` to `eval "${hdfsCmd}" -chmod 644 "${InnerHdfsDir}"/ssl.jceks >> "${PRESTART_LOG}" 2>&1`.
3. Restart the SparkResource instance.

## 24.8.15 Spark Shuffle Exception Handling

### Question

In some scenarios, the following exception occurs in the Spark shuffle phase:

```
2021-06-18 02:53:08.364 INFO [shuffle-server-6-1] | DIGEST1:Unmatched MACs | javax.security.sasl.unwrap(DigestMD5Base.java:148)
2021-06-18 02:53:08.368 WARN [shuffle-server-6-1] | Exception in connection from /XXXXXXXXXXXXXXXXXXXX | org.apache.spark.network.server.TransportChannelHandler.exceptionCaught(TransportChannelHandler.java:97)
io.netty.handler.codec.DecoderException: javax.security.sasl.SaslException: DIGEST-MD5: Out of order sequencing of messages from server. Got: 16 Expected: 14
at io.netty.handler.codec.MessageToMessageDecoder.channelRead(MessageToMessageDecoder.java:98)
at io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChannelHandlerContext.java:379)
at io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelHandlerContext.java:365)
at org.apache.spark.network.util.TransportFrameDecoder.channelRead(TransportFrameDecoder.java:102)
at io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChannelHandlerContext.java:379)
at io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelHandlerContext.java:365)
at io.netty.channel.DefaultChannelPipeline.fireChannelRead(DefaultChannelPipeline.java:145)
at io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChannelHandlerContext.java:379)
at io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelHandlerContext.java:365)
at io.netty.channel.DefaultChannelPipeline.fireChannelRead(DefaultChannelPipeline.java:919)
at io.netty.channel.nio.AbstractNioByteChannel$NioByteUnsafe.read(AbstractNioByteChannel.java:162)
at io.netty.channel.nio.NioEventLoop.processSelectedKey(NioEventLoop.java:714)
at io.netty.channel.nio.NioEventLoop.processSelectedKeysOptimized(NioEventLoop.java:650)
at io.netty.channel.nio.NioEventLoop.processSelectedKeys(NioEventLoop.java:576)
at io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:493)
at io.netty.util.concurrent.SingleThreadEventExecutor$5.run(SingleThreadEventExecutor.java:999)
at io.netty.util.internal.ThreadExecutorMap$2.run(ThreadExecutorMap.java:74)
at io.netty.util.concurrent.FastThreadLocalRunnable.run(FastThreadLocalRunnable.java:30)
at java.lang.Thread.run(Thread.java:748)
Caused by: javax.security.sasl.SaslException: DIGEST-MD5: Out of order sequencing of messages from server. Got: 16 Expected: 14
at com.sun.security.sasl.digest.DigestMD5Base.unwrap(DigestMD5Base.java:148)
at com.sun.security.sasl.digest.DigestMD5Base.unwrap(DigestMD5Base.java:213)
at org.apache.spark.network.sasl.SaslSaslServer.unwrap(SaslSaslServer.java:149)
at org.apache.spark.network.sasl.SslEncryptionDecryptionHandler.decode(SslEncryption.java:126)
at org.apache.spark.network.sasl.SslEncryptionDecryptionHandler.decode(SslEncryption.java:103)
at io.netty.handler.codec.MessageToMessageDecoder.channelRead(MessageToMessageDecoder.java:88)
at io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChannelHandlerContext.java:379)
at io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelHandlerContext.java:365)
at org.apache.spark.network.util.TransportFrameDecoder.channelRead(TransportFrameDecoder.java:102)
at io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChannelHandlerContext.java:379)
at io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelHandlerContext.java:365)
at io.netty.channel.DefaultChannelPipeline.fireChannelRead(DefaultChannelPipeline.java:145)
at io.netty.channel.AbstractChannelHandlerContext.invokeChannelRead(AbstractChannelHandlerContext.java:379)
at io.netty.channel.AbstractChannelHandlerContext.fireChannelRead(AbstractChannelHandlerContext.java:365)
at io.netty.channel.DefaultChannelPipeline.fireChannelRead(DefaultChannelPipeline.java:919)
at io.netty.channel.nio.AbstractNioByteChannel$NioByteUnsafe.read(AbstractNioByteChannel.java:162)
at io.netty.channel.nio.NioEventLoop.processSelectedKey(NioEventLoop.java:714)
at io.netty.channel.nio.NioEventLoop.processSelectedKeysOptimized(NioEventLoop.java:650)
at io.netty.channel.nio.NioEventLoop.processSelectedKeys(NioEventLoop.java:576)
at io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:493)
at io.netty.util.concurrent.SingleThreadEventExecutor$5.run(SingleThreadEventExecutor.java:999)
at io.netty.util.internal.ThreadExecutorMap$2.run(ThreadExecutorMap.java:74)
at io.netty.util.concurrent.FastThreadLocalRunnable.run(FastThreadLocalRunnable.java:30)
at java.lang.Thread.run(Thread.java:748)
```

## Solution

For JDBC:

Log in to FusionInsight Manager, change the value of the JDBCServer parameter **spark.authenticate.enableSaslEncryption** to **false**, and restart the corresponding instance.

For client jobs:

When the client submits the application, change the value of **spark.authenticate.enableSaslEncryption** in the **spark-defaults.conf** file to **false**.



# 25 Using Storm

---

## 25.1 Using Storm from Scratch

You can submit and delete Storm topologies on the MRS cluster client.

### Prerequisites

The MRS cluster client has been installed, for example, in the `/opt/hadoopclient` directory. The client directory in the following operations is only an example. Change it based on the actual installation directory onsite.

### Procedure

**Step 1** Prepare the client based on service requirements. Log in to the node where the client is installed.

Log in to the node where the client is installed. For details, see [Installing a Client](#).

**Step 2** Run the following command to switch to the client directory, for example, `/opt/hadoopclient`:

```
cd /opt/hadoopclient
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 4** For clusters with Kerberos authentication enabled, run the following command to authenticate the user. For clusters with Kerberos authentication disabled, skip this step.

```
kinit Storm user
```

**Step 5** Run the following command to submit the Storm topology:

```
storm jar Path of the topology package Class name of the topology Main method
Topology name
```

If the following information is displayed, the topology is submitted successfully.

```
Finished submitting topology: topo1
```

**Step 6** Run the following command to query Storm topologies. For clusters with Kerberos authentication enabled, only users in the **stormadmin** or **storm** group can query all topologies.

```
storm list
```

**Step 7** Run the following command to delete a Storm topology.

```
storm kill Topology name
```

```
----End
```

## 25.2 Using the Storm Client

### Scenario

This section describes how to use the Storm client in an O&M scenario or service scenario.

### Prerequisites

- You have installed the client. For example, the installation directory is **/opt/hadoopclient**.
- Service component users are created by the MRS cluster administrator as required. In security mode, machine-machine users have downloaded the keytab file. A human-machine user must change the password upon the first login. (Not involved in normal mode)

### Procedure

**Step 1** Prepare the client based on service requirements. Log in to the node where the client is installed.

Log in to the node where the client is installed. For details, see [Installing a Client](#).

**Step 2** Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 4** If multiple Storm instances are installed, run the following command to load the environment variables of a specific instance when running the Storm command to submit the topology. Otherwise, skip this step. The following command uses the instance Storm-2 as an example.

```
source Storm-2/component_env
```

**Step 5** Run the following command to perform user authentication (skip this step in normal mode):

```
kinit Component service user
```

**Step 6** Run the following command to perform operations on the client:

For example, run the following command:

- `cql`
- `storm`

 NOTE

A Storm client cannot be connected to secure and non-secure ZooKeepers at the same time.

----End

## 25.3 Submitting Storm Topologies on the Client

### Scenario

You can submit Storm topologies on the cluster client to continuously process stream data. For clusters with Kerberos authentication enabled, users who submit topologies must be members of the `stormadmin` or `storm` group.

### Prerequisites

The client has been updated.

### Procedure

**Step 1** Prepare the client based on service requirements. Log in to the node where the client is installed.

Log in to the node where the client is installed. For details, see [Installing a Client](#).

**Step 2** Run the following command to set the permissions on the topology JAR file:

For example, run the following command to change the permissions on `/opt/storm/topology.jar`:

```
chmod 600 /opt/storm/topology.jar
```

**Step 3** Run the following command to switch to the client directory, for example, `/opt/client`.

```
cd /opt/client
```

**Step 4** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 5** If multiple Storm instances are installed, run the following command to load the environment variables of a specific instance when running the Storm command to submit the topology. Otherwise, skip this step. The following command uses the instance Storm-2 as an example.

```
source Storm-2/component_env
```

**Step 6** For clusters with Kerberos authentication enabled, run the following command to authenticate the user. For clusters with Kerberos authentication disabled, skip this step.

```
kinit Storm user
```

**Step 7** For versions earlier than MRS 3.x, run the following command to submit the Storm topology:

```
storm jar Path of the topology package Class name of the topology Main method Topology name
```

If the following information is displayed, the topology is submitted successfully.

```
Finished submitting topology: topo1
```

 **NOTE**

- To support sampling messages, add the **topology.debug** and **topology.eventlogger.executors** parameters.
- Data processing methods vary with topologies. The topology in the example generates characters randomly and separates character strings. To query the processing status, enable the sampling function and perform operations according to [Querying Storm Topology Logs](#).

**Step 8** Run the following command to submit a topology task for MRS 3.x or later:

```
storm jar topology-jar-path class input parameter list
```

- *topology-jar-path* indicates the path of the JAR file of the topology.
- *class* indicates the class name of the main method used by the topology.
- *Input parameter list* includes input parameters of the main method used by the topology.

If the following information is displayed, the topology is submitted successfully:

```
Finished submitting topology: topology1
```

 **NOTE**

- The login authentication user must correspond to the loaded environment variable (**component\_env**). Otherwise, an error occurs when you run the **storm** command to submit the topology task.
- After the client environment variable is loaded and the corresponding user login succeeds, the user can run the Storm command on any Storm client to submit the topology task. After the command is executed, the successfully submitted topology is still in the Storm cluster of the user.
- If cluster domain name is modified, you need to reset the domain name before submitting the topology. Run the cql statement.

**Step 9** Run the following command to query Storm topologies. For clusters with Kerberos authentication enabled, only users in the **stormadmin** or **storm** group can query all topologies.

```
storm list
```

```
----End
```

## 25.4 Accessing the Storm Web UI

### Scenario

The Storm web UI provides a graphical interface for using Storm.

The following information can be queried on the Storm web UI:

- Storm cluster summary
- Nimbus summary
- Topology summary
- Supervisor summary
- Nimbus configurations

## Prerequisites

- The password of user **admin** has been obtained. The password of user **admin** is specified by you during the cluster creation.
- If a user other than **admin** is used to access the Storm web UI, the user must be added to the **storm** or **stormadmin** user group.

## Procedure

**Step 1** Access the component management page.

- For versions earlier than MRS 3.x, click the cluster name to go to the cluster details page and choose **Components**.

### NOTE

If the **Components** tab is unavailable, complete IAM user synchronization first. (On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.)

- For MRS 3.x or later, log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Choose **Cluster** > *Name of the desired cluster* > **Services**.

**Step 2** Log in to the Storm WebUI.

- For versions earlier than MRS 3.x: Choose **Storm**. On the **Storm Summary** area, click any UI link on the right side of **Storm Web UI** to open the Storm web UI.

### NOTE

When accessing the Storm web UI for the first time, you must add the address to the trusted site list.

- For MRS 3.x or later, choose **Storm** > **Overview**. In the **Basic Information** area, click any UI link on the right side of **Storm Web UI** to open the Storm web UI.

----End

## Related Tasks

- Click a topology name to view details, status, Spouts information, Bolts information, and configuration information of the topology.
- In the **Topology actions** area, click **Activate**, **Deactivate**, **Rebalance**, **Kill**, **Debug**, **Stop Debug**, and **Change Log Level** to activate, deactivate, redeploy, delete, debug, and stop debugging the topology, and modify the log levels, respectively. You need to set the waiting time for the redeployment and deletion operations. The unit is second.

- In the **Topology Visualization** area, click **Show Visualization** to visualize a topology. After the topology is visualized, the WebUI displays the topology structure.

## 25.5 Managing Storm Topologies

### Scenario

You can manage Storm topologies on the Storm web UI. Users in the **storm** group can manage only the topology tasks submitted by themselves, while users in the **stormadmin** group can manage all topology tasks.

### Procedure

**Step 1** For details about how to access the Storm WebUI, see [Accessing the Storm Web UI](#).

**Step 2** In the **Topology summary** area, click the desired topology.

**Step 3** Use options in **Topology actions** to manage the Storm topology.

- Activating a topology  
Click **Activate** to activate the topology.
- Deactivating a topology  
Click **Deactivate** to deactivate the topology.
- Re-deploying a topology  
Click **Rebalance** and specify the wait time (in seconds) of re-deployment. Generally, if the number of nodes in a cluster changes, the topology can be re-deployed to maximize resource usage.
- Deleting a topology  
Click **Kill** and specify the wait time (in seconds) of the deletion.
- Starting or stopping sampling messages  
Click **Debug**. In the dialog box displayed, specify the percentage of the sampled data volume. For example, if the value is set to **10**, 10% of data is sampled.  
To stop sampling, click **Stop Debug**.

#### NOTE

This function is available only if the sampling function is enabled when the topology is submitted. For details about querying data processing information, see [Querying Storm Topology Logs](#).

- Modifying the topology log level  
Click **Change Log Level** to specify a new log level.

**Step 4** Displaying a topology

In the **Topology Visualization** area, click **Show Visualization** to visualize the topology.

----End

## 25.6 Querying Storm Topology Logs

### Scenario

You can query topology logs to check the execution of a Storm topology in a worker process. To query the data processing logs of a topology, enable the **Debug** function when submitting the topology. Only streaming clusters with Kerberos authentication enabled support this function. In addition, the user who queries topology logs must be the one who submits the topology or a member of the **stormadmin** group.

### Prerequisites

- The network of the working environment has been configured.
- The sampling function has been enabled for the topology.

### Querying Worker Process Logs

**Step 1** For details about how to access the Storm WebUI, see [Accessing the Storm Web UI](#).

**Step 2** In the **Topology Summary** area, click the desired topology to view details.

**Step 3** Click the desired **Spouts** or **Bolts** task. In the **Executors (All time)** area, click a port in **Port** to view detailed logs.

----End

### Querying Data Processing Logs of a Topology

**Step 1** For details about how to access the Storm WebUI, see [Accessing the Storm Web UI](#).

**Step 2** In the **Topology Summary** area, click the desired topology to view details.

**Step 3** Click **Debug**, specify the data sampling ratio, and click **OK**.

**Step 4** Click the **Spouts** or **Bolts** task. In **Component summary**, click **events** to view data processing logs.

----End

## 25.7 Storm Common Parameters

This section applies to MRS 3.x or later.

### Navigation Path

For details about how to set parameters, see [Modifying Cluster Service Configuration Parameters](#).

## Parameter Description

**Table 25-1** Parameter description

Parameter	Description	Default Value
supervisor.slots.ports	Specifies the list of ports that can run workers on the supervisor. Each worker occupies a port, and each port runs only one worker. This parameter is used to set the number of workers that can run on each server. Ports range from 1024 to 65535, and ports are separated by commas (,).	6700,6701,6702,6703
WORKER_GC_OPTS	Specifies the JVM option used for supervisor to start worker. It is recommended that you set this parameter based on memory usage of a service. For simple service processing, the recommended value is <b>-Xmx1G</b> . If window cache is used, the value of this parameter is calculated based on the following formula: Size of each record x Period x 2	-Xms1G -Xmx1G -XX:+UseG1GC -XX:+PrintGCDetails -Xloggc:artifacts/gc.log -XX:+PrintGCDateStamps -XX:+PrintGCTimeStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -XX:GCLogFileSize=1M -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=artifacts/heapdump
default.scheduler.mode	Specifies the default scheduling mode of the scheduler. Options are as follows: <ul style="list-style-type: none"> <li>● <b>AVERAGE</b>: indicates that the scheduling mechanism that uses the number of idle slots as the priority is used.</li> <li>● <b>RATE</b>: indicates that the scheduling mechanism that uses the rate of idle slots as the priority is used.</li> </ul>	AVERAGE
nimbus.thrift.threads	Set the maximum number of connection threads when the active Nimbus externally provides services. If the Storm cluster is large and the number of Supervisor instances is large, increase connection threads.	512



## 25.8 Configuring a Storm Service User Password Policy

### Scenario

This section applies to MRS 3.x or later.

After submitting a topology task, a Storm service user must ensure that the task continuously runs. During topology running, the worker process may need to restart to ensure continuous topology work. If the password of a service user is changed or the number of days that a password is used exceeds the maximum number specified in a password policy, topology running may be affected. A MRS cluster administrator must configure a separate password policy for Storm service users based on enterprise security requirements.

#### NOTE

If a separate password policy is not configured for Storm service users, an old topology can be deleted and then submitted again after a service user password is changed so that the topology can continuous run.

### Impact on the System

- After a separate password policy is configured for a Storm service user, the user is not affected by **Password Policy** on the Manager page.
- If a separate password policy is configured for a Storm service user and cross-cluster entrusted relationships are configured, a password must be reset for the Storm service user on Manager based on the password policy.

### Prerequisites

You have understood service requirements and created a **Human-Machine** user, for example, **testpol**.

### Procedure

**Step 1** Log in to any node in the cluster as user **omm**.

**Step 2** Run the following command to disable logout upon timeout:

```
TMOUT=0
```

#### NOTE

After the operations in this section are complete, run the **TMOUT=Timeout interval** command to restore the timeout interval in a timely manner. For example, **TMOUT=600** indicates that a user is logged out if the user does not perform any operation within 600 seconds.

**Step 3** Run the following commands to export the environment variables:

```
EXECUTABLE_HOME="${CONTROLLER_HOME}/kerberos_user_specific_binay/
kerberos"
```

```
LD_LIBRARY_PATH=${EXECUTABLE_HOME}/lib:$LD_LIBRARY_PATH
```

```
PATH=${EXECUTABLE_HOME}/bin:$PATH
```

- Step 4** Run the following command and enter the Kerberos administrator password to log in to the Kerberos console:

```
kadmin -p kadmin/admin
```

 **NOTE**

For initial use, the **kadmin/admin** password must be changed for the **kadmin/admin** user.

If the following information is displayed, you have successfully logged in to the Kerberos console.

```
kadmin:
```

- Step 5** Run the following command to check details about the created **Human-Machine** user:

```
getprinc Username
```

Sample command for viewing details about the **testpol** user:

```
getprinc testpol
```

If the following information is displayed, the specified user has used the default password policy:

```
Principal: testpol@<System domain name>
.....
Policy: default
```

- Step 6** Run the following command to create a separate password policy, such as **streampol**, for the Storm service user:

```
addpol -maxlife 0day -minlife 0sec -history 1 -maxfailure 5 -
failurecountinterval 5min -lockoutduration 5min -minlength 8 -minclasses 4
streampol
```

In the command, **-maxlife** indicates the maximum validity period of a password, and **0day** indicates that a password will never expire.

- Step 7** Run the following command to view the newly created policy **streampol**:

```
getpol streampol
```

If the following information is displayed, the new policy specifies that the password will never expire:

```
Policy: streampol
Maximum password life: 0 days 00:00:00
.....
```

- Step 8** Run the following command to apply the new policy **streampol** to the **testpol** Storm user:

```
modprinc -policy streampol testpol
```

In the command, **streampol** indicates a policy name, and **testpol** indicates a username.

If the following information is displayed, the properties of the specified user have been modified:

```
Principal "testpol@<System domain name>" modified.
```

**Step 9** Run the following command to view current information about the **testpol** Storm user:

**getprinc testpol**

If the following information is displayed, the specified user has used the new password policy:

```
Principal: testpol@<System domain name>
```

```
.....
Policy: streampol
```

```
----End
```

## 25.9 Migrating Storm Services to Flink

### 25.9.1 Overview

This section applies to MRS 3.x or later.

From 0.10.0, Flink provides a set of APIs to smoothly migrate services compiled using Storm APIs to the Flink platform. This can be used in most of the service scenarios.

Flink supports the following service migration modes:

1. Complete migration of Storm services: Convert and run a complete Storm topology developed by Storm APIs.
2. Embedded migration of Storm services: Storm code is embedded in DataStream of Flink, for example, Spout/Bolt compiled using Storm APIs.

Flink provides the flink-storm package for the preceding service migration.

### 25.9.2 Completely Migrating Storm Services

#### Scenarios

This section describes how to convert and run a complete Storm topology developed using Storm API.

#### Procedure

**Step 1** Open the Storm service project, modify the POM file of the project, and add the reference of **flink-storm\_2.11**, **flink-core**, and **flink-streaming-java\_2.11**. The following figure shows an example.

```
<dependency>
 <groupId>org.apache.flink</groupId>
 <artifactId>flink-storm_2.11</artifactId>
 <version>1.4.0</version>
 <exclusions>
 <exclusion>
 <groupId>*</groupId>
 <artifactId>*</artifactId>
 </exclusion>
 </exclusions>
```

```

 </exclusions>
 </dependency>
 <dependency>
 <groupId>org.apache.flink</groupId>
 <artifactId>flink-core</artifactId>
 <version>1.4.0</version>
 <exclusions>
 <exclusion>
 <groupId>*</groupId>
 <artifactId>*</artifactId>
 </exclusion>
 </exclusions>
 </dependency>

 <dependency>
 <groupId>org.apache.flink</groupId>
 <artifactId>flink-streaming-java_2.11</artifactId>
 <version>1.4.0</version>
 <exclusions>
 <exclusion>
 <groupId>*</groupId>
 <artifactId>*</artifactId>
 </exclusion>
 </exclusions>
 </dependency>

```

 **NOTE**

If the project is not a non-Maven project, manually collect the preceding JAR packages and add them to the `classpath` environment variable of the project.

**Step 2** Modify the code for submission of the topology. The following uses WordCount as an example:

1. Keep the structure of the Storm topology unchanged, including the Spout and Bolt developed using Storm API.

```

TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("spout", new RandomSentenceSpout(), 5);
builder.setBolt("split", new SplitSentenceBolt(), 8).shuffleGrouping("spout");
builder.setBolt("count", new WordCountBolt(), 12).fieldsGrouping("split", new Fields("word"));

```

2. Modify the code for submission of the topology. An example is described as follows:

```

Config conf = new Config();
conf.setNumWorkers(3);
StormSubmitter.submitTopology("word-count", conf, builder.createTopology());

```

Perform the following operations:

```

Config conf = new Config();
conf.setNumWorkers(3);
//converts Storm Config to StormConfig of Flink.
StormConfig stormConfig = new StormConfig(conf);
//Construct FlinkTopology using TopologBuilder of Storm.
FlinkTopology topology = FlinkTopology.createTopology(builder);
//Obtain the Stream execution environment.
StreamExecutionEnvironment env = topology.getExecutionEnvironment();
//Set StormConfig to the environment variable of Job to construct Bolt and Spout.
//If StormConfig is not required during the initialization of Bolt and Spout, you do not need to set this parameter.
env.getConfig().setGlobalJobParameters(stormConfig);
//Submit the topology.
topology.execute();

```

3. After the package is repacked, run the following command to submit the package:

```
flink run -class {MainClass} WordCount.jar
```

```
----End
```

## 25.9.3 Performing Embedded Service Migration

### Scenarios

This section describes how to embed Storm code in DataStream of Flink in embedded migration mode. For example, the code of Spout or Bolt compiled using Storm API is embedded.

### Procedure

- Step 1** In Flink, perform embedded conversion to Spout and Bolt in the Storm topology to convert them to Flink operators. The following is an example of the code:

```
//set up the execution environment
final StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();
//get input data
final DataStream<String> text = getTextDataStream(env);
final DataStream<Tuple2<String, Integer>> counts = text
 //split up the lines in pairs (2-tuples) containing: (word,1)
 //this is done by a bolt that is wrapped accordingly
 .transform("CountBolt",
 TypeExtractor.getForObject(new Tuple2<String, Integer>("", 0)),
 new BoltWrapper<String, Tuple2<String, Integer>>(new CountBolt()))
 //group by the tuple field "0" and sum up tuple field "1"
 .keyBy(0).sum(1);
// execute program
env.execute("Streaming WordCount with bolt tokenizer");
```

- Step 2** After the modification, run the following command to submit the modification:

```
flink run -class {MainClass} WordCount.jar
```

```
----End
```

## 25.9.4 Migrating Services of External Security Components Interconnected with Storm

### Migrating Services for Interconnecting Storm with HDFS and HBase

If the Storm services use the **storm-hdfs** or **storm-hbase** plug-in package for interconnection, you need to specify the following security parameters when migrating Storm services as instructed in [Completely Migrating Storm Services](#).

```
//Initialize Storm Config.
Config conf = new Config();

//Initialize the security plug-in list.
List<String> auto_tgts = new ArrayList<String>();
//Add the AutoTGT plug-in.
auto_tgts.add("org.apache.storm.security.auth.kerberos.AutoTGT");
//Add the AutoHDFS plug-in.
//If HBase is interconnected, use auto_tgts.add("org.apache.storm.hbase.security.AutoHBase") to replace the
following:
auto_tgts.add("org.apache.storm.hdfs.common.security.AutoHDFS");

//Set security parameters.
conf.put(Config.TOPOLOGY_AUTO_CREDENTIALS, auto_tgts);
```

```
//Set the number of workers.
conf.setNumWorkers(3);

//Convert Storm Config to StormConfig of Flink.
StormConfig stormConfig = new StormConfig(conf);

//Construct FlinkTopology using TopologBuilder of Storm.
FlinkTopology topology = FlinkTopology.createTopology(builder);

//Obtain the StreamExecutionEnvironment.
StreamExecutionEnvironment env = topology.getExecutionEnvironment();

//Add StormConfig to the environment variable of Job to construct Bolt and Spout.
//If Config is not required during the initialization of Bolt and Spout, do not set this parameter.
env.getConfig().setGlobalJobParameters(stormConfig);

//Submit the topology.
topology.execute();
```

After the preceding security plug-in is configured, unnecessary logins during the initialization of HDFS Bolt and HBase Bolt are avoided because the security context has been configured in Flink.

## Migrating Services of Storm Interconnected with Other Security Components

If the plug-in packages, such as **storm-kakfa-client** and **storm-solr** are used for interconnection between Storm and other components for service migration, the previously configured security plug-ins need to be deleted.

```
List<String> auto_tgts = new ArrayList<String>();
//keytab mode
auto_tgts.add("org.apache.storm.security.auth.kerberos.AutoTGTFFromKeytab");

//Write the plug-in list configured on the client to the specified config parameter.
//Mandatory in security mode
//This configuration is not required in common mode, and you can comment out the following line.
conf.put(Config.TOPOLOGY_AUTO_CREDENTIALS, auto_tgts);
```

The AutoTGTFFromKeytab plug-in must be deleted during service migration. Otherwise, the login will fail when Bolt or Spout is initialized.

## 25.10 Storm Log Introduction

This section applies to MRS 3.x or later.

### Log Description

Log paths: The default paths of Storm log files are **/var/log/Bigdata/storm/Role name** (run logs) and **/var/log/Bigdata/audit/storm/Role name** (audit logs).

- Nimbus: **/var/log/Bigdata/storm/nimbus** (run logs) and **/var/log/Bigdata/audit/storm/nimbus** (audit logs)
- Supervisor: **/var/log/Bigdata/storm/supervisor** (run logs) and **/var/log/Bigdata/audit/storm/supervisor** (audit logs)
- UI: **/var/log/Bigdata/storm/ui** (run logs) and **/var/log/Bigdata/audit/storm/ui** (audit logs)
- Logviewer: **/var/log/Bigdata/storm/logviewer** (run logs) and **/var/log/Bigdata/audit/storm/logviewer** (audit logs)

Log archive rule: The automatic Storm log compression function is enabled. By default, when the size of logs exceeds 10 MB, logs are automatically compressed into a log file named in the following format: *<Original log name>.log.[ID].gz*. A maximum of 20 latest compressed files are reserved by default. You can configure the number of compressed files and the compression threshold.

Names of compressed audit log files are in the format of **audit.log.[yyyy-MM-dd].[ID].zip**. These files permanently exist.

**Table 25-2** Storm log list

Log Type	Log File Name	Description
Run log	nimbus/access.log	Nimbus user access log
	nimbus/nimbus-<PID>-gc.log	GC log of the Nimbus process
	nimbus/checkavailable.log	Nimbus availability check log
	nimbus/checkService.log	Nimbus serviceability check log
	nimbus/metrics.log	Nimbus monitoring statistics log
	nimbus/nimbus.log	Run log of the Nimbus process
	nimbus/postinstall.log	Work log after Nimbus installation
	nimbus/prestart.log	Work log before Nimbus startup
	nimbus/start.log	Work log of Nimbus startup
	nimbus/stop.log	Work log of Nimbus shutdown
	supervisor/access.log	Supervisor access log
	supervisor/metrics.log	Supervisor monitoring statistics log
	supervisor/postinstall.log	Work log after supervisor installation
	supervisor/prestart.log	Work log before supervisor startup
	supervisor/start.log	Work log of supervisor startup
	supervisor/stop.log	Work log of supervisor shutdown

Log Type	Log File Name	Description
	supervisor/supervisor.log	Run log of the supervisor process
	supervisor/supervisor- <PID>-gc.log	GC log of the supervisor process
	ui/access.log	UI access log
	ui/metric.log	UI monitoring statistics log
	ui/ui-<PID>-gc.log	GC log of the UI process
	ui/postinstall.log	Work log after UI installation
	ui/prestart.log	Work log before UI startup
	ui/start.log	Work log of UI startup
	ui/stop.log	Work log of UI shutdown
	ui/ui.log	Run log of the UI process
	logviewer/access.log	Logviewer access log
	logviewer/metric.log	Logviewer monitoring statistics log
	logviewer/logviewer- <PID>-gc.log	GC log file of the logviewer process
	logviewer/logviewer.log	Run log of the logviewer process
	logviewer/postinstall.log	Work log after logviewer installation
	logviewer/prestart.log	Work log before logviewer startup
	logviewer/start.log	Work log of logviewer startup
	logviewer/stop.log	Work log of logviewer shutdown
	supervisor/[topologyId]- worker-[Port number].log	Run log of the Worker process. One port occupies one log file. By default, the system contains five ports: 29100, 29101, 29102, 29103 and 29304.



Log Type	Log File Name	Description
	supervisor/metadata/[topologyid]-worker-[Port number].yaml	Worker log metadata file, which is used by logviewer to delete logs. This file is automatically deleted by the logviewer log deletion thread based on certain conditions.
	nimbus/cleanup.log	Cleanup log of Nimbus uninstallation
	logviewer/cleanup.log	Cleanup log of logviewer uninstallation
	ui/cleanup.log	Cleanup log of UI uninstallation
	supervisor/cleanup.log	Cleanup log of supervisor uninstallation
	leader_switch.log	Run log file that records the Storm active/standby switchover
Audit log	nimbus/audit.log	Nimbus audit log
	ui/audit.log	UI audit log
	supervisor/audit.log	Supervisor audit log
	logviewer/audit	Logviewer audit log

## Log Levels

[Table 25-3](#) describes the log levels supported by Storm.

Levels of run logs and audit logs are ERROR, WARN, INFO, and DEBUG from the highest to the lowest priority. Run logs of equal or higher levels are recorded. The higher the specified log level, the fewer the logs recorded.

**Table 25-3** Log levels

Level	Description
ERROR	Logs of this level record error information about system running.
WARN	Logs of this level record exception information about the current event processing.

Level	Description
INFO	Logs of this level record normal running status information about the system and events.
DEBUG	Logs of this level record the system information and system debugging information.

To modify log levels, perform the following operations:

- Step 1** Go to the **All Configurations** page of Storm by referring to [Modifying Cluster Service Configuration Parameters](#).
- Step 2** On the menu bar on the left, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Save the configuration. In the displayed dialog box, click **OK** to make the configurations take effect.

----End

## Log Format

The following table lists the Storm log formats:

**Table 25-4** Log Formats

Log Type	Format	Example
Run log	%d{yyyy-MM-dd HH:mm:ss,SSS}   %-5p   [%t]   %m   %logger (%F:%L) %n	2015-03-11 23:04:00,241   INFO   [RMI TCP Connection(2646)-10.0.0.2]   The baseSleepTimeMs [1000] the maxSleepTimeMs [1000] the maxRetries [1]   backtype.storm.utils.StormBoundedExponentialBackoffRetry (StormBoundedExponentialBackoffRetry.java:46)
	<yyyy-MM-dd HH:mm:ss,SSS><HostName><RoleName><logLevel><Message>	2017-03-28 02:57:52 493 10-5-146-1 storm- INFO Nimbus start normally

Log Type	Format	Example
Audit log	<i>&lt;Username&gt;&lt;User IP address&gt;&lt;Time&gt;&lt;Operation&gt;&lt;Operation object&gt;&lt;Operation result&gt;</i>	UserName=storm/hadoop, UserIP=10.10.0.2, Time=Tue Mar 10 01:15:35 CST 2015, Operation=Kill, Resource=test, Result=Success

## 25.11 Performance Tuning

### 25.11.1 Storm Performance Tuning

#### Scenario

You can modify Storm parameters to improve Storm performance in specific service scenarios.

This section applies to MRS 3.x or later.

Modify the service configuration parameters. For details, see [Modifying Cluster Service Configuration Parameters](#).

#### Topology Tuning

This task enables you to optimize topologies to improve efficiency for Storm to process data. It is recommended that topologies be optimized in scenarios with lower reliability requirements.

**Table 25-5** Tuning parameters

Parameter	Default Value	Scenario
topology.acker.executors	null	Specifies the number of acker executors. If a service application has lower reliability requirements and certain data does not need to be processed, this parameter can be set to <b>null</b> or <b>0</b> so that you can set acker off, flow control is weakened, and message delay is not calculated. This improves performance.

Parameter	Default Value	Scenario
topology.max.spout.pending	null	Specifies the number of messages cached by spout. The parameter value takes effect only when acker is not <b>0</b> or <b>null</b> . Spout adds each message sent to downstream bolt into the pending queue. The message is removed from the queue after downstream bolt processes the message and the processing is confirmed. When the pending queue is full, spout stops sending messages. Increasing the pending value improves the message throughput of spout per second but prolongs the delay.
topology.transfer.buffer.size	32	Specifies the size of the Distruptor message queue for each worker process. It is recommended that the size be between 4 to 32. Increasing the queue size improves the throughput but may prolong the delay.
RES_CPUSET_PERCENTAGE	80	Specifies the percentage of physical CPU resources used by the supervisor role instance (including startup and management worker processes) on each node. Adjust the parameter value based on service volume requirements of the node on which the supervisor exists, to optimize CPU usage.

## JVM Tuning

If an application must occupy more memory resources to process a large volume of data and the size of worker memory is greater than 2 GB, the G1 garbage collection algorithm is recommended.

**Table 25-6** Tuning parameters

Parameter	Default Value	Scenario
WORKER_GC_OPTS	-Xms1G -Xmx1G -XX:+UseG1GC -XX:+PrintGCDetails -Xloggc:artifacts/gc.log -XX:+PrintGCDateStamps -XX:+PrintGCTimeStamps -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=10 -XX:GCLogFileSize=1M -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=artifacts/heapdump	If an application must occupy more memory resources to process a large volume of data and the size of worker memory is greater than 2 GB, the G1 garbage collection algorithm is recommended. In this case, change the parameter value to <b>-Xms2G -Xmx5G -XX:+UseG1GC</b> .

# 26 Using Tez

---

## 26.1 Precautions

This section applies to MRS 3.x or later.

## 26.2 Common Tez Parameters

### Navigation path for setting parameters:

On Manager, choose **Cluster > Service > Tez > Configuration > All Configurations**. Enter a parameter name in the search box.

### Parameter description

**Table 26-1** Parameter description

Parameter	Description	Default Value
property.tez.log.dir	TezUI log directory	/var/log/Bigdata/tez/tezui
property.tez.log.level	TezUI log level	INFO

## 26.3 Accessing TezUI

Tez displays the Tez task execution process on a GUI. You can view the task execution details on the GUI.

### Prerequisite

The TimelineServer instance of the Yarn service has been installed.

## How to Use

Log in to Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). On Manager, choose **Cluster > Services > Tez**. Click the link on the right of **Tez WebUI** in the **Basic Information** area, and go to Tez web UI. You can view the details about Tez task execution.

## 26.4 Log Overview

### Log Description

**Log path:** The default save path of Tez logs is `/var/log/Bigdata/tez/role name`.

TezUI: `/var/log/Bigdata/tez/tezui` (run logs) and `/var/log/Bigdata/audit/tez/tezui` (audit logs)

**Log archive rule:** The automatic compression and archiving function of Tez is enabled. By default, when the size of a log file exceeds 20 MB (which is adjustable), the log file is automatically compressed. The naming rule of the compressed log file is as follows: `<Original log file name>-<yyyy-mm-dd_hh-mm-ss>.[/D].log.zip`. A maximum of 20 latest compressed files are retained. The number of compressed files and compression threshold can be configured.

**Table 26-2** Tez log list

Log Type	Name	Description
Run log	tezui.out	Log file that records TezUI running environment information
	tezui.log	Run log of the TezUI process
	tezui-omm-<Date>-gc.log.<No.>	GC log of the TezUI process
	prestartDetail.log	Work logs generated before the TezUI is started
	check-serviceDetail.log	Log file that records whether the TezUI service starts successfully
	postinstallDetail.log	Work logs after the TezUI is installed
	startDetail.log	Startup log of the TezUI process
	stopDetail.log	Stop log of the TezUI process
Audit log	tezui-audit.log	TezUI audit log

## Log Level

**Table 26-3** describes the log levels supported by TezUI.

Levels of run logs are ERROR, WARN, INFO, and DEBUG from the highest to the lowest priority. Run logs of equal or higher levels are recorded. The higher the specified log level, the fewer the logs recorded.

**Table 26-3** Log levels

Level	Description
ERROR	Logs of this level record error information about system running.
WARN	Exception information about the current event processing
INFO	Logs of this level record normal running status information about the system and events.
DEBUG	Logs of this level record the system information and system debugging information.

To modify log levels, perform the following operations:

- Step 1** Log in to Manager.
- Step 2** Choose **Cluster > Service > Tez > Configuration**.
- Step 3** Select **All Configurations**.
- Step 4** In the navigation pane, choose **TezUI > Log**.
- Step 5** Select a desired log level.
- Step 6** Click **Save**. In the dialog box that is displayed, click **OK** to save the configuration.
- Step 7** Click **Instance**, select the **TezUI** role, choose **More > Restart Instance**, enter the user password, and click **OK** in the dialog box that is displayed.
- Step 8** Wait until the instance is restarted for the configuration to take effect.

----End

## Log Format

The following table lists the Tez log formats.



**Table 26-4** Log formats

Log Type	Format	Example
Run log	<yyyy-MM-dd HH:mm:ss,SSS> <LogLevel> <Thread that generates the log> <Message in the log> <Location of the log event>	2020-07-31 11:44:21,378   INFO   TezUI-health-check   Start health check   com.XXX.tez.HealthCheck.run(HealthCheck.java:30)
Audit logs	<yyyy-MM-dd HH:mm:ss,SSS> <LogLevel> <Thread that generates the log> <UserName><User IP><Time><Operation><Resource><Result><Detail > <Location of the log event >	2018-12-24 12:16:25,319   INFO   HiveServer2-Handler-Pool: Thread-185   UserName=hive UserIP=10.153.2.204 Time=2018/12/24 12:16:25 Operation=CloseSession Result=SUCCESS Detail=   org.apache.hive.service.cli.thrift.ThriftCLIService.logAuditEvent(ThriftCLIService.java:434)

## 26.5 Common Issues

### 26.5.1 TezUI Cannot Display Tez Task Execution Details

#### Question

After a user logs in to Manager and switches to the Tez web UI, the submitted Tez tasks are not displayed.

#### Answer

The Tez task data displayed on the Tez WebUI requires the support of TimelineServer of Yarn. Ensure that TimelineServer has been enabled and is running properly before the task is submitted.

When setting the Hive execution engine to Tez, you need to set **yarn.timeline-service.enabled** to **true**. For details, see [Switching the Hive Execution Engine to Tez](#).

### 26.5.2 Error Occurs When a User Switches to the Tez Web UI

#### Question

When a user logs in to Manager and switches to the Tez web UI, error 404 or 503 is displayed.

## HTTP ERROR 404

Problem accessing /null/applicationhistory. Reason:

Not Found

Powered by Jetty:// 9.3.20.v20170531

❗ Adapter operation failed ⚠️ 503: Error accessing https://[redacted]:20026/Yarn/TimelineServer/57/ws/v1/timeline/TEZ\_DAG\_ID

### Answer

The Tez web UI depends on the TimelineServer instance of Yarn. Therefore, TimelineServer must be installed in advance and in the **Good** state.

## 26.5.3 Yarn Logs Cannot Be Viewed on the TezUI Page

### Question

A user logs in to the Tez web UI and clicks **Logs**, but the Yarn log page fails to be displayed and data cannot be loaded.



### This site can't be reached

10-244-224-251's server IP address could not be found.

Try running Windows Network Diagnostics.

DNS\_PROBE\_FINISHED\_NXDOMAIN

Reload

### Answer

Currently, the hostname is used for the access to the Yarn log page from the Tez web UI. Therefore, you need to configure the mapping between the hostname and IP address on the Windows host. Perform the following steps:

Modify the **C:\Windows\System32\drivers\etc\hosts** file on the Windows host and add a line indicating the mapping between the host name and IP address, for example, **10.244.224.45 10-044-224-45**. Save the modification and access the host again.

## 26.5.4 Table Data Is Empty on the TezUI HiveQueries Page

### Question

A user logs in to Manager and switches to the Tez web UI page, but no data for the submitted task is displayed on the **Hive Queries** page.

### Answer

To display task data on the **Hive Queries** page on the Tez web UI, you need to set the following parameters:

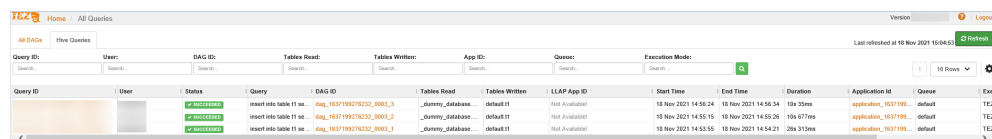
On Manager, choose **Cluster > Service > Hive > Configurations > All Configurations > HiveServer > Customization**. Add the following configuration to **hive-site.xml**:

Attribute	Attribute Value
hive.exec.pre.hooks	org.apache.hadoop.hive.ql.hooks.ATSHook
hive.exec.post.hooks	org.apache.hadoop.hive.ql.hooks.ATSHook
hive.exec.failure.hooks	org.apache.hadoop.hive.ql.hooks.ATSHook

### NOTE

Data display on TezUI depends on the TimelineServer instance of Yarn. If the TimelineServer instance is faulty or not started, you need to set **yarn.timeline-service.enabled** to **false** in **yarn-site.xml**. Otherwise, the Hive task fails to be executed.

After you configure the parameters and re-execute the Hive task, data can be displayed on the **Hive Queries** page. However, data of previous tasks cannot be displayed.



Query ID	User	Status	Query	DAG ID	Tables Read	Tables Written	LLAP App ID	Start Time	End Time	Duration	Application Id	Queue	Exit
		<span style="color: green;">✔</span> Succeeded	insert into table tt se...	dag_163719978232_8003_2	_dummy_database...	default11	Not Available!	18 Nov 2021 14:56:24	18 Nov 2021 14:56:34	10s 35ms	application_1637199...	default	TEZ
		<span style="color: green;">✔</span> Succeeded	insert into table tt se...	dag_163719978232_8003_2	_dummy_database...	default11	Not Available!	18 Nov 2021 14:55:15	18 Nov 2021 14:55:26	10s 877ms	application_1637199...	default	TEZ
		<span style="color: green;">✔</span> Succeeded	insert into table tt se...	dag_163719978232_8003_1	_dummy_database...	default11	Not Available!	18 Nov 2021 14:53:55	18 Nov 2021 14:54:01	26s 313ms	application_1637199...	default	TEZ

# 27 Using Yarn

## 27.1 Common Yarn Parameters

### Allocating Queue Resources

The Yarn service provides queues for users. Users allocate system resources to each queue. After the configuration is complete, you can click **Refresh Queue** or restart the Yarn service for the configuration to take effect.

#### Navigation path for setting parameters:

For versions earlier than MRS 3.x, perform the following operations:

On the MRS console, choose **Tenants > Resource Distribution Policies**.

The following uses the **default** queue as an example. The configurations of other queues are similar. Click **Modify** to edit the parameters.

**Table 27-1** Parameter description

Parameter	Description	Default Value
Capacity	Queue resource capacity (percentage). Ensure that the capacity requirement of each queue is satisfied when the system is busy. If only a few application programs are running in a queue, the remaining resource of the queue can be shared with other queues. Note that the total capacity of all queues must be smaller than 100.	20
Maximum Capacity	Maximum queue resource usage (percentage). Due to resource sharing, the resources used by a queue may exceed its capacity. The maximum resource usage can be limited using this parameter.	100

For MRS 3.x or later, perform the following operations:

On Manager, choose **Tenant Resources > Dynamic Resource Plan > Queue Configuration**.

The following uses the **default** tenant who modifies the Superior scheduler as an example. The configurations of other queues are similar. Click **Modify** to edit the parameters.

**Table 27-2** Queue configuration parameters

Parameter	Description
Max Master Shares(%)	Indicates the maximum percentage of resources occupied by all ApplicationMasters in the current queue.
Max Allocated vCores	Indicates the maximum number of cores that can be allocated to a single YARN container in the current queue. The default value is <b>-1</b> , indicating that the number of cores is not limited within the value range.
Max Allocated Memory(MB)	Indicates the maximum memory that can be allocated to a single YARN container in the current queue. The default value is <b>-1</b> , indicating that the memory is not limited within the value range.
Max Running Apps	Maximum number of tasks that can be executed at the same time in the current queue. The default value is <b>-1</b> , indicating that the number is not limited within the value range (the meaning is the same if the value is empty). The value <b>0</b> indicates that the task cannot be executed. The value ranges from <b>-1</b> to <b>2147483647</b> .
Max Running Apps per User	Maximum number of tasks that can be executed by each user in the current queue at the same time. The default value is <b>-1</b> , indicating that the number is not limited within the value range. If the value is <b>0</b> , the task cannot be executed. The value ranges from <b>-1</b> to <b>2147483647</b> .
Max Pending Apps	Maximum number of tasks that can be suspended at the same time in the current queue. The default value is <b>-1</b> , indicating that the number is not limited within the value range (the meaning is the same if the value is empty). The value <b>0</b> indicates that tasks cannot be suspended. The value ranges from <b>-1</b> to <b>2147483647</b> .
Resource Allocation Rule	Indicates the rule for allocating resources to different tasks of a user. The rule can be FIFO or FAIR. If a user submits multiple tasks in the current queue and the rule is FIFO, the tasks are executed one by one in sequential order; if the rule is FAIR, resources are evenly allocated to all tasks.

Parameter	Description
Default Resource Label	Indicates that tasks are executed on a node with a specified resource label.
Active	<ul style="list-style-type: none"><li>● <b>ACTIVE</b>: indicates that the current queue can receive and execute tasks.</li><li>● <b>INACTIVE</b>: indicates that the current queue can receive but cannot execute tasks. Tasks submitted to the queue are suspended.</li></ul>
Open	<ul style="list-style-type: none"><li>● <b>OPEN</b>: indicates that the current queue is opened.</li><li>● <b>CLOSED</b>: indicates that the current queue is closed. Tasks submitted to the queue are rejected.</li></ul>

## Displaying Container Logs on the Web UI

By default, the system collects container logs to HDFS. If you do not need to collect container logs to HDFS, configure the parameters in [Table 27-3](#). For details, see [Modifying Cluster Service Configuration Parameters](#).

**Table 27-3** Parameter description

Parameter	Description	Default Value
yarn.log-aggregation-enable	<p>Select whether to collect container logs to HDFS.</p> <ul style="list-style-type: none"> <li>If the parameter is set to <b>true</b>, container logs are collected to an HDFS directory. The default directory is <b>{yarn.nodemanager.remote-app-log-dir}/{user}/{thisParam}</b>. You can set the directory by setting the <b>yarn.nodemanager.remote-app-log-dir-suffix</b> parameter on the web UI.</li> <li>If this parameter is set to <b>false</b>, container logs will not be collected to HDFS.</li> </ul> <p>After changing the parameter value, restart the Yarn service for the setting to take effect.</p> <p><b>NOTE</b> The container logs that are generated before the parameter is set to <b>false</b> and the setting takes effect cannot be obtained from the web UI. You can obtain container logs from the directory specified by the <b>yarn.nodemanager.remote-app-log-dir-suffix</b> parameter before the setting takes effect.</p> <p>If you want to view the logs generated before on the web UI, you are advised to set this parameter to <b>true</b>.</p>	true

## Increasing the Number of Historical Jobs to Be Displayed on the web UI

By default, the Yarn web UI supports task list pagination. A maximum of 5,000 historical jobs can be displayed on each page, and a maximum of 10,000 historical jobs can be retained. If you need to view more jobs on the WebUI, configure parameters by referring to [Table 27-4](#). For details, see [Modifying Cluster Service Configuration Parameters](#).

**Table 27-4** Parameter description

Parameter	Description	Default Value
yarn.resourcemanager.max-completed-applications	Set the total number of historical jobs to be displayed on the web UI.	10000
yarn.resourcemanager.webapp.pagination.enable	Select whether to enable the job list background pagination function for the Yarn web UI.	true

Parameter	Description	Default Value
yarn.resourcemanager.webapp.pagination.threshold	Set the maximum number of jobs displayed on each page after the job list background pagination function of the Yarn web UI is enabled.	5000

 NOTE

- If a large number of historical jobs are displayed, the performance will be affected and the time for opening the Yarn web UI will be increased. Therefore, you are advised to enable the background pagination function and modify the **yarn.resourcemanager.max-completed-applications** parameter according to the actual hardware performance.
- After changing the parameter value, restart the Yarn service for the setting to take effect.

## 27.2 Creating Yarn Roles

### Scenario

This section describes how to create and configure a Yarn role. The Yarn role can be assigned with Yarn administrator permission and manage Yarn queue resources.

 NOTE

If the current component uses Ranger for permission control, you need to configure permission management policies based on Ranger. Refer to [Adding a Ranger Access Permission Policy for Yarn](#) for clusters of MRS 3.x or later.

### Prerequisites

- You have understood the service requirements.
- You have logged in to Manager.

### Procedure

For versions earlier than MRS 3.x, perform the following operations:

- Step 1** Choose **System > Manage Role > Create Role**.
- Step 2** Click **Create Role** and fill in **Role Name** and **Description**.
- Step 3** Set permissions. For details, see [Table 27-5](#).

Yarn permissions:

- **Cluster Admin Operations:** Yarn administrator permissions.
- **Scheduler Queue:** queue resources management .



**Table 27-5** Setting a role

Task	Operation
Setting the Yarn administrator permission	In the <b>Permission</b> table, click <b>Yarn</b> and select <b>Cluster Admin Operations</b> . <b>NOTE</b> The Yarn service needs to be restarted to set the Yarn administrator permission so that the saved role configuration can take effect.
Setting the permission for a user to submit tasks in a specified Yarn queue	1. In the <b>Permission</b> table, choose <b>Yarn &gt; Scheduler Queue</b> . 2. In the <b>Permission</b> column of the specified queue, select <b>Submit</b> .
Setting the permission for a user to manage tasks in a specified Yarn queue	1. In the <b>Permission</b> table, choose <b>Yarn &gt; Scheduler Queue</b> . 2. In the <b>Permission</b> column of the specified queue, select <b>Admin</b> .

If the Yarn role contains the **Submit** or **Manage** permission of a parent queue, the sub-queue inherits the permission by default, that is, the **Submit** or **Manage** permission is automatically added for the sub-queue. Permissions inherited by sub-queues will not be displayed as selected in the **Configure Resource Permission** table.

If you select only the **Submit** permission of a parent queue when setting the Yarn role, you need to manually specify the queue name when submitting tasks as a user with the permission of this role. Otherwise, when the parent queue has multiple sub-queues, the system does not automatically determine the queue to which the task is submitted and therefore submits the task to the **default** queue.

**Step 4** Click **OK**.

----End

For MRS 3.x or later, perform the following operations:

**Step 1** Choose System > Permission > Role.

**Step 2** Click **Create Role** and set a role name and enter description.

**Step 3** Refer [Table 27-6](#) to configure resource permissions for roles.

Yarn permissions:

- Cluster management: Yarn administrator permissions.
- Queue scheduling: queue resource management.

**Table 27-6** Setting a role

Task	Operation
Setting the Yarn administrator permission	In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> > <b>Yarn</b> > <b>Cluster Management</b> . <b>NOTE</b> The Yarn service needs to be restarted to set the Yarn administrator permission so that the saved role configuration can take effect.
Setting the permission for a user to submit tasks in a specified Yarn queue	1. In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> > <b>Yarn</b> > <b>Scheduling Queue</b> > <b>root</b> . 2. In the <b>Permission</b> column of the specified queue, select <b>Submit</b> .
Setting the permission for a user to manage tasks in a specified Yarn queue	1. In the <b>Configure Resource Permission</b> table, choose <i>Name of the desired cluster</i> > <b>Yarn</b> > <b>Scheduling Queue</b> > <b>root</b> . 2. In the <b>Permission</b> column of the specified queue, select <b>Manage</b> .

If the Yarn role contains the **Submit** or **Manage** permission of a parent queue, the sub-queue inherits the permission by default, that is, the **Submit** or **Manage** permission is automatically added for the sub-queue. Permissions inherited by sub-queues will not be displayed as selected in the **Configure Resource Permission** table.

If you select only the **Submit** permission of a parent queue when setting the Yarn role, you need to manually specify the queue name when submitting tasks as a user with the permission of this role. Otherwise, when the parent queue has multiple sub-queues, the system does not automatically determine the queue to which the task is submitted and therefore submits the task to the **default** queue.

**Step 4** Click **OK**.

----End

## 27.3 Using the Yarn Client

### Scenario

This section guides users to use a Yarn client in an O&M or service scenario.

### Prerequisites

- The client has been installed.  
For example, the installation directory is **/opt/hadoopclient**. The client directory in the following operations is only an example. Change it to the actual installation directory.

- Service component users are created by the MRS cluster administrator as required. In security mode, machine-machine users need to download the keytab file. A human-machine user must change the password upon the first login. In common mode, you do not need to download the keytab file or change the password.

## Using the Yarn Client

**Step 1** Log in to the node where the client is installed as the client installation user.

**Step 2** Run the following command to go to the client installation directory:

```
cd /opt/hadoopclient
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

**Step 4** If the cluster is in security mode, run the following command to authenticate the user. In normal mode, user authentication is not required.

```
kinit Component service user
```

**Step 5** Run the Yarn command. The following provides an example:

```
yarn application -list
```

```
----End
```

## Client-related FAQs

1. What Do I Do When the Yarn Client Exits Abnormally and Error Message "java.lang.OutOfMemoryError" Is Displayed After the Yarn Client Command Is Run?

This problem occurs because the memory required for running the Yarn client exceeds the upper limit (128 MB by default) set on the Yarn client. For clusters of MRS 3.x or later: You can modify **CLIENT\_GC\_OPTS** in *<Client installation path>/HDFS/component\_env* to change the memory upper limit of the Yarn client. For example, if you want to set the maximum memory to 1 GB, run the following command:

```
export CLIENT_GC_OPTS="-Xmx1G"
```

For clusters earlier than MRS 3.x: You can modify **GC\_OPTS\_YARN** in *<Client installation path >/HDFS/component\_env* to change the memory upper limit of the Yarn client. For example, if you want to set the maximum memory to 1 GB, run the following command:

```
export GC_OPTS_YARN="-Xmx1G"
```

After the modification, run the following command to make the modification take effect:

```
source <Client installation path>/bigdata_env
```

2. How Can I Set the Log Level When the Yarn Client Is Running?

By default, the logs generated during the running of the Yarn client are printed to the console. The default log level is INFO. To enable the DEBUG log level for fault locating, run the following command to export an environment variable:

```
export YARN_ROOT_LOGGER=DEBUG,console
```

Then run the Yarn Shell command to print DEBUG logs.

If you want to print INFO logs again, run the following command:

```
export YARN_ROOT_LOGGER=INFO,console
```

## 27.4 Configuring Resources for a NodeManager Role Instance

### Scenario

If the hardware resources (such as the number of CPU cores and memory size) of the nodes for deploying NodeManagers are different but the NodeManager available hardware resources are set to the same value, the resources may be wasted or the status may be abnormal. You need to change the hardware resource configuration for each NodeManager to ensure that the hardware resources can be fully utilized.

### Impact on the System

NodeManager role instances must be restarted for the new configuration to take effect, and the role instances are unavailable during restart.

### Prerequisites

- For versions earlier than MRS 3.x: You have logged in to the MRS management console.
- Clusters of MRS 3.x or later: You have logged in to Manager.

### Procedure

For versions earlier than MRS 3.x, perform the following operations:

- Step 1** Choose **Clusters > Active Clusters**, and click a cluster name. Choose **Components > Yarn > Instances**.
- Step 2** Click **NodeManager** in the **Role** column and go to the **Instance Configuration** tab page. Select **All** from the **Basic** drop-down list, and search for the required parameters.
- Step 3** Enter **yarn.nodemanager.resource.cpu-vcores** in the search box, and set the number of vCPUs that can be used by NodeManager on the current node. You are advised to set this parameter to 1.5 to 2 times the number of actual logical CPUs on the node. Enter **yarn.nodemanager.resource.memory-mb** in the search box, and set the physical memory size that can be used by NodeManager on the current node. You are advised to set this parameter to 75% to 90% of the actual physical memory size of the node.

 NOTE

Enter **yarn.scheduler.maximum-allocation-vcores** in the search box, and set the maximum number of available CPUs in a container. Enter **yarn.scheduler.maximum-allocation-mb** in the search box, and set the maximum available memory of a container. The instance level cannot be changed. The parameter values need to be changed in the configuration of the Yarn service, and the Yarn service needs to be restarted for the changes to take effect.

**Step 4** Click **Save Configuration**, select **Restart the affected services or instances**, and click **OK** to restart the NodeManager role instance.

**Operation succeeded** is displayed. Click **Finish**. The NodeManager role instance is started successfully.

----End

For MRS 3.x or later, perform the following operations:

**Step 1** Choose **Cluster > Name of the desired cluster > Services > Yarn > Instance**.

**Step 2** Click the role instance name corresponding to the node where NodeManager is deployed, switch to **Instance Configuration**, and select **All Configurations**.

**Step 3** Enter **yarn.nodemanager.resource.cpu-vcores** in the search box, and set the number of vCPUs that can be used by NodeManager on the current node. You are advised to set this parameter to 1.5 to 2 times the number of actual logical CPUs on the node. Enter **yarn.nodemanager.resource.memory-mb** in the search box, and set the physical memory size that can be used by NodeManager on the current node. You are advised to set this parameter to 75% of the actual physical memory size of the node.

 NOTE

Enter **yarn.scheduler.maximum-allocation-vcores** in the search box, and set the maximum number of available CPUs in a container. Enter **yarn.scheduler.maximum-allocation-mb** in the search box, and set the maximum available memory of a container. The instance level cannot be changed. The parameter values need to be changed in the configuration of the Yarn service, and the Yarn service needs to be restarted for the changes to take effect.

**Step 4** Click **Save**, and then click **OK**. to restart the NodeManager role instance.

A message is displayed, indicating that the operation is successful. Click **Finish**. The NodeManager role instance is started successfully.

----End

## 27.5 Changing NodeManager Storage Directories

### Scenario

If the storage directories defined by the Yarn NodeManager are incorrect or the Yarn storage plan changes, the MRS cluster administrator needs to modify the NodeManager storage directories on Manager to ensure that the Yarn works properly. The storage directories of NodeManager include the local storage directory **yarn.nodemanager.local-dirs** and log directory **yarn.nodemanager.log-dirs**. Changing the ZooKeeper storage directory includes the following scenarios:

- Change the storage directory of the NodeManager role. In this way, the storage directories of all NodeManager instances are changed.
- Change the storage directory of a single NodeManager instance. In this way, only the storage directory of this instance is changed, and the storage directories of other instances remain the same.

## Impact on the System

- The cluster needs to be stopped and restarted during the process of changing the storage directory of the NodeManager role, and the cluster cannot provide services before started.
- The NodeManager instance needs to be stopped and restarted during the process of changing the storage directory of the instance, and the instance at this node cannot provide services before it is started.
- The directory for storing service parameter configurations must also be updated.
- After the storage directories of NodeManager are changed, you need to download and install the client again.

## Prerequisites

- New disks have been prepared and installed on each data node, and the disks are formatted.
- New directories have been planned for storing data in the original directories.
- The user **admin** has been prepared.

## Procedure

For versions earlier than MRS 3.x, perform the following operations:

### Step 1 Check the environment.

1. Log in to the MRS console. In the left navigation pane, choose **Clusters > Active Clusters**, and click a cluster name. Choose **Components** and check whether health status of Yarn is **Good**.
  - If yes, go to [Step 1.3](#).
  - If no, the Yarn status is unhealthy. Go to [Step 1.2](#).
2. Rectify the Yarn fault. No further action is required.
3. Determine whether to change the storage directory of the NodeManager role or that of a single NodeManager instance:
  - To change the storage directory of the NodeManager role, go to [Step 2](#).
  - To change the storage directory of a single NodeManager instance, go to [Step 3](#).

### Step 2 Change the storage directory of the NodeManager role.

1. Choose **Clusters > Active Clusters**, and click a cluster name. Choose **Components > Yarn > Stop** to stop the Yarn service.
2. Log in to the ECS server and go to each node where Yarn is installed as user **root**. Perform the following operations:

- a. Create a target directory.  
For example, to create the target directory `${BIGDATA_DATA_HOME}/data2`, run the following command:  
**mkdir `${BIGDATA_DATA_HOME}/data2`**
  - b. Mount the target directory to the new disk.  
For example, mount `${BIGDATA_DATA_HOME}/data2` to the new disk.
  - c. Modify permissions on the new directory.  
For example, to modify permissions on the `${BIGDATA_DATA_HOME}/data2` directory, run the following commands:  
**chmod 750 `${BIGDATA_DATA_HOME}/data2` -R and chown omm:wheel `${BIGDATA_DATA_HOME}/data2` -R**
3. On the MRS console, choose **Clusters > Active Clusters** and click a cluster name. Choose **Components > Yarn > Instances**. Select the NodeManager instance of the corresponding host. Choose **Instance Configuration > All Configurations**.  
Change the value of `yarn.nodemanager.local-dirs` or `yarn.nodemanager.log-dirs` to the new target directory.  
For example, change the value of `yarn.nodemanager.local-dirs` or `yarn.nodemanager.log-dirs` to `/srv/BigData/data2/nm/containerlogs`.
  4. Click **Save Configuration**, select **Restart the affected services or instances**, and click **OK** Restart the Yarn service.  
Click **Finish** when the system displays "Operation successful". Yarn is successfully started. No further action is required.

**Step 3** Change the storage directory of a single NodeManager instance.

1. Choose **Clusters > Active Clusters**, and click a cluster name. Choose **Components > Yarn > Instances**. Select the NodeManager instance whose storage directory needs to be modified, and choose **More > Stop Instance**.
2. Log in to the ECS and go to the NodeManager node as user `root`. Perform the following operations:
  - a. Create a target directory.  
For example, to create the target directory `${BIGDATA_DATA_HOME}/data2`, run the following command:  
**mkdir `${BIGDATA_DATA_HOME}/data2`**
  - b. Mount the target directory to the new disk.  
For example, mount `${BIGDATA_DATA_HOME}/data2` to the new disk.
  - c. Modify permissions on the new directory.  
For example, to modify permissions on the `${BIGDATA_DATA_HOME}/data2` directory, run the following commands:  
**chmod 750 `${BIGDATA_DATA_HOME}/data2` -R and chown omm:wheel `${BIGDATA_DATA_HOME}/data2` -R**
3. On the MRS console, click the specified NodeManager instance and switch to the **Instance Configuration** tab page.  
Change the value of `yarn.nodemanager.local-dirs` or `yarn.nodemanager.log-dirs` to the new target directory.

For example, change the value of `yarn.nodemanager.local-dirs` or `yarn.nodemanager.log-dirs` to `/srv/BigData/data2/nm/containerlogs`.

4. Click **Save Configuration** and select **Restart the affected services or instances**. Click **OK** to restart the NodeManager instance.

Click **Finish** when the system displays "Operation successful". The NodeManager instance is successfully started.

----End

For MRS 3.x or later, perform the following operations:

#### Step 1 Check the environment.

1. Log in to Manager, choose **Cluster** > *Name of the desired cluster* > **Service** to check whether **Running Status** of Yarn is **Normal**.
  - If yes, go to **1.c**.
  - If no, the Yarn status is unhealthy. In this case, go to **1.b**.
2. Rectify faults of Yarn. No further action is required.
3. Determine whether to change the storage directory of the NodeManager role or that of a single NodeManager instance:
  - To change the storage directory of the NodeManager role, go to **2**.
  - To change the storage directory of a single NodeManager instance, go to **3**.

#### Step 2 Change the storage directory of the NodeManager role.

1. Choose **Cluster** > *Name of the desired cluster* > **Service** > **Yarn** > **Stop** to stop the Yarn service.
2. Log in to each data node where the Yarn service is installed as user **root** and perform the following operations:
  - a. Create a target directory.  
For example, to create the target directory `/${BIGDATA_DATA_HOME}/data2`, run the following command:  
**mkdir `/${BIGDATA_DATA_HOME}/data2`**
  - b. Mount the target directory to the new disk.  
For example, mount `/${BIGDATA_DATA_HOME}/data2` to the new disk.
  - c. Modify permissions on the new directory.  
For example, to modify permissions on the `/${BIGDATA_DATA_HOME}/data2` directory, run the following commands:  
**chmod 750 `/${BIGDATA_DATA_HOME}/data2` -R** and **chown omm:wheel `/${BIGDATA_DATA_HOME}/data2` -R**
3. On the Manager portal, choose **Cluster** > *Name of the desired cluster* > **Services** > **Yarn** > **Instance**. Select the NodeManager instance of the corresponding host, click **Instance Configuration**, and select **All Configurations**.

Change the value of `yarn.nodemanager.local-dirs` or `yarn.nodemanager.log-dirs` to the new target directory.

For example, change the value of `yarn.nodemanager.local-dirs` or `yarn.nodemanager.log-dirs` to `/srv/BigData/data2/nm/containerlogs`.



4. Click **Save**, and then click **OK**. Restart the Yarn service.  
Click **Finish** when the system displays "Operation successful". Yarn is successfully started. No further action is required.

**Step 3** Change the storage directory of a single NodeManager instance.

1. Choose **Cluster** > *Name of the desired cluster* > **Service** > **Yarn** > **Instance**, select the NodeManager instance whose storage directory needs to be modified, and choose **More** > **Stop**.
2. Log in to the NodeManager node as user **root**, and perform the following operations:
  - a. Create a target directory.  
For example, to create the target directory `${BIGDATA_DATA_HOME}/data2`, run the following command:  
**mkdir `${BIGDATA_DATA_HOME}/data2`**
  - b. Mount the target directory to the new disk.  
For example, mount `${BIGDATA_DATA_HOME}/data2` to the new disk.
  - c. Modify permissions on the new directory.  
For example, to modify permissions on the `${BIGDATA_DATA_HOME}/data2` directory, run the following commands:  
**chmod 750 `${BIGDATA_DATA_HOME}/data2` -R** and **chown omm:wheel `${BIGDATA_DATA_HOME}/data2` -R**
3. On Manager, click the specified NodeManager instance, and switch to the **Instance Configuration** page.  
Change the value of **yarn.nodemanager.local-dirs** or **yarn.nodemanager.log-dirs** to the new target directory.  
For example, change the value of **yarn.nodemanager.local-dirs** or **yarn.nodemanager.log-dirs** to `/srv/BigData/data2/nm/containerlogs`.
4. Click **Save**, and then click **OK** to restart the NodeManager instance.  
Click **Finish** when the system displays "Operation successful". The NodeManager instance is successfully started.

----End

## 27.6 Configuring Strict Permission Control for Yarn

### Scenario

In the multi-tenant scenario in security mode, a cluster can be used by multiple users, and tasks of multiple users can be submitted and executed. Users are invisible to each other. A permission control mechanism is required to prevent task information of users from being obtained by other users.

For example, if user B logs in to the system and views the application list when the application submitted by user A is running, user B should not be able to view the application information of user A.

## Configuration Description

- Viewing Yarn configuration parameters

Go to the **All Configurations** page of Yarn and enter a parameter name list in [Table 27-7](#) in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

**Table 27-7** Parameter description

Parameter	Description	Default Value
yarn.acl.enable	Whether to enable Yarn permission control	true
yarn.webapp.filter-entity-list-by-user	Whether to enable the strict view function. After this function is enabled, a login user can view only the content that the user has the permission to view. To enable this function, set <b>yarn.acl.enable</b> to <b>true</b> . <b>NOTE</b> This parameter applies to clusters of MRS 3.x or later.	true

- Viewing MapReduce configuration parameters

Go to the **All Configurations** page of MapReduce and enter a parameter name in [Table 27-8](#) in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

**Table 27-8** Parameter description

Parameter	Description	Default Value
mapreduce.cluster.acls.enabled	Whether to enable permission control of MapReduce JobHistoryServer This parameter is a client parameter and takes effect after permission control is enabled on the JobHistoryServer server.	true

Parameter	Description	Default Value
yarn.webapp.filter-entity-list-by-user	<p>Whether to enable the strict view of MapReduce JobHistoryServer. After the strict view is enabled, a login user can view only the content that the user has the permission to view. This parameter is a server parameter of JobHistoryServer. It indicates that permission control is enabled for JHS. However, whether to control a specific application is determined by the client parameter <b>mapreduce.cluster.acls.enabled</b>.</p> <p><b>NOTE</b> This parameter applies to clusters of MRS 3.x or later.</p>	true

#### NOTICE

The preceding configurations affect the RESTful API and Shell command results. After the preceding configurations are enabled, the return results of RESTful API calls and shell commands contain only the information that the user has the permission to view.

If **yarn.acl.enable** or **mapreduce.cluster.acls.enabled** is set to **false**, the Yarn or MapReduce permission verification function is disabled. In this case, any user can submit tasks and view task information on Yarn or MapReduce, which poses security risks. Exercise caution when performing this operation.

## 27.7 Configuring Container Log Aggregation

### Scenario

Yarn provides the container log aggregation function to collect logs generated by containers on each node to HDFS to release local disk space. You can collect logs in either of the following ways:

- After the application is complete, collect container logs to HDFS at a time.
- During application running, periodically collect log segments generated by containers and save them to HDFS.

### Configuration Description

Navigation path for setting parameters:

Go to the **All Configurations** page of Yarn and enter a parameter name list in [Table 27-9](#) in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

The **yarn.nodemanager.remote-app-log-dir-suffix** parameter must be configured on the Yarn client. The configurations on the ResourceManager, NodeManager, and JobHistory nodes must be the same as those on the Yarn client.

The periodic log collection function applies only to MapReduce applications, for which rolling output of log files must be configured. [Table 27-11](#) describes the configurations in the **mapred-site.xml** configuration file on the MapReduce client node.

**Table 27-9** Parameter description

Parameter	Description	Default Value
yarn.log-aggregation-enable	<p>Whether to enable container log aggregation</p> <ul style="list-style-type: none"> <li>• If this parameter is set to <b>true</b>, logs are collected to the HDFS directory.</li> <li>• If this parameter is set to <b>false</b>, the function is disabled, and logs are not collected to HDFS.</li> </ul> <p>After changing the parameter value, restart the Yarn service for the setting to take effect.</p> <p><b>NOTE</b></p> <ul style="list-style-type: none"> <li>• The container logs that are generated before the parameter is set to <b>false</b> and the setting takes effect cannot be obtained from the web UI.</li> <li>• If you want to view the logs generated before on the web UI, you are advised to set this parameter to <b>true</b>.</li> </ul>	true

Parameter	Description	Default Value
yarn.nodemanager.log-aggregation.roll-monitoring-interval-seconds	<p>Interval for NodeManager to periodically collect logs</p> <ul style="list-style-type: none"> <li>• If this parameter is set to <b>-1</b> or <b>0</b>, periodic log collection is disabled. Logs are collected at a time after application running is complete.</li> <li>• The minimum collection interval can be set to 3,600 seconds. If this parameter is set to a value greater than 0 and less than 3,600, the collection interval is 3,600 seconds.</li> </ul> <p>Interval for NodeManager to wake up and upload logs. If this parameter is set to <b>-1</b> or <b>0</b>, rolling monitoring is disabled and logs are aggregated when the application task is complete. The value must be greater than or equal to <b>-1</b>.</p>	-1

Parameter	Description	Default Value
<code>yarn.nodemanager.disk-health-checker.log-dirs.max-disk-utilization-per-disk-percentage</code>	<p>Maximum percentage of the Yarn disk quota that can be occupied by the container log directory on each disk. When the space occupied by the log directory exceeds the value of this parameter, the periodic log collection service is triggered to start a log collection activity beyond the period to release the local disk space. Maximum space for container logs that can be provided on each disk. If the disk space occupied by container logs exceeds this threshold, data aggregation in rolling mode is triggered.</p> <ul style="list-style-type: none"> <li>For clusters of versions earlier than MRS 3.x: The valid value range of the maximum disk quota percentage is 0 to 100. If the value is less than or equal to <b>0</b>, it is forcibly reset to <b>25</b>. If the value is greater than 100, the value is forcibly reset to <b>25</b>.</li> <li>For clusters of MRS 3.x or later: The valid value range of the maximum disk quota percentage is -1 to 100. If the value is less than -1, it is forcibly reset to <b>25</b>. If the value is greater than 100, the value is forcibly reset to <b>25</b>. If you set the value to <b>-1</b>, the disk capacity detection function for Container log directory is disabled.</li> </ul> <p><b>NOTE</b></p> <ul style="list-style-type: none"> <li>Percentage of the available disk space of the container log directory = Percentage of the available disk space of Yarn (<b>yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage</b>) x Percentage of the available disk space of the container log directory (<b>yarn.nodemanager.disk-health-checker.log-dirs.max-disk-utilization-per-disk-percentage</b>)</li> <li>Only applications with the periodic log collection function enabled can trigger log collection when the disk quota of the log directory exceeds the threshold.</li> </ul>	25

Parameter	Description	Default Value
yarn.nodemanager.remote-app-log-dir-suffix	Name of the HDFS folder in which container logs are to be stored. This parameter and <b>yarn.nodemanager.remote-app-log-dir</b> form the full path for storing container logs. That is, <b>{yarn.nodemanager.remote-app-log-dir}/{user}/{yarn.nodemanager.remote-app-log-dir-suffix}</b> . <b>NOTE</b> <i>{user}</i> indicates the username for running the task.	logs
yarn.nodemanager.log-aggregator.on-fail.retain-log-in-sec	Duration for retaining container logs on the local host after the logs fail to be collected, in second <ul style="list-style-type: none"> <li>• If this parameter is set to <b>0</b>, local logs are deleted immediately.</li> <li>• If this parameter is set to a positive number, local logs are retained for this period.</li> </ul>	604800

Go to the **All Configurations** page of MapReduce and enter a parameter name in [Table 27-10](#) in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

**Table 27-10** Parameter description

Parameter	Description	Default Value
yarn.log-aggregation.retain-seconds	Duration for retaining aggregated logs, in second <ul style="list-style-type: none"> <li>• If this parameter is set to <b>-1</b>, the container logs will be retained permanently in the HDFS.</li> <li>• If this parameter is set to <b>0</b> or a positive integer, container logs will be stored for such a period and deleted after the period expires.</li> </ul> <b>NOTE</b> A short period may increase load of the NameNode. Therefore, you are advised to set this parameter to a proper value.	1296000

Parameter	Description	Default Value
yarn.log-aggregation.retain-check-interval-seconds	<p>Interval for storing container logs in HDFS, in second</p> <ul style="list-style-type: none"> <li>If this parameter is set to <b>-1</b> or <b>0</b>, the interval will be one tenth of the period specified by <b>yarn.log-aggregation.retain-seconds</b>.</li> </ul> <p><b>NOTE</b> If this parameter is set to <b>-1</b> or <b>0</b>, <b>yarn.log-aggregation.retain-seconds</b> cannot be set to <b>0</b>.</p> <ul style="list-style-type: none"> <li>If this parameter is set to a positive number, container logs in HDFS will be scanned at such an interval.</li> </ul> <p><b>NOTE</b> A short interval may increase load of the NameNode. Therefore, you are advised to set this parameter to a proper value.</p>	86400

Go to the **All Configurations** page of Yarn and enter a parameter name list in [Table 27-11](#) in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

**Table 27-11** Configuring rolling output of MapReduce application log files

Parameter	Description	Default Value
mapreduce.task.userlog.limit.kb	Maximum size of a single task log file of the MapReduce application. When the maximum size of the log file has been reached, a new log file is generated. The value <b>0</b> indicates that the size of the log file is not limited.	51200



Parameter	Description	Default Value
yarn.app.mapreduce.task.container.log.backups	<p>Maximum number of task logs that can be retained for the MapReduce application. If this parameter is set to <b>0</b>, rolling output is disabled.</p> <p>Number of task log backup files when ContainerRollingLogAppender (CRLA) is used. By default, ContainerLogAppender (CLA) is used and container logs are not rolled back.</p> <p>When both <b>mapreduce.task.userlog.limit.kb</b> and <b>yarn.app.mapreduce.task.container.log.backups</b> are greater than 0, CRLA is enabled. The value ranges from 0 to 999.</p>	10
yarn.app.mapreduce.am.container.log.limit.kb	<p>Maximum size of a single ApplicationMaster log file of the MapReduce application, in KB. When the maximum size of the log file has been reached, a new log file is generated. The value <b>0</b> indicates that the size of a single ApplicationMaster log file is not limited.</p>	51200
yarn.app.mapreduce.am.container.log.backups	<p>Maximum number of ApplicationMaster logs that can be retained for the MapReduce application. If this parameter is set to <b>0</b>, rolling output is disabled. Number of ApplicationMaster log backup files when CRLA is used. By default, CLA is used and container logs are not rolled back.</p> <p>When both <b>yarn.app.mapreduce.am.container.log.limit.kb</b> and <b>yarn.app.mapreduce.am.container.log.backups</b> are greater than 0, CRLA is enabled for the ApplicationMaster. The value ranges from 0 to 999.</p>	20
yarn.app.mapreduce.shuffle.log.backups	<p>Maximum number of shuffle logs that can be retained for the MapReduce application. If this parameter is set to <b>0</b>, rolling output is disabled.</p> <p>When both <b>yarn.app.mapreduce.shuffle.log.limit.kb</b> and <b>yarn.app.mapreduce.shuffle.log.backups</b> are greater than 0, <b>syslog.shuffle</b> uses CRLA. The value ranges from 0 to 999.</p>	10

Parameter	Description	Default Value
yarn.app.mapreduce.shuffle.log.limit.kb	Maximum size of a single shuffle log file of the MapReduce application, in KB. When the maximum size of the log file has been reached, a new log file is generated. If this parameter is set to <b>0</b> , the size of a single shuffle log file is not limited. The value must be greater than or equal to <b>0</b> .	51200

## 27.8 Using CGroups with YARN

This section applies to clusters of MRS 3.x or later.

### Scenario

CGroups is a Linux kernel feature. In YARN this feature allows containers to be limited in their resource usage (example, CPU usage). Without CGroups, it is hard to limit the container CPU usage. Without CGroups, it is hard to limit the container CPU usage.

#### NOTE

Currently, CGroups is only used for limiting the CPU usage.

### Configuration Description

For details about how to configure the CGroups function for CPU isolation and security, see the Hadoop official website: <http://hadoop.apache.org/docs/r3.1.1/hadoop-yarn/hadoop-yarn-site/NodeManagerCgroups.html>

CGroups is a Linux kernel feature and is enabled by using LinuxContainerExecutor. For details about how to configure the LinuxContainerExecutor for security, see the official website. You can learn the file system permissions assigned for users and user groups from documentation published on the official website. For details, see <http://hadoop.apache.org/docs/r3.1.1/hadoop-project-dist/hadoop-common/SecureMode.html#LinuxContainerExecutor>.

#### NOTE

- Do not modify users, user groups, and related permissions of various paths in the corresponding file system. Otherwise, functions of CGroups may become abnormal.
- If the parameter value of **yarn.nodemanager.resource.percentage-physical-cpu-limit** is too small, the number of available cores may be less than one. For example, if the parameter of a four-core node is set to 20%, the number available core is less than one. As a result, all cores will be used. The Quota mode can be used in Linux versions, for example, Cent OS, that do not support Quota mode.

The table below describes the parameter for configuring cpuset mode, that is, only configured CPUs can be used by YARN.

**Table 27-12** Parameter description

Parameter	Description	Default Value
yarn.nodemanager.linux-container-executor.cgroups.cpu-set-usage	Whether to enable the cpuset mode. If this parameter is set to <b>true</b> , the cpuset mode is enabled.	false

The table below describes the parameters for configuring the strictcpuset mode, that is, only configured CPUs can be used by containers.

**Table 27-13** Parameter description

Parameter	Description	Default Value
yarn.nodemanager.linux-container-executor.cgroups.cpu-set-usage	Whether to enable the cpuset mode. If this parameter is set to <b>true</b> , the cpuset mode is enabled.	false
yarn.nodemanager.linux-container-executor.cgroups.cpuset.strict.enabled	Whether containers use allocated CPUs. If this parameter is set to <b>true</b> , the container can use the allocated CPUs.	false

To switch from cpuset mode to quota mode, the following conditions must be met:

- Set the **yarn.nodemanager.linux-container-executor.cgroups.cpu-set-usage** parameter to **false**.
- Delete container folders if exists.
- Delete all the CUPs configured in the **cpuset.cpus** file.

## Procedure

**Step 1** Log in to Manager. Choose **Cluster** > *Name of the desired cluster* > **Services** > **Yarn** > **Configurations** and select **All Configurations**.

**Step 2** In the navigation pane on the left, choose **NodeManager** > **Customization** and find the **yarn-site.xml** file.

**Step 3** Add the parameters in [Table 27-12](#) and [Table 27-13](#) as user-defined parameters.

Based on the configuration files and parameter functions, locate the row where parameter **yarn-site.xml** resides. Enter the parameter name in the **Name** column and enter the parameter value in the **Value** column.

Click + to add a customized parameter.

**Step 4** Click **Save**. In the displayed **Save Configuration** dialog box, confirm the modification and click **OK**. Click **Finish** when the system displays "Operation succeeded". The configuration is successfully saved.

After the configuration is saved, restart the Yarn service whose configuration has expired for the configuration to take effect.

----End

## 27.9 Configuring the Number of ApplicationMaster Retries

### Scenario

When resources are insufficient or ApplicationMaster fails to start, a client probably encounters running errors.

### Configuration Description

Go to the **All Configurations** page of Yarn and enter a parameter name list in [Table 27-14](#) in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

**Table 27-14** Parameter description

Parameter	Description	Default Value
yarn.resource manager.am.max-attempts	Number of retries of the ApplicationMaster. Increasing the number of retries can prevent ApplicationMaster startup failures caused by insufficient resources. This applies to global settings of all ApplicationMasters. Each ApplicationMaster can use an API to set an independent maximum number of retries. However, the number of retries cannot be greater than the global maximum number of retries. If the value is greater than the global maximum number of retries, the ResourceManager overwrites the value to allow at least one retry. The value must be greater than or equal to 1.	2

## 27.10 Configure the ApplicationMaster to Automatically Adjust the Allocated Memory

This section applies to clusters of MRS 3.x or later.

## Scenario

During the process of starting the configuration, when the ApplicationMaster creates a container, the allocated memory is automatically adjusted according to the total number of tasks, which makes resource utilization more flexible and improves the fault tolerance of the client application.

## Configuration Description

### Navigation path for setting parameters:

On Manager, choose **Cluster** > *Name of the desired cluster* > **Service** > **Yarn** > **Configuration**. On the displayed page, select **All Configurations** and enter **mapreduce.job.am.memory.policy**.

### Configuration description

If the default value of the parameter is left empty. In this case, the automatic adjustment policy is not enabled. The memory of ApplicationMaster is still affected by the value of **yarn.app.mapreduce.am.resource.mb**.

The value of **mapreduce.job.am.memory.policy** consists of five items, and they are separated by colons (:) and commas (,) in the following format: **baseTaskCount:taskStep:memoryStep,minMemory:maxMemory**. The format is strictly checked when the value is entered.

**Table 27-15** Parameter description

Parameter	Description	Setting Requirement
baseTaskCount	Indicates the total number of tasks. The configuration of ApplicationMaster is valid only when the total number of tasks (on the sum of the Map and Reduce ends) is greater than or equal to the value of this parameter.	The value cannot be empty and must be greater than 0.
taskStep	Indicates the incremental step length of tasks. This parameter and <b>memoryStep</b> determine the memory adjustment amount.	The value cannot be empty and must be greater than 0.
memoryStep	Indicates the incremental memory step. The memory capacity is increased based on the value of <b>yarn.app.mapreduce.am.resource.mb</b> .	The value cannot be empty and must be greater than 0. The unit is MB.
minMemory	Indicates the lower limit of the memory that can be automatically adjusted. If the memory after the automatic adjustment is less than or equal to the value of this parameter, the value of <b>yarn.app.mapreduce.am.resource.mb</b> is used.	The value cannot be empty. It must be greater than 0 and cannot be greater than the value of <b>maxMemory</b> . Unit: MB

Parameter	Description	Setting Requirement
maxMemory	Indicates the upper limit of memory that can be automatically adjusted. If the adjusted memory exceeds the upper limit, use this value as the final value.	The value cannot be empty. It must be greater than 0 and cannot be less than the value of <b>minMemory</b> . Unit: MB

## Example Value

Configuration:

- yarn.app.mapreduce.am.resource.mb=1536
- mapreduce.job.am.memory.policy=100:10:50,1200:2000
- Total number of tasks of an application =120

The calculation process is as follows:

Memory after adjustment =  $1536 + [(120 - 100)/10] \times 50 = 1636$ . In this example, memory after adjustment 1636 is greater than the value of **minMemory 1200**, and less than the value of **maxMemory 2000**. Therefore, the ApplicationMaster memory is set to **1636 MB**.

If the value of **memStep** is changed to **250**, the calculation formula is as follows: Memory after adjustment =  $1536 + [(120 - 100) / 10] \times 250 = 2136$ . In this case, the memory after adjustment is greater than the value of **maxMemory 2000**. As a result, the value of **ApplicationMaster** is set to **2000 MB**.

### NOTE

If the memory after adjustment is lower than the value of **minMemory**, the configuration does not take effect but the value is still printed on the backend server. This value is provided as the reference for adjusting the value of **minMemory**.

## 27.11 Configuring the Access Channel Protocol

### Scenario

The value of the **yarn.http.policy** parameter must be consistent on both the server and clients. Web UIs on clients will be garbled if an inconsistency exists, for example, the parameter value is **HTTPS\_ONLY** on the server but it is left unspecified on a client (the parameter value **HTTP\_ONLY** is applied to the client by default). Set the **yarn.http.policy** parameters on the clients and server to prevent garbled characters from being displayed on the clients.

## Procedure

- Step 1** On Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Yarn** > **Configurations**. On the displayed page, select **All Configurations** and enter **yarn.http.policy**.
- In security mode, set this parameter to **HTTPS\_ONLY**.
  - In normal mode, set this parameter to **HTTP\_ONLY**.
- Step 2** Log in to the node where the client is installed as the client installation user.
- Step 3** Run the following command to switch to the client installation directory:
- ```
cd /opt/client
```
- Step 4** Run the following command to edit the **yarn-site.xml** file:
- ```
vi Yarn/config/yarn-site.xml
```
- Change the value of **yarn.http.policy**.
- In security mode, set this parameter to **HTTPS\_ONLY**.
- In normal mode, set this parameter to **HTTP\_ONLY**.
- Step 5** Run the **:wq** command to save execution.
- Step 6** Restart the client for the settings to take effect.
- End

## 27.12 Configuring Memory Usage Detection

### Scenario

If memory usage of the submitted application cannot be estimated, you can modify the configuration on the server to determine whether to check the memory usage.

If the memory usage is not checked, the container occupies the memory until the memory overflows. If the memory usage exceeds the configured memory size, the corresponding container is killed.

### Configuration Description

Go to the **All Configurations** page of Yarn and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

**Table 27-16** Parameter description

Parameter	Description	Default Value
yarn.nodemanager.vmem-check-enabled	<p>Whether to enable virtual memory usage detection. If the memory used by a task exceeds the allocated memory size, the task is forcibly stopped.</p> <ul style="list-style-type: none"> <li>• If the value is <b>true</b>, the virtual memory will be checked.</li> <li>• If the value is <b>false</b>, the virtual memory will not be checked.</li> </ul>	<p>For versions earlier than MRS 3.x: false</p> <p>For MRS 3.x or later: true</p>
yarn.nodemanager.pmem-check-enabled	<p>Whether to enable physical memory usage detection. If the memory used by a task exceeds the allocated memory size, the task is forcibly stopped.</p> <ul style="list-style-type: none"> <li>• If the value is <b>true</b>, the physical memory will be checked.</li> <li>• If the value is <b>false</b>, the physical memory will not be checked.</li> </ul>	true

## 27.13 Configuring the Additional Scheduler WebUI

### Scenario

If the custom scheduler is set in ResourceManager, you can set the corresponding web page and other Web applications for the custom scheduler.

### Configuration Description

Go to the **All Configurations** page of Yarn and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

**Table 27-17** Configuring the Additional Scheduler WebUI

Parameter	Description	Default Value
hadoop.http.rmwebapp.scheduler.page.classes	<p>Load the corresponding web page for the custom scheduler on the RM WebUI. This parameter is valid only when</p> <p><b>yarn.resourcemanager.scheduler.class</b> is set to a custom scheduler.</p>	-
yarn.http.rmwebapp.external.classes	<p>Load the custom web application in the RM Web service.</p>	-



## 27.14 Configuring Yarn Restart

### Scenario

The Yarn Restart feature includes ResourceManager Restart and NodeManager Restart.

- When ResourceManager Restart is enabled, the new active ResourceManager node loads the information of the previous active ResourceManager node, and takes over container status information on all NodeManager nodes to continue service running. In this way, status information can be saved by periodically executing checkpoint operations, avoiding data loss.
- When NodeManager Restart is enabled, NodeManager locally saves information about containers running on the node. After NodeManager is restarted, the container running progress on the node will not be lost by restoring the saved status information.

### Configuration Description

Go to the **All Configurations** page of Yarn and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Configure ResourceManager Restart as follows:

**Table 27-18** Parameter description of ResourceManager Restart

Parameter	Description	Default Value
yarn.resourcemanager.recovery.enabled	Whether to enable ResourceManager to restore the status after startup. If this parameter is set to <b>true</b> , <b>yarn.resourcemanager.store.class</b> must also be set.	true
yarn.resourcemanager.store.class	State-store class used to store the application and task statuses and certificate content.	For clusters of versions earlier than MRS 3.x: <b>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore</b>  For clusters of MRS 3.x or later: org.apache.hadoop.yarn.server.resourcemanager.recovery.AsyncZKRMStateStore

Parameter	Description	Default Value
yarn.resourcemanager.zk-state-store.parent-path	Directory for storing ZKRMStateStore in ZooKeeper	/rmstore
yarn.resourcemanager.work-preserving-recovery.enabled	Whether to enable ResourceManager work serving. This configuration is used only for Yarn feature verification.	true
yarn.resourcemanager.state-store.async.load	Whether to apply asynchronous restoration to completed applications.	For clusters of versions earlier than MRS 3.x: <b>false</b> For MRS 3.x or later: <b>true</b>
yarn.resourcemanager.zk-state-store.num-fetch-threads	If asynchronous restoration is enabled, increasing the number of working threads can speed up the restoration of task information stored in ZooKeeper. The value must be greater than 0.	For clusters of versions earlier than MRS 3.x: <b>1</b> For MRS 3.x or later: <b>20</b>

Configure NodeManager Restart as follows:

**Table 27-19** Parameter description of NodeManager Restart

Parameter	Description	Default Value
yarn.nodemanager.recovery.enabled	Whether to enable the function of collecting logs upon a log collection failure when NodeManager is restarted and whether to restore the unfinished application	true
yarn.nodemanager.recovery.dir	Local directory used by NodeManager to store container status It applies to clusters of MRS 3.x or later.	\${SRV_HOME}/tmp/yarn-nm-recovery
yarn.nodemanager.recovery.supervised	Whether NodeManager is monitored. After this parameter is enabled, NodeManager does not clear containers after exit. NodeManager assumes that it will restart and restore containers immediately.	true

## 27.15 Configuring ApplicationMaster Work Preserving

This section applies to clusters of MRS 3.x or later.

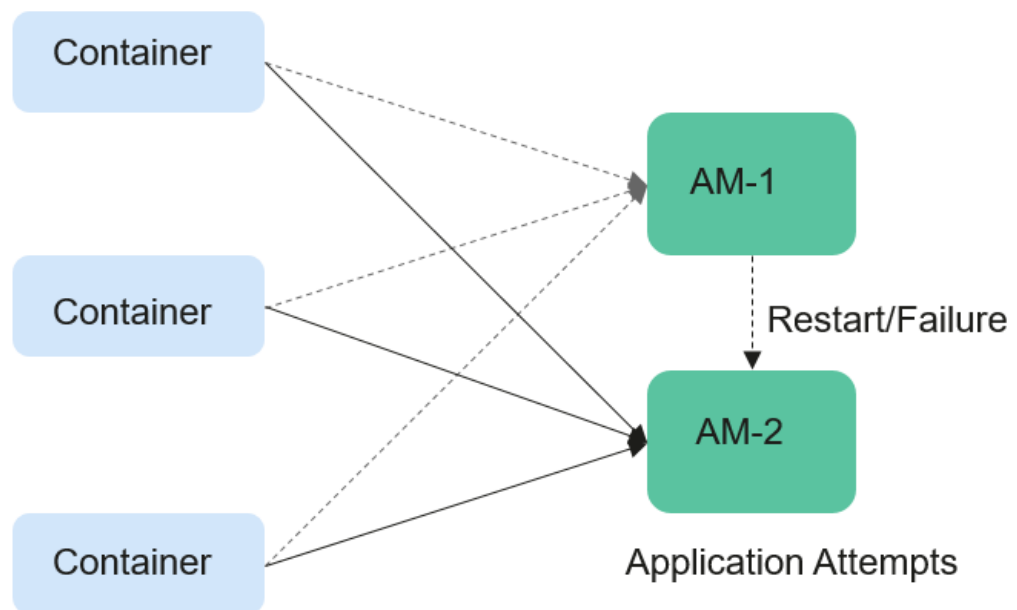
### Scenario

In YARN, ApplicationMasters run on NodeManagers just like every other container (ignoring unmanaged ApplicationMasters in this context). ApplicationMasters may break down, exit, or shut down. If an ApplicationMaster node goes down, ResourceManager kills all the containers of ApplicationAttempt, including containers running on NodeManager. ResourceManager starts a new ApplicationAttempt node on another compute node.

For different types of applications, we want to handle ApplicationMaster restart events in different ways. MapReduce applications aim to prevent task loss but allow the loss of the currently running container. However, for the long-period YARN service, users may not want the service to stop due to the ApplicationMaster fault.

YARN can retain the status of the container when a new ApplicationAttempt is started. Therefore, running jobs can continue to operate without faults.

**Figure 27-1** ApplicationMaster job preserving



### Configuration Description

Go to the **All Configurations** page of Yarn and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

Set the following parameters based on [Table 27-20](#).

**Table 27-20** Parameter description

Parameter	Description	Default Value
yarn.app.mapreduce.am.work-preserve	Whether to enable the ApplicationMaster job retention feature.	false
yarn.app.mapreduce.am.umbilical.max.retries	Maximum number of attempts to restore a running container in the ApplicationMaster job retention feature.	5
yarn.app.mapreduce.am.umbilical.retry.interval	Specifies the interval at which a running container attempts to recover in the ApplicationMaster job retention feature. Unit: millisecond	10000
yarn.resourcemanager.am.max-attempts	The number of retries of ApplicationMaster. Increasing the number of retries prevents ApplicationMaster startup failures caused by insufficient resources.  This applies to global settings of all ApplicationMasters. Each ApplicationMaster can use an API to set an independent maximum number of retries. However, the number of retries cannot be greater than the global maximum number of retries. If the value is greater than the global maximum number of retries, the ResourceManager overwrites the value. The value must be greater than or equal to 1.	2

## 27.16 Configuring the Localized Log Levels

This section applies to clusters of MRS 3.x or later.

### Scenarios

The default log level of localized container is **INFO**. You can change the log level by configuring **yarn.nodemanager.container-localizer.java.opts**.

### Configuration Description

On Manager, choose **Cluster** > *Name of the desired cluster* > **Service** > **Yarn** > **Configuration**. Select **All Configurations** and set the following parameters in the configuration file **yarn-site.xml** of NodeManager to change the log level.

**Table 27-21** Parameter description

Parameter	Description	Default Value
yarn.nodemanager.container-localizer.java.opts	The additional <b>jvm</b> parameters are provided for the localized container process.	-Xmx256m -Djava.security.krb5.conf=\${KRB5_CONFIG}

The default value is **-Xmx256m -Djava.security.krb5.conf=\${KRB5\_CONFIG}** and the default log level is info. To change the localized log level of the container, add the following content:

```
-Dhadoop.root.logger=<LOG_LEVEL>,localizationCLA
```

**Example:**

To change the local log level to **DEBUG**, set the parameter as follows:

```
-Xmx256m -Dhadoop.root.logger=DEBUG,localizationCLA
```

 **NOTE**

Allowed log levels are as follows: FATAL, ERROR, WARN, INFO, DEBUG, TRACE, and ALL.

## 27.17 Configuring Users That Run Tasks

This section applies to clusters of MRS 3.x or later.

### Scenario

Currently, YARN allows the user that starts the NodeManager to run the task submitted by all other users, or the users to run the task submitted by themselves.

### Configuration Description

On Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Yarn** > **Configurations**. Click **All Configurations** Enter a parameter name in the search box.

**Table 27-22** Parameter description

Parameter	Description	Default Value
yarn.nodemanager.linux-container-executor.user	Indicates the user who runs a task.	The value is left blank by default. <b>NOTE</b> The value is left blank by default. The user who submits a task is the actual person who runs the task.

Parameter	Description	Default Value
yarn.nodemanager.container-executor.class	Indicates the executor who starts a task.	org.apache.hadoop.yarn.server.nodemanager.EnhancedLinuxContainerExecutor

 NOTE

- Set **yarn.nodemanager.linux-container-executor.user** to configure the user who runs the container. This parameter is left blank by default. The user who submits the task is the person who runs the container. This parameter is valid only when **yarn.nodemanager.container-executor.class** is set to **org.apache.hadoop.yarn.server.nodemanager.EnhancedLinuxContainerExecutor**.
- In non-security mode, if **yarn.nodemanager.linux-container-executor.user** is set to **omm**, **yarn.nodemanager.linux-container-executor.nonsecure-mode.local-user** must also be set to **omm**.
- For security reasons, it is advised to retain the default values of **yarn.nodemanager.linux-container-executor.user** and **yarn.nodemanager.container-executor.class**.

## 27.18 Yarn Log Overview

### Log Description

The default paths for saving Yarn logs are as follows:

- ResourceManager: **/var/log/Bigdata/yarn/rm** (run logs) and **/var/log/Bigdata/audit/yarn/rm** (audit logs)
- NodeManager: **/var/log/Bigdata/yarn/nm** (run logs) and **/var/log/Bigdata/audit/yarn/nm** (audit logs)

Log archive rule: The automatic compression and archive function is enabled for Yarn logs. By default, when the size of a log file exceeds 50 MB, the log file is automatically compressed. The naming rule of the compressed log file is as follows: *<Original log file name>-<yyyy-mm-dd\_hh-mm-ss>.[ID].log.zip*. A maximum of 100 latest compressed files are retained. The number of compressed files can be configured on Manager.

**Log archive rule:**

**Table 27-23** Yarn log list

Log Type	Log File Name	Description
Run log	hadoop-<SSH_USER>-<process_name>-<hostname>.log	Yarn component log file that records most of the logs generated when the Yarn component is running

Log Type	Log File Name	Description
	hadoop-<SSH_USER>-<process_name>-<hostname>.out	Log file that records Yarn running environment information
	<process_name>-<SSH_USER>-<DATE>-<PID>-gc.log	Garbage collection log file
	yarn-haCheck.log	ResourceManager active/standby status detection log file
	yarn-service-check.log	Log file that records the health check details of the Yarn service
	yarn-start-stop.log	Log file that records the startup and stop of the Yarn service
	yarn-prestart.log	Log file that records cluster operations before the Yarn service startup
	yarn-postinstall.log	Work log file after installation and before startup of the Yarn service
	hadoop-commission.log	Yarn service entry log file
	yarn-cleanup.log	Log file that records the cleanup operation during uninstallation of the Yarn service
	yarn-refreshqueue.log	Yarn queue refresh log file
	upgradeDetail.log	Upgrade log file
	stderr/stdin/syslog	Container log file of the applications running on the Yarn service
	yarn-application-check.log	Check log file of applications running on the Yarn service
	yarn-appsummary.log	Running result log file of applications running on the Yarn service
	yarn-switch-resourcemanager.log	Run log file that records the Yarn active/standby switchover

Log Type	Log File Name	Description
	ranger-yarn-plugin-enable.log	Log file that records the enabling of Ranger authentication for Yarn
	yarn-nodemanager-period-check.log	Periodic check log of Yarn NodeManager
	yarn-resourcemanager-period-check.log	Periodic check log of Yarn ResourceManager
	hadoop.log	Hadoop client logs
	env.log	Environment information log file before the instance is started or stopped.
Audit logs	yarn-audit-<process_name>.log ranger-plugin-audit.log	Yarn operation audit log file
	SecurityAuth.audit	Yarn security audit log file

## Log Level

**Table 27-24** describes the log levels supported by Yarn, including OFF, FATAL, ERROR, WARN, INFO, and DEBUG, from high priority to low. Logs whose levels are higher than or equal to the specified level are printed. The number of printed logs decreases as the specified log level increases.

**Table 27-24** Log levels

Level	Description
FATAL	Logs of this level record critical error information about the current event processing.
ERROR	Logs of this level record error information about the current event processing.
WARN	Logs of this level record exception information about the current event processing.
INFO	Logs of this level record normal running status information about the system and events.
DEBUG	Logs of this level record the system as well as system debugging information.

To modify log levels, perform the following operations:



- Step 1** Go to the **All Configurations** page of the Yarn service by referring to **Modifying Cluster Service Configuration Parameters**.
- Step 2** On the menu bar on the left, select the log menu of the target role.
- Step 3** Select a desired log level.
- Step 4** Click **Save Configuration**. In the dialog box that is displayed, click **OK** to make the setting take effect.

 **NOTE**

The configurations take effect immediately without the need to restart the service.

----End

## Log Format

The following table lists the Yarn log formats.

**Table 27-25** Log formats

Log Type	Format	Example
Run log	<yyyy-MM-dd HH:mm:ss,SSS> <Log Level>  <Thread that generates the log> <Message in the log>  <Location of the log event>	2014-09-26 14:18:59,109   INFO   main   Client environment:java.compiler= <NA>   org.apache.zookeeper.Enviro nment.logEnv(Environment. java:100)
Audit log	<yyyy-MM-dd HH:mm:ss,SSS> <Log Level>  <Thread that generates the log> <Message in the log>  <Location of the log event>	2014-09-26 14:24:43,605   INFO   main-EventThread   USER=omm OPERATION=refreshAdmin Acls TARGET=AdminService RESULT=SUCCESS   org.apache.hadoop.yarn.ser ver.resourcemanager.RMAu ditLogger\$LogLevel \$6.printLog(RMAuditLogger. java:91)

## 27.19 Yarn Performance Tuning

### 27.19.1 Preempting a Task

#### Scenario

The capacity scheduler of ResourceManager implements job preemption to simplify job running in queues and improve resource utilization. The process is as follows:

1. Assume that there are two queues (Queue A and Queue B). The capacity of Queue A is 25%, and the capacity of Queue B is 75%.
2. In the initial state, Task 1 is distributed to Queue A for processing, requiring 75% cluster resources. Task 2 is distributed to Queue B for processing, requiring 50% cluster resources.
3. Task 1 uses 25% cluster resources provided by Queue A and 50% resources from Queue B. Queue B reserves 25% cluster resources.
4. If task preemption is enabled, the resources of Task 1 will be preempted. Queue B preempts 25% cluster resources from Queue A for Task 2.
5. Task 1 will be executed when Task 2 is complete and the cluster has sufficient resources.

## Procedure

Navigation path for setting parameters:

Go to the **All Configurations** page of Yarn and enter a parameter name in the search box by referring to [Modifying Cluster Service Configuration Parameters](#).

**Table 27-26** Parameter description

Parameter	Description	Default Value
yarn.resourcemanager.scheduler.monitor.enable	Whether to start scheduler monitoring according to <b>yarn.resourcemanager.scheduler.monitor.policies</b> . If this parameter is set to <b>true</b> , scheduler monitoring is enabled based on policies specified by <b>yarn.resourcemanager.scheduler.monitor.policies</b> and task resource preemption is enabled based on the scheduler information. If this parameter is set to <b>false</b> , scheduler monitoring is disabled.	false
yarn.resourcemanager.scheduler.monitor.policies	List of the SchedulingEditPolicy class to be used with the scheduler	org.apache.hadoop.yarn.server.resourcemanager.monitor.capacity.ProportionalCapacityPreemptionPolicy

Parameter	Description	Default Value
yarn.resourcemanager.monitor.capacity.preemption.observe_only	<ul style="list-style-type: none"> <li>If this parameter is set to <b>true</b>, policies will be applied but task resource preemption will not be performed.</li> <li>If this parameter is set to <b>false</b>, policies will be applied and task resource preemption will be performed based on the policies.</li> </ul>	false
yarn.resourcemanager.monitor.capacity.preemption.monitoring_interval	Monitoring interval, in millisecond. If this parameter is set to a larger value, capacity detection will not be performed frequently.	3000
yarn.resourcemanager.monitor.capacity.preemption.max_wait_before_kill	Interval between the time when a resource preemption request is sent and the time when the container is stopped (resources are released), in millisecond. The value must be greater than or equal to <b>0</b> . By default, if ApplicationMaster does not stop the container within 15 seconds, ResourceManager will forcibly stop the container after 15 seconds.	15000
yarn.resourcemanager.monitor.capacity.preemption.total_preemption_per_round	Maximum resource preemption ratio in a period. This value can be used to limit the speed at which containers are reclaimed from the cluster. After the expected total preemption value is calculated, the policy scales the preemption ratio back to this limit.	0.1
yarn.resourcemanager.monitor.capacity.preemption.max_ignored_over_capacity	Resource preemption dead zone = Total number of resources in the cluster x Value of this configuration item + Original resources of a queue (for example, Queue A). When resources actually used by a task in Queue A exceeds the preemption dead zone, the resource beyond the preemption dead zone is preempted. The value range is 0 to 1. <b>NOTE</b> A smaller value is recommended for effective preemption.	0

Parameter	Description	Default Value
yarn.resourcemanager.monitor.capacity.preemption.natural_termination_factor	<p>Preemption percentage. Containers preempt only this percentage of the resources.</p> <p>For example, a termination factor of 0.5 will reclaim almost 95% of resources within 5 times of <b>yarn.resourcemanager.monitor.capacity.preemption.max_wait_before_kill</b>, even in the absence of natural termination. That is, 5 consecutive preemptions will be performed and each time half of the target resources will be preempted. The trend is geometric convergence. The interval of each preemption is <b>yarn.resourcemanager.monitor.capacity.preemption.max_wait_before_kill</b>. The value range is 0 to 1.</p>	1

## 27.19.2 Setting the Task Priority

### Scenario

The resource contention scenarios of a cluster are as follows:

1. Submit two jobs (Job 1 and Job 2) with lower priorities.
2. Some tasks of running Job 1 and Job 2 are in the running state. However, some tasks are pending due to resource deficiency because the capacity of cluster or queue resources is limited.
3. Submit a job (Job 3) with a higher priority. In this case, after the running tasks of Job 1 and Job 2 are complete, their resources will be released and then allocated to the pending tasks of Job 3.
4. After Job 3 is complete, its resources will be released and then allocated to Job 1 and Job 2.

Users can use capacity scheduler of ResourceManager to set the task priority in Yarn because the task priority is implemented by the scheduler of ResourceManager.

### Procedure

Set the **mapreduce.job.priority** parameter and use CLI or API to set the task priority.

- Through the CLI  
When submitting tasks, add the **-Dmapreduce.job.priority=<priority>** parameter.

*<priority>* can be set to any of the following values:

- VERY\_HIGH
  - HIGH
  - NORMAL
  - LOW
  - VERY\_LOW
- Through the API  
You can also set the task priority through the API.  
Set `Configuration.set("mapreduce.job.priority", <priority>)` or `Job.setPriority(JobPriority priority)`.

## 27.19.3 Optimizing Node Configuration

### Scenario

After the scheduler of a big data cluster is properly configured, you can adjust the available memory, CPU resources, and local disk of each node to optimize the performance.

The configuration items are as follows:

- Available memory
- Number of vCPUs
- Physical CPU usage
- Coordination of memory and CPU resources
- Local disk

### Procedure

For details about how to adjust parameter settings, see [Modifying Cluster Service Configuration Parameters](#).

- **Available memory**

Except the memory allocated to the OS and other services, allocate as much as possible memory to Yarn. You can adjust the following parameters to improve resource utilization.

Assume that a container uses 512 MB memory by default, then the memory usage formula is: 512 MB x Number of containers.

By default, the Map or Reduce container uses one vCPU and 1,024 MB memory, and ApplicationMaster uses 1,536 MB memory.

Parameter	Description	Default Value
yarn.nodemanager.resourcememory-mb	Physical memory that can be allocated to containers, in MB. The value must be greater than 0. You are advised to set the parameter value to 75% to 90% of the total physical memory of nodes. If the node has permanent processes of other services, reduce this parameter value to reserve sufficient resources for the processes.	MRS 3.x or later: <b>16384</b> Versions earlier than MRS 3.x: <b>8192</b>

- **Number of vCPUs**

You are advised to set this parameter to 1.5 to 2 times the number of logical CPUs. If the upper layer computing applications have low computing capability requirements, you can set the parameter to two times the number of logical CPUs.

Parameter	Description	Default Value
yarn.nodemanager.resourcememory-cpu-vcores	Number of vCPUs that can be used by Yarn on the node. The default value is <b>8</b> . You are advised to set the value to 1.5 to 2 times the number of logical CPUs.	8

- **Physical CPU usage**

You are advised to reserve appropriate CPUs for the OS and the processes, such as database and HBase, and allocate the remaining CPUs to Yarn. You can set the following parameters to adjust the physical CPU usage.

Parameter	Description	Default Value
yarn.nodemanager.resource.percentage-physical-cpu-limit	<p>Physical CPU percentage that can be used by Yarn on a node. The default value is <b>90</b>, indicating that no CPU control is implemented and Yarn can use all CPU resources. You can only view the parameter. To change the value of this parameter, set the value of RES_CPUSET_PERCENTAGE of YARN. You are advised to set this parameter to the percentage of CPU resources that can be used by the YARN cluster.</p> <p>For example, If 20% of CPU resources are used by other services (such as HBase, HDFS, and Hive) and system processes on the node, the CPU resources can be scheduled for Yarn is <math>1 - 20\% = 80\%</math>. Therefore, you can set this parameter to <b>80</b>.</p>	90

- **Local disk**

MapReduce writes the intermediate job execution results in local disks. Therefore, configure disks as much as possible and disk space as large as possible. A simple way is to configure the same number of disks as DataNode except for the last directory.

 **NOTE**

Use commas (,) to separate multiple disks.

Parameter	Description	Default Value
yarn.nodemanager.log-dirs	<p>Directories in which logs are stored. Multiple directories can be specified.</p> <p>Storage location of container logs. The default value is % <b>{@auto.detect.datapart.nm.logs}</b>. If there is a data partition, a path list similar to <b>/srv/BigData/hadoop/data1/nm/containerlogs,/srv/BigData/hadoop/data2/nm/containerlogs</b> is generated based on the data partition. If there is no data partition, the default path <b>/srv/BigData/yarn/data1/nm/containerlogs</b> is generated. In addition to using expressions, you can enter a complete list of paths, such as <b>/srv/BigData/yarn/data1/nm/containerlogs</b> or <b>/srv/BigData/yarn/data1/nm/containerlogs,/srv/BigData/yarn/data2/nm/containerlogs</b>. In this way, data is stored in all the configured directories, which are usually on different devices. To ensure disk I/O load balancing, you are advised to provide several paths and each path corresponds to an independent disk. The localized log directory of the application exists in the relative path <b>/application_%{appid}</b>. The log directory of an independent container, that is, <b>container_{\$contid}</b>, is the subdirectory of this directory. Each container directory contains the <b>stderr</b>, <b>stdin</b>, and <b>syslog</b> files generated by the container. To add a directory, for example, <b>/srv/BigData/yarn/data2/nm/containerlogs</b>, you need to delete the files in <b>/srv/BigData/yarn/data2/nm/containerlogs</b> first. Then, assign the same read and write permissions to <b>/srv/BigData/yarn/data2/nm/containerlogs</b> as those of <b>/srv/</b></p>	<p>% {@auto.detect.datapart.nm.logs}</p>



Parameter	Description	Default Value
	<p><b>BigData/yarn/data1/nm/containerlogs</b>, and change <b>/srv/BigData/yarn/data1/nm/containerlogs</b> to <b>/srv/BigData/yarn/data1/nm/containerlogs,/srv/BigData/yarn/data2/nm/containerlogs</b>. You can add directories, but do not modify or delete existing directories. Otherwise, NodeManager data will be lost and services will be unavailable.</p> <p>Default value: % <b>{@auto.detect.datapart.nm.logs}</b> }</p> <p>Exercise caution when modifying this parameter. If the configuration is incorrect, the services are unavailable. If the value of this configuration item at the role level is changed, the value of this configuration item at all instance levels will be changed. If the value of this configuration item at the instance level is changed, the value of this configuration item of other instances remains unchanged.</p>	

Parameter	Description	Default Value
yarn.nodemanager.local-dirs	<p>Storage location of files after localization. The default value is %  <b>{@auto.detect.datapart.nm.localdir}</b>. If there is a data partition, a path list similar to <b>/srv/BigData/hadoop/data1/nm/localdir,/srv/BigData/hadoop/data2/nm/localdir</b> is generated based on the data partition. If there is no data partition, the default path <b>/srv/BigData/yarn/data1/nm/localdir</b> is generated. In addition to using expressions, you can enter a complete list of paths, such as <b>/srv/BigData/yarn/data1/nm/localdir</b> or <b>/srv/BigData/yarn/data1/nm/localdir,/srv/BigData/yarn/data2/nm/localdir</b>. In this way, data is stored in all the configured directories, which are usually on different devices. To ensure disk I/O load balancing, you are advised to provide several paths and each path corresponds to an independent disk. The localized file directory of the application is stored in the relative path <b>/usercache/%{user}/appcache/application_%{appid}</b>. The working directory of an independent container, that is, <b>container_%{contid}</b>, is the subdirectory of the directory. To add a directory, for example, <b>/srv/BigData/yarn/data2/nm/localdir</b>, you need to delete the files in <b>/srv/BigData/yarn/data2/nm/localdir</b> first. Then, assign the same read and write permissions to <b>/srv/BigData/hadoop/data2/nm/localdir</b> as those of <b>/srv/BigData/hadoop/data1/nm/localdir</b>, and change <b>/srv/BigData/yarn/data1/nm/localdir</b> to <b>/srv/BigData/yarn/data1/nm/localdir,/srv/BigData/yarn/data2/nm/localdir</b>. You can add</p>	<p>%  <b>{@auto.detect.datapart.nm.localdir}</b></p>

Parameter	Description	Default Value
	<p>directories, but do not modify or delete existing directories. Otherwise, NodeManager data will be lost and services will be unavailable.</p> <p>Default value: % <b>{@auto.detect.datapart.nm.local dir}</b></p> <p>Exercise caution when modifying this parameter. If the configuration is incorrect, the services are unavailable. If the value of this configuration item at the role level is changed, the value of this configuration item at all instance levels will be changed. If the value of this configuration item at the instance level is changed, the value of this configuration item of other instances remains unchanged.</p>	

## 27.20 Common Issues About Yarn

### 27.20.1 Why Mounted Directory for Container is Not Cleared After the Completion of the Job While Using CGroups?

#### Question

Why mounted directory for Container is not cleared after the completion of the job while using CGroups?

#### Answer

The mounted path for the Container should be cleared even if job is failed.

This happens due to the deletion timeout. Some task takes more time to complete than the deletion time.

To avoid this scenario, you can go to the **All Configurations** page of Yarn by referring to [Modifying Cluster Service Configuration Parameters](#). Search for the **yarn.nodemanager.linux-container-executor.cgroups.delete-timeout-ms** configuration item in the search box to change the deletion interval. The value is in milliseconds.

## 27.20.2 Why the Job Fails with HDFS\_DELEGATION\_TOKEN Expired Exception?

### Question

Why is the HDFS\_DELEGATION\_TOKEN expired exception reported when a job fails in security mode?

### Answer

HDFS\_DELEGATION\_TOKEN expires because the token is not updated or it is accessed after max. lifetime.

Ensure the following parameter value of max. lifetime of the token is greater than the job running time.

**dfs.namenode.delegation.token.max-lifetime=604800000** (1 week by default)

Go to the **All Configurations** page of HDFS by referring to [Modifying Cluster Service Configuration Parameters](#) and search for this parameter in the search box.

#### NOTE

You are advised to set this parameter to a value that is multiple times of the number of hours within the max. lifecycle of the token.

## 27.20.3 Why Are Local Logs Not Deleted After YARN Is Restarted?

### Question

If Yarn is restarted in either of the following scenarios, local logs will not be deleted as scheduled and will be retained permanently:

- When Yarn is restarted during task running, local logs are not deleted.
- When the task is complete and logs fail to be collected, restart Yarn before the logs are cleared as scheduled. In this case, local logs are not deleted.

### Answer

NodeManager has a restart recovery mechanism (for details, see [https://hadoop.apache.org/docs/r3.1.1/hadoop-yarn/hadoop-yarn-site/NodeManager.html#NodeManager\\_Restart](https://hadoop.apache.org/docs/r3.1.1/hadoop-yarn/hadoop-yarn-site/NodeManager.html#NodeManager_Restart)). Go to the **All Configurations** page of Yarn by referring to [Modifying Cluster Service Configuration Parameters](#). Set **yarn.nodemanager.recovery.enabled** of NodeManager to **true** to make the configuration take effect. The default value is **true**. In this way, redundant local logs are periodically deleted when the YARN is restarted.

## 27.20.4 Why the Task Does Not Fail Even Though AppAttempts Restarts for More Than Two Times?

### Question

Why the task does not fail even though AppAttempts restarts due to failure for more than two times?

### Answer

During the task execution process, if the **ContainerExitStatus** returns value **ABORTED**, **PREEMPTED**, **DISKS\_FAILED**, or **KILLED\_BY\_RESOURCEMANAGER**, the system will not count it as a failed attempt. Therefore, the task fails only when the AppAttempts fails actually, that is, the return value is not **ABORTED**, **PREEMPTED**, **DISKS\_FAILED**, or **KILLED\_BY\_RESOURCEMANAGER** for two times.

## 27.20.5 Why Is an Application Moved Back to the Original Queue After ResourceManager Restarts?

### Question

After I moved an application from one queue to another, why is it moved back to the original queue after ResourceManager restarts?

### Answer

This problem is caused by the constraints of the ResourceManager. If a running application is moved to another queue, information about the new queue will not be stored in the ResourceManager after the ResourceManager restarts.

Assume that a user submits a MapReduce application to the leaf queue test11. If the leaf queue test11 is deleted when the application is running, the application will go to the lost\_and\_found queue and the application stops. To start the application, the user moves the application to the leaf queue test21 and the application resumes running. If the ResourceManager restarts, the displayed submission queue is lost\_and\_found, but not test21.

If the application is not complete, the ResourceManager only stores the queue information before the application is moved. As a result, the application is moved back to the original queue. To solve this problem, move the application again after the ResourceManager is restarted to write information about the new queue to the ResourceManager.

## 27.20.6 Why Does Yarn Not Release the Blacklist Even All Nodes Are Added to the Blacklist?

### Question

Why does Yarn not release the blacklist even all nodes are added to the blacklist?

## Answer

In Yarn, when the number of application nodes added to the blacklist by ApplicationMaster (AM) reaches a certain proportion (the default value is 33% of the total number of nodes), the AM automatically releases the blacklist. In this way, all available nodes are added to the blacklist and tasks can obtain node resources.

Assume that there are 8 nodes in a cluster and they are divided into pool A and pool B by NodeLabel. There are two nodes in pool B. A user submits a task App1 to pool B, but there is not sufficient HDFS space and App1 fails to run. As a result, two nodes in pool B are added to the blacklist by the AM of App1. According to the preceding principles, 2 is less than the 33% of 8. Therefore, Yarn does not release the blacklist, and App1 cannot obtain resources and keeps running. Even if the node that is added to the blacklisted is recovered, App1 still cannot obtain resources.

The preceding principles do not apply to the resource pool scenario. Therefore, you can change the value of the client parameter **yarn.resourcemanager.am-scheduling.node-blacklisting-disable-threshold** to **(nodes number of pool / total nodes) \* 33%** to solve this problem.

## 27.20.7 Why Does the Switchover of ResourceManager Occur Continuously?

### Question

The switchover of ResourceManager occurs continuously when multiple, for example 2,000, tasks are running concurrently, causing the Yarn service unavailable.

### Answer

The cause is that the time of full GarbageCollection exceeds the interaction duration threshold between the ResourceManager and ZooKeeper duration threshold. As a result, the connection between the ResourceManager and ZooKeeper fails and the switchover of ResourceManager occurs continuously.

When there are multiple tasks, ResourceManager saves the authentication information about multiple tasks and transfers the information to NodeManagers through heartbeat, which is called heartbeat response. The lifecycle of heartbeat response is short. The default value is 1s. Normally, heartbeat response can be reclaimed during the JVM minor GarbageCollection. However, if there are multiple tasks and there are a lot of nodes, for example 5000 nodes, in the cluster, the heartbeat response of multiple nodes occupy a large amount of memory. As a result, the JVM cannot completely reclaim the heartbeat response during minor GarbageCollection. The heartbeat response failed to be reclaimed accumulate and the JVM full GarbageCollection is triggered. The JVM GarbageCollection is in a blocking mode, in other words, no jobs are performed during the GarbageCollection. Therefore, if the duration of full GarbageCollection exceeds the periodical interaction duration threshold between the ResourceManager and ZooKeeper, the switchover occurs.

Log in to the active node of the cluster as user **root**. Add the **yarn.resourcemanager.zk-timeout-ms** customized parameter to the **yarn-**

**site.xml** file in the *Client installation path/Yarn/config/* directory to increase the threshold of the periodic interaction duration between ResourceManager and ZooKeeper (the value must be less than or equal to 90,000 ms) to solve the problem of continuous active/standby ResourceManager switchover.

## 27.20.8 Why Does a New Application Fail If a NodeManager Has Been in Unhealthy Status for 10 Minutes?

### Question

Why does a new application fail if a NodeManager has been in unhealthy status for 10 minutes?

### Answer

When **nodeSelectPolicy** is set to **SEQUENCE** and the first NodeManager connected to the ResourceManager is unavailable, the ResourceManager attempts to assign tasks to the same NodeManager in the period specified by **yarn.nm.liveness-monitor.expiry-interval-ms**.

You can use either of the following methods to avoid the preceding problem:

- Use another nodeSelectPolicy, for example, **RANDOM**.
- Go to the **All Configurations** page of Yarn by referring to [Modifying Cluster Service Configuration Parameters](#). Search for the following parameters in the search box and modify the following attributes in the **yarn-site.xml** file:  
**yarn.resourcemanager.am-scheduling.node-blacklisting-enabled = true;**  
**yarn.resourcemanager.am-scheduling.node-blacklisting-disable-threshold = 0.5.**

## 27.20.9 Why Does an Error Occur When I Query the ApplicationID of a Completed or Non-existing Application Using the RESTful APIs?

### Question

Why does an error occur when I query the applicationID of a completed or non-existing application using the RESTful APIs?

### Answer

The Superior scheduler only stores the applicationIDs of running applications. If you view the applicationID of a completed or non-existing application by accessing the RESTful API at **https://<SS\_REST\_SERVER>/ws/v1/sscheduler/applications/{application\_id}**, the 404 error is returned by the server. If Chrome web browser is used, the **Error Occurred** message is displayed because Chrome preferentially responds in the application/xml format. If Internet Explorer is used, the **404** error code is displayed because IE web browser preferentially responds in the application/json format.

## 27.20.10 Why May A Single NodeManager Fault Cause MapReduce Task Failures in the Superior Scheduling Mode?

### Question

In Superior scheduling mode, if a single NodeManager is faulty, why may the MapReduce tasks fail?

### Answer

In normal cases, when the attempt of a single task of an application fails on a node for three consecutive times, the AppMaster of the application adds the node to the blacklist. Then, the AppMaster instructs the scheduler not to schedule the task to the node to avoid task failure.

However, by default, if 33% nodes in the cluster are added to the blacklist, the scheduler ignores the blacklisted nodes. Therefore, the blacklist feature is prone to become invalid in small cluster scenarios. For example, there are only three nodes in the cluster. If one node is faulty, the blacklist mechanism becomes invalid. The scheduler continues to schedule the task to the node no matter how many times the attempt of the task fails on the node. As a result, the number of attempts of the task reaches the maximum (4 times by default for MapReduce). And the MapReduce tasks failed.

Workaround:

The **yarn.resourcemanager.am-scheduling.node-blacklisting-disable-threshold** parameter indicates the threshold for ignoring blacklisted nodes, in percentage. You are advised to increase the value of this parameter based on the cluster scale. For example, you are advised to set this parameter to **50%** for a three-node cluster.

#### NOTE

The framework design of the Superior scheduler is time-based asynchronous scheduling. When the NodeManager is faulty, ResourceManager cannot quickly detect that the NodeManager is faulty (10 minutes by default). Therefore, the Superior scheduler still schedules tasks to the node, causing task failures.

## 27.20.11 Why Are Applications Suspended After They Are Moved From Lost\_and\_Found Queue to Another Queue?

### Question

When a queue is deleted when there are applications running in it, these applications are moved to the "lost\_and\_found" queue. When these applications are moved back to another healthy queue, some tasks are suspended.

### Answer

If no label expression is set for the current application, the default label expression of the queue is used as label expression for new container/resource demands requested by the application. If there is no default label expression of the queue,



then **default label** is considered as the label expression for new container/resource demands requested by the application.

When application app1 is submitted to the queue Q1, **label1**, the default label expression of the queue, is used for the application's new resource requests/containers. If Q1 is deleted when app1 is running, app1 is moved to the "lost\_and\_found" queue. Because there is no label expression of the "lost\_and\_found" queue, **default label** is used as the label expression of app1's new resource requests/containers. Assume that app1 is moved to another normal queue Q2. If Q2 supports **label1** and **default label**, app1 can run properly. If Q2 does not support **label1** or **default label**, the resource request with **label1** or **default label** cannot obtain resources, causing task suspension.

To solve this problem, ensure that the queue to which the application is moved from "lost\_and\_found" queue supports label expression of the moved application.

You are not advised to delete a queue in which there are running applications.

## 27.20.12 How Do I Limit the Size of Application Diagnostic Messages Stored in the ZKstore?

### Question

How do I limit the size of application diagnostic messages stored in the ZKstore?

### Answer

In some cases, it has been observed that diagnostic messages may grow infinitely. Because diagnostic messages are stored in the ZKstore, it is not recommended that you allow diagnostic messages to grow indefinitely. Therefore, a property parameter is needed to set the maximum size of the diagnostic message.

If you need to set **yarn.app.attempt.diagnostics.limit.kc**, go to the **All Configurations** page by referring to [Modifying Cluster Service Configuration Parameters](#) and search for the following parameters in the search box:

**Table 27-27** Parameter description

Parameter	Description	Default Value
yarn.app.attempt.diagnostics.limit.kc	Data size of the diagnosis message for each application connection, in kilobytes (number of characters x 1,024). When ZooKeeper is used to store the behavior status of applications, the size of diagnosis messages needs to be limited to prevent Yarn from overloading ZooKeeper. If <b>yarn.resourcemanager.state-store.max-completed-applications</b> is set to a large value, you need to decrease the value of this property to limit the total size of stored data.	64

## 27.20.13 Why Does a MapReduce Job Fail to Run When a Non-ViewFS File System Is Configured as ViewFS?

### Question

Why does a MapReduce job fail to run when a non-ViewFS file system is configured as ViewFS?

### Answer

When a non-ViewFS file system is configured as a ViewFS using cluster, the user permissions on folders in the ViewFS file system are different from those of non-ViewFS folders in the default NameService. The submitted MapReduce job fails to be executed because the directory permissions are inconsistent.

When configuring the ViewFS user in the cluster, you need to check and verify the directory permissions. Before submitting a job, change the ViewFS folder permissions based on the default NameService folder permissions.

The following table lists the default permission structure of directories configured in ViewFS. If the configured directory permissions are not included in the following table, you must change the directory permissions accordingly.

**Table 27-28** Default permission structure of directories configured in ViewFS

Parameter	Description	Default Value	Default value and default permissions on the parent directory
yarn.nodemanager.remote-app-log-dir	On the default file system (usually HDFS), specify the directory to which the NM aggregates logs.	logs	777
yarn.nodemanager.remote-app-log-archive-dir	Directory for archiving logs	-	777
yarn.app.mapreduce.am.staging-dir	Staging directory used when a job is submitted	/tmp/hadoop-yarn/staging	777
mapreduce.jobhistory.intermediate-done-dir	Directory for storing historical files of MapReduce jobs	\${yarn.app.mapreduce.am.staging-dir}/history/done_intermediate	777

Parameter	Description	Default Value	Default value and default permissions on the parent directory
mapreduce.jobhistory.done-dir	Directory of historical files managed by the MR JobHistory Server.	\${yarn.app.mapreduce.am.staging-dir}/history/done	777

## 27.20.14 Why Do Reduce Tasks Fail to Run in Some OSs After the Native Task Feature is Enabled?

### Question

After the Native Task feature is enabled, Reduce tasks fail to run in some OSs.

### Answer

When - **Dmapreduce.job.map.output.collector.class=org.apache.hadoop.mapred.native.task.NativeMapOutputCollectorDelegator** is executed to enable the Native Task feature during the running of MapReduce tasks that contain Reduce tasks, the tasks fail to run in some OSs, and the error message "version 'GLIBCXX\_3.4.20' not found" is displayed in logs. The cause is that the GLIBCXX version of the OSs is too early. As a result, the libnativetask.so.1.0.0 library on which the feature depends cannot be loaded, leading to task failures.

Workaround:

Set **mapreduce.job.map.output.collector.class** to **org.apache.hadoop.mapred.MapTask\$MapOutputBuffer**.

# 28 Using ZooKeeper

---

## 28.1 Configuring the ZooKeeper Permissions

### Scenario

Configure znode permission of ZooKeeper.

ZooKeeper uses an access control list (ACL) to implement znode access control. The ZooKeeper client specifies a znode ACL, and the ZooKeeper server determines whether a client that requests for a znode has related operation permission according to the ACL. ACL configuration involves the following four operations:

- Check znode ACLs in ZooKeeper.
- Add znode ACLs to ZooKeeper.
- Modify znode ACLs in ZooKeeper.
- Delete znode ACLs from ZooKeeper.

The ZooKeeper ACL permission is described as follows:

ZooKeeper supports five types of permission, create, delete, read, write, and admin. ZooKeeper permission control is of a znode level. That is, the permission configuration for a parent znode is not inherited by its child znodes. The ZooKeeper znode default permission is **world:anyone: cdrwa**. That is, any user has all permissions.

#### NOTE

ACL has three parts:

The first part is the authentication type. For example, **world** indicates all authentication types and **sasl** indicates the kerberos authentication type.

The second part is the account. For example, anyone indicates any user.

The third part is permission. For example, **cdrwa** indicates all permissions.

In particular, because starting the client in common mode does not need authentication, ACL with **sasl** authentication type cannot be used in common mode. Authentications of **sasl** scheme in this document are performed in clusters that have the security mode enabled.

**Table 28-1** Five types of ZooKeeper ACLs

Permission Description	Permission Name	Permission Details
Create permission	create(c)	Users with this permission can create child znodes in the current znode.
Delete permission	delete(d)	Users with this permission can delete the current znode.
Read permission	read(r)	Users with this permission can obtain data of the current znode and list all the child znodes of the current znode.
Write permission	write(w)	Users with this permission can write data to the current znode and its child znodes.
Administration permission	admin(a)	Users with this permission can set permission for the current znode.

## Impact on the System

### NOTICE

Modifying ZooKeeper ACLs is a critical operation. If znode permission is modified in ZooKeeper, other users may have no permission to access the znode and some system functions are abnormal. In 3.5.6 and later versions, users must have the read permission for the **getAcl** operation.

## Prerequisites

- The ZooKeeper client has been installed. For example, the installation directory is **/opt/client**.
- You have obtained the password of the MRS cluster administrator account.

## Procedure

**Start the ZooKeeper client.**

**Step 1** Log in to the server where the ZooKeeper client is installed as user **root**.

**Step 2** Run the following command to go to the client installation directory:

```
cd /opt/client
```

**Step 3** Run the following command to configure environment variables:

```
source bigdata_env
```

- Step 4** If the cluster has the security mode enabled, run the following command for user authentication and enter the username and password (Any authorized user. **admin** is used as an example.):

**kinit admin**

- Step 5** On the ZooKeeper client, run the following command to go to the ZooKeeper command-line interface (CLI):

**sh zkCli.sh -server ZooKeeper plane IP address of any instance:clientPort**

The default **clientPort** is **2181**.

Example: **sh zkCli.sh -server 192.168.0.151:2181**

- Step 6** Run the **ls** command to view the znode list in ZooKeeper. For example, you can view the list of znodes in the root directory.

**ls /**

```
[zk: 192.168.0.151:2181(CONNECTED) 1] ls /
[hadoop-flag, hadoop-ha, test, test2, test3, test4, test5, test6, zookeeper]
```

**View the ZooKeeper znode ACL.**

- Step 7** Start the ZooKeeper client.

- Step 8** Run the **getAcl** command to view znodes. The following command can be used to view the created znode ACL named **test**:

**getAcl /znode name**

```
[zk: 192.168.0.151:2181(CONNECTED) 2] getAcl /test
'world,'anyone
: cdrwa
```

Add a ZooKeeper znode ACL.

- Step 9** Start the ZooKeeper client.

- Step 10** View the old ACL information to check whether the current account has the permission to modify the znode ACL information (a permission). If no, use **kinit** to switch to a user that has the permission and restart the ZooKeeper client.

**getAcl /znode name**

```
[zk: 192.168.0.151:2181(CONNECTED) 3] getAcl /test
'world,'anyone
: cdrwa
```

- Step 11** Run the **setAcl** command to add an ACL. The command for adding an ACL is as follows:

**setAcl /test world:anyone:cdrwa,sasl: username@: <system domain name>:ACL value**

For example, to add the ACL of the **admin** user to the **test** znode, run the following command:

**setAcl /test world:anyone:cdrwa,sasl:admin@HADOOP.COM:cdrwa**

 **NOTE**

When adding a new ACL, reserve the existing ones. The new and old ACLs are separated by a comma. The newly added ACL has three parts:

The first part is the authentication type. For example, **sasl** indicates the kerberos authentication type.

The second part is the account. For example, **admin@HADOOP.COM** indicates user **admin**.

The third part is permission. For example, **cdrwa** indicates all permissions.

**Step 12** After adding the ACL, run the **getAcl** command to check whether the permission is added successfully:

**getAcl** /znode name

```
[zk: 192.168.0.151:2181(CONNECTED) 4] getAcl /test
'world,'anyone
: cdrwa
'sasl,'admin@<system domain name>
: cdrwa
```

**Modify the ZooKeeper znode ACL.**

**Step 13** Start the ZooKeeper client.

**Step 14** View the old ACL information to check whether the current account has the permission to modify the znode ACL information (a permission). If no, use kinit to switch to a user that has the permission and restart the ZooKeeper client.

**getAcl** /znode name

```
[zk: 192.168.0.151:2181(CONNECTED) 5] getAcl /test
'world,'anyone
: cdrwa
'sasl,'admin@<system domain name>
: cdrwa
```

**Step 15** Run the **setAcl** command to modify an ACL. The command for adding an ACL is as follows:

**setAcl** /test sasl:Username@<System domain name>:ACL value

For example, to reserve only **admin** user permission and delete **anyone** rw permission, run the following command:

**setAcl** /test sasl:admin@HADOOP.COM:cdrwa

**Step 16** After modifying the ACL, run the **getAcl** command to check whether the permission is modified successfully:

**getAcl** /znode name

```
[zk: 192.168.0.151:2181(CONNECTED) 6] getAcl /test
'sasl,'admin@<system domain name>
: cdrwa
```

**Delete the ZooKeeper znode ACL.**

**Step 17** Start the ZooKeeper client.

**Step 18** View the old ACL information to check whether the current account has the permission to modify the znode ACL information (a permission). If no, use kinit to switch to a user that has the permission and restart the ZooKeeper client.

**getAcl** /znode name

```
[zk: 192.168.0.151:2181 (CONNECTED) 5] getAcl /test
'world,'anyone
: rw
'sasl,'admin@<system domain name>
: cdrwa
```

**Step 19** Run the **setAcl** command to add an ACL. The command for adding an ACL is as follows:

**setAcl** /test sasl:Username@<System domain name>:ACL value

For example, to reserve only **admin** user permission and delete **anyone** rw permission, run the following command:

**setAcl** /test sasl:admin@HADOOP.COM:cdrwa

**Step 20** After modifying the ACL, run the **getAcl** command to check whether the permission is modified successfully:

**getAcl** /znode name

```
[zk: 192.168.0.151:2181 (CONNECTED) 6] getAcl /test
'sasl,'admin@<system domain name>
: cdrwa
```

----End



# 29 Appendix

---

## 29.1 Modifying Cluster Service Configuration Parameters

- You can modify service configuration parameters on the cluster management page of the MRS management console.
  - a. Log in to the MRS console. In the left navigation pane, choose **Clusters > Active Clusters**, and click a cluster name.
  - b. Choose **Components > Name of the desired service > Service Configuration**.

The **Basic Configuration** tab page is displayed by default. To modify more parameters, click the **All Configurations** tab. The navigation tree displays all configuration parameters of the service. The level-1 nodes in the navigation tree are service names or role names. The parameter category is displayed after the level-1 node is expanded.
  - c. In the navigation tree, select the specified parameter category and change the parameter values on the right.

If you are not sure about the location of a parameter, you can enter the parameter name in search box in the upper right corner. The system searches for the parameter in real time and displays the result.
  - d. Click **Save Configuration**. In the displayed dialog box, click **OK**.
  - e. Wait until the message **Operation successful** is displayed. Click **Finish**.

The configuration is modified.

Check whether there is any service whose configuration has expired in the cluster. If yes, restart the corresponding service or role instance for the configuration to take effect. You can also select **Restart the affected services or instances** when saving the configuration. .
- For MRS 3.x or earlier: You can log in to MRS Manager to modify service configuration parameters.
  - a. Log in to MRS Manager.
  - b. Click **Services**.

- c. Click the specified service name on the service management page.
- d. Click **Service Configuration**.

The **Basic Configuration** tab page is displayed by default. To modify more parameters, click the **All Configurations** tab. The navigation tree displays all configuration parameters of the service. The level-1 nodes in the navigation tree are service names or role names. The parameter category is displayed after the level-1 node is expanded.

- e. In the navigation tree, select the specified parameter category and change the parameter values on the right.

If you are not sure about the location of a parameter, you can enter the parameter name in search box in the upper right corner. The system searches for the parameter in real time and displays the result.

- f. Click **Save**. In the confirmation dialog box, click **OK**.
- g. Wait until the message **Operation successful** is displayed. Click **Finish**.  
The configuration is modified.

Check whether there is any service whose configuration has expired in the cluster. If yes, restart the corresponding service or role instance for the configuration to take effect. You can also select **Restart the affected services or instances** when saving the configuration.

- For MRS 3.x or later: You can log in to FusionInsight Manager to modify service configuration parameters.

- a. You have logged in to FusionInsight Manager.
- b. Choose **Cluster > Service**.
- c. Click the specified service name on the service management page.
- d. Click **Configuration**.

The **Basic Configuration** tab page is displayed by default. To modify more parameters, click the **All Configurations** tab. The navigation tree displays all configuration parameters of the service. The level-1 nodes in the navigation tree are service names or role names. The parameter category is displayed after the level-1 node is expanded.

- e. In the navigation tree, select the specified parameter category and change the parameter values on the right.

If you are not sure about the location of a parameter, you can enter the parameter name in search box in the upper right corner. The system searches for the parameter in real time and displays the result.

- f. Click **Save**. In the confirmation dialog box, click **OK**.
- g. Wait until the message **Operation successful** is displayed. Click **Finish**.  
The configuration is modified.

Check whether there is any service whose configuration has expired in the cluster. If yes, restart the corresponding service or role instance for the configuration to take effect.

## 29.2 Accessing Manager

## 29.2.1 Accessing MRS Manager (Versions Earlier Than MRS 3.x)

### Scenario

Clusters of versions earlier than MRS 3.x use MRS Manager to monitor, configure, and manage clusters. You can open the MRS Manager page on the MRS console.

### Accessing MRS manager

**Step 1** Log in to the MRS management console.

**Step 2** In the navigation pane, choose **Clusters > Active Clusters**. Click the target cluster name to access the cluster details page.

**Step 3** Click **Access Manager**. The **Access MRS Manager** page is displayed.

- If you have bound an EIP when creating a cluster,
  - a. Select the security group to which the security group rule to be added belongs. The security group is configured when the cluster is created.
  - b. Add a security group rule. By default, your public IP address used for accessing port 9022 is filled in the rule. To enable multiple IP address segments to access MRS Manager, see [Step 6](#) to [Step 9](#). If you want to view, modify, or delete a security group rule, click **Manage Security Group Rule**.

#### NOTE

- It is normal that the automatically generated public IP address is different from the local IP address and no action is required.
- If port 9022 is a Knox port, you need to enable the permission of port 9022 to access Knox for accessing MRS Manager.
- c. Select the checkbox stating that **I confirm that xx.xx.xx.xx is a trusted public IP address and MRS Manager can be accessed using this IP address**.
- If you have not bound an EIP when creating a cluster,
  - a. Select an available EIP from the drop-down list or click **Manage EIP** to create one.
  - b. Select the security group to which the security group rule to be added belongs. The security group is configured when the cluster is created.
  - c. Add a security group rule. By default, your public IP address used for accessing port 9022 is filled in the rule. To enable multiple IP address segments to access MRS Manager, see [Step 6](#) to [Step 9](#). If you want to view, modify, or delete a security group rule, click **Manage Security Group Rule**.

 NOTE

- It is normal that the automatically generated public IP address is different from the local IP address and no action is required.
  - If port 9022 is a Knox port, you need to enable the permission of port 9022 to access Knox for accessing MRS Manager.
- d. Select the checkbox stating that **I confirm that xx.xx.xx.xx is a trusted public IP address and MRS Manager can be accessed using this IP address.**

**Step 4** Click **OK**. The MRS Manager login page is displayed.

**Step 5** Enter the default username **admin** and the password set during cluster creation, and click **Log In**. The MRS Manager page is displayed.

**Step 6** On the MRS console, click **Clusters** and choose **Active Clusters**. Click the target cluster name to access the cluster details page.

 NOTE

To assign MRS Manager access permissions to other users, follow instructions from [Step 6](#) to [Step 9](#) to add the users' public IP addresses to the trusted range.

**Step 7** Click **Add Security Group Rule** on the right of **EIP**.

**Step 8** On the **Add Security Group Rule** page, add the IP address segment for users to access the public network and select **I confirm that the authorized object is a trusted public IP address range. Do not use 0.0.0.0/0. Otherwise, security risks may arise.**

By default, the IP address used for accessing the public network is filled. You can change the IP address segment as required. To enable multiple IP address segments, repeat steps [Step 6](#) to [Step 9](#). If you want to view, modify, or delete a security group rule, click **Manage Security Group Rule**.

**Step 9** Click **OK**.

----End

## Granting the Permission to Access MRS Manager to Other Users

**Step 1** On the MRS console, click **Clusters** and choose **Active Clusters**. Click the target cluster name to access the cluster details page.

**Step 2** Click **Add Security Group Rule** on the right of **EIP**.

**Step 3** On the **Add Security Group Rule** page, add the IP address segment for users to access the public network and select **I confirm that the authorized object is a trusted public IP address range. Do not use 0.0.0.0/0. Otherwise, security risks may arise.**

By default, the IP address used for accessing the public network is filled. You can change the IP address segment as required. To enable multiple IP address segments, repeat steps [Step 1](#) to [Step 4](#). If you want to view, modify, or delete a security group rule, click **Manage Security Group Rule**.

**Step 4** Click **OK**.

----End

## 29.2.2 Accessing FusionInsight Manager (MRS 3.x or Later)

### Scenario

In MRS 3.x or later, FusionInsight Manager is used to monitor, configure, and manage clusters. After the cluster is installed, you can use the account to log in to FusionInsight Manager.

#### NOTE

If you cannot log in to the WebUI of the component, access FusionInsight Manager by referring to [Accessing FusionInsight Manager from an ECS](#).

### Accessing FusionInsight Manager Using EIP

**Step 1** Log in to the MRS management console.

**Step 2** In the navigation pane, choose **Clusters > Active Clusters**. Click the target cluster name to access the cluster details page.

**Step 3** Click **Manager** next to **MRS Manager**. In the displayed dialog box, configure the EIP information.

1. If no EIP is bound during MRS cluster creation, select an available EIP from the EIP drop-down list or click **Manage EIP** to create an EIP. If you have bound an EIP when creating a cluster, go to [Step 3.2](#).
2. Select the security group to which the security group rule to be added belongs. The security group is configured when the cluster is created.
3. Add a security group rule. By default, the filled-in rule is used to access the EIP. To enable multiple IP address segments to access Manager, see steps [Step 6](#) to [Step 9](#). If you want to view, modify, or delete a security group rule, click **Manage Security Group Rule**.
4. Select the information to be confirmed and click **OK**.

**Step 4** Click **OK**. The Manager login page is displayed.

**Step 5** Enter the default username **admin** and the password set during cluster creation, and click **Log In**. The Manager page is displayed.

**Step 6** On the MRS management console, choose **Clusters > Active Clusters**. Click the target cluster name to access the cluster details page.

#### NOTE

To grant other users the permission to access Manager, perform [Step 6](#) to [Step 9](#) to add the users' public IP addresses to the trusted IP address range.

**Step 7** Click **Add Security Group Rule** on the right of **EIP**.

**Step 8** On the **Add Security Group Rule** page, add the IP address segment for users to access the public network and select **I confirm that *public network IP/port* is a trusted public IP address. I understand that using 0.0.0.0/0. poses security risks.**

By default, the IP address used for accessing the public network is filled. You can change the IP address segment as required. To enable multiple IP address segments, repeat steps [Step 6](#) to [Step 9](#). If you want to view, modify, or delete a security group rule, click **Manage Security Group Rule**.

**Step 9** Click **OK**.

----End

## Accessing FusionInsight Manager from an ECS

**Step 1** On the MRS management console, click **Clusters**.

**Step 2** On the **Active Clusters** page, click the name of the specified cluster.

Record the **AZ, VPC, MRS ManagerSecurity Group** of the cluster.

**Step 3** On the homepage of the management console, choose **Service List > Elastic Cloud Server** to switch to the ECS management console and create an ECS.

- The **AZ, VPC, and Security Group** of the ECS must be the same as those of the cluster to be accessed.
- Select a Windows public image. For example, a standard image **Windows Server 2012 R2 Standard 64bit(40GB)**.
- For details about other configuration parameters, see **Elastic Cloud Server > User Guide > Getting Started > Creating and Logging In to a Windows ECS**.

### NOTE

If the security group of the ECS is different from **Default Security Group** of the Master node, you can modify the configuration using either of the following methods:

- Change the security group of the ECS to the default security group of the Master node. For details, see **Elastic Cloud Server > User Guide > Security Group > Changing a Security Group**.
- Add two security group rules to the security groups of the Master and Core nodes to enable the ECS to access the cluster. Set **Protocol** to **TCP**, **Ports** of the two security group rules to **28443** and **20009**, respectively. For details, see **Virtual Private Cloud > User Guide > Security > Security Group > Adding a Security Group Rule**.

**Step 4** On the VPC management console, apply for an EIP and bind it to the ECS.

For details, see **Virtual Private Cloud > User Guide > Elastic IP > Assigning an EIP and Binding It to an ECS**.

**Step 5** Log in to the ECS.

The Windows system account, password, EIP, and the security group rules are required for logging in to the ECS. For details, see **Elastic Cloud Server > User Guide > Instances > Logging In to a Windows ECS**.

**Step 6** On the Windows remote desktop, use your browser to access Manager.

For example, you can use Internet Explorer 11 in the Windows 2012 OS.

The address for accessing Manager is the address of the **MRS Manager** page. Enter the name and password of the cluster user, for example, user **admin**.

 NOTE

- If you access Manager with other cluster usernames, change the password upon your first access. The new password must meet the requirements of the current password complexity policies. For details, contact the MRS cluster administrator.
- By default, a user is locked after inputting an incorrect password five consecutive times. The user is automatically unlocked after 5 minutes.

----End

## 29.3 Using an MRS Client

### 29.3.1 Installing a Client (Version 3.x or Later)

#### Scenario

This section describes how to install clients of all services (excluding Flume) in an MRS cluster. For details about how to install the Flume client, see [Installing the Flume Client](#).

A client can be installed on a node inside or outside the cluster. This section uses the installation directory `/opt/hadoopclient` as an example. Replace it to the actual one.

#### Prerequisites

- A Linux ECS has been prepared. For details about the supported OS of the ECS, see [Table 29-1](#).

**Table 29-1** Reference list

CPU Architecture	OS	Supported Version
x86 computing	Euler	Euler OS 2.5
	SuSE	SUSE Linux Enterprise Server 12 SP4 (SUSE 12.4)
	RedHat	Red Hat-7.5-x86_64 (Red Hat 7.5)
	CentOS	CentOS 7.6
Kunpeng computing (Arm)	Euler	Euler OS 2.8
	CentOS	CentOS 7.6

In addition, sufficient disk space is allocated for the ECS, for example, 40 GB.

- The ECS and the MRS cluster are in the same VPC.

- The security group of the ECS must be the same as that of the master node in the MRS cluster.
- The NTP service has been installed on the ECS OS and is running properly. If the NTP service is not installed, run the **yum install ntp -y** command to install it when the **yum** source is configured.
- A user can log in to the Linux ECS using the password (in SSH mode).

## Installing a Client on a Node Inside a Cluster

1. Obtain the software package.

Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Click the name of the cluster to be operated in the **Cluster** drop-down list.

Choose **More > Download Client**. The **Download Cluster Client** dialog box is displayed.

### NOTE

In the scenario where only one client is to be installed, choose **Cluster > Service > Service name > More > Download Client**. The **Download Client** dialog box is displayed.

2. Set the client type to **Complete Client**.

**Configuration Files Only** is to download client configuration files in the following scenario: After a complete client is downloaded and installed and modify server configurations on Manager, developers need to update the configuration files during application development.

The platform type can be set to **x86\_64** or **aarch64**.

- **x86\_64**: indicates the client software package that can be deployed on the x86 servers.
- **aarch64**: indicates the client software package that can be deployed on the TaiShan servers.

### NOTE

The cluster supports two types of clients: **x86\_64** and **aarch64**. The client type must match the architecture of the node for installing the client. Otherwise, client installation will fail.

3. Select **Save to Path** and click **OK** to generate the client file.

The generated file is stored in the **/tmp/FusionInsight-Client** directory on the active management node by default. You can also store the client file in a directory on which user **omm** has the read, write, and execute permissions. Copy the software package to the file directory, for example, **/opt/Bigdata/client**, on the server where the client is to be installed as user **omm** or **root**.

The name of the client software package is in the follow format:

**FusionInsight\_Cluster\_<Cluster ID>\_Services\_Client.tar**.

The following steps and sections use

**FusionInsight\_Cluster\_1\_Services\_Client.tar** as an example.



 NOTE

If you cannot obtain the permissions of user **root**, use user **omm**.

To install the client on another node in the cluster, run the following command to copy the client to the node where the client is to be installed:

```
scp -p /tmp/FusionInsight-Client/FusionInsight_Cluster_1_Services_Client.tar IP
address of the node where the client is to be installed:/opt/Bigdata/client
```

4. Log in to the server where the client software package is located as user **user\_client**.
5. Decompress the software package.  
Go to the directory where the installation package is stored, such as **/opt/Bigdata/client**. Run the following command to decompress the installation package to a local directory:  
**tar -xvf FusionInsight\_Cluster\_1\_Services\_Client.tar**
6. Verify the software package.  
Run the following command to verify the decompressed file and check whether the command output is consistent with the information in the **sha256** file.  
**sha256sum -c FusionInsight\_Cluster\_1\_Services\_ClientConfig.tar.sha256**  
FusionInsight\_Cluster\_1\_Services\_ClientConfig.tar: OK
7. Decompress the obtained installation file.  
**tar -xvf FusionInsight\_Cluster\_1\_Services\_ClientConfig.tar**
8. Go to the directory where the installation package is stored, and run the following command to install the client to a specified directory (an absolute path), for example, **/opt/hadoopclient**:  
**cd /opt/Bigdata/client/FusionInsight\_Cluster\_1\_Services\_ClientConfig**  
Run the **./install.sh /opt/hadoopclient** command to install the client. The client is successfully installed if information similar to the following is displayed:

```
The component client is installed successfully
```

 NOTE

- If the clients of all or some services use the **/opt/hadoopclient** directory, other directories must be used when you install other service clients.
- You must delete the client installation directory when uninstalling a client.
- To ensure that an installed client can only be used by the installation user (for example, **user\_client**), add parameter **-o** during the installation. That is, run the **./install.sh /opt/hadoopclient -o** command to install the client.
- If an HBase client is installed, it is recommended that the client installation directory contain only uppercase and lowercase letters, digits, and characters (**\_-?.@+=**) due to the limitation of the Ruby syntax used by HBase.

## Using a Client

1. On the node where the client is installed, run the **sudo su - omm** command to switch the user. Run the following command to go to the client directory:  
**cd /opt/hadoopclient**
2. Run the following command to configure environment variables:  
**source bigdata\_env**

3. If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step.

**kinit** *MRS cluster user*

Example: **kinit admin**

 **NOTE**

User **admin** is created by default for MRS clusters with Kerberos authentication enabled and is used for administrators to maintain the clusters.

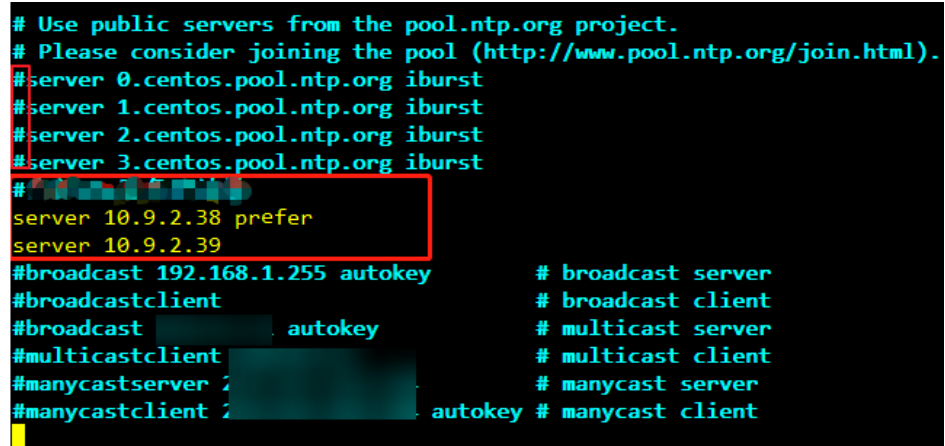
4. Run the client command of a component directly.  
For example, run the **hdfs dfs -ls /** command to view files in the HDFS root directory.

## Installing a Client on a Node Outside a Cluster

1. Create an ECS that meets the requirements in [Prerequisites](#).
2. Perform NTP time synchronization to synchronize the time of nodes outside the cluster with that of the MRS cluster.
  - a. Run the **vi /etc/ntp.conf** command to edit the NTP client configuration file, add the IP addresses of the master node in the MRS cluster, and comment out the IP address of other servers.

```
server master1_ip prefer
server master2_ip
```

**Figure 29-1** Adding the master node IP addresses



```
Use public servers from the pool.ntp.org project.
Please consider joining the pool (http://www.pool.ntp.org/join.html).
#server 0.centos.pool.ntp.org iburst
#server 1.centos.pool.ntp.org iburst
#server 2.centos.pool.ntp.org iburst
#server 3.centos.pool.ntp.org iburst
#server 4.centos.pool.ntp.org iburst
server 10.9.2.38 prefer
server 10.9.2.39
#broadcast 192.168.1.255 autokey # broadcast server
#broadcastclient # broadcast client
#broadcast [redacted] autokey # multicast server
#multicastclient # multicast client
#manycastserver [redacted] # manycast server
#manycastclient [redacted] autokey # manycast client
```

- b. Run the **service ntpd stop** command to stop the NTP service.
  - c. Run the **/usr/sbin/ntpdate IP address of the active master node** command to manually synchronize time.
  - d. Run the **service ntpd start** or **systemctl restart ntpd** command to start the NTP service.
  - e. Run the **ntpstat** command to check the time synchronization result.
3. Perform the following steps to download the cluster client software package from FusionInsight Manager, copy the package to the ECS node, and install the client:
    - a. Log in to FusionInsight Manager and download the cluster client to the specified directory on the active master management node by referring to

[Accessing FusionInsight Manager \(MRS 3.x or Later\)](#) and [Installing a Client on a Node Inside a Cluster](#).

- b. Run the following command to log in to the active management node as user **root**:  
**sudo su - omm**
- c. Run the following command to copy the client to the node where the client is to be installed:  
**scp -p /tmp/FusionInsight-Client/  
FusionInsight\_Cluster\_1\_Services\_Client.tar IP address of the node  
where the client is to be installed:/tmp**
- d. Log in to the node on which the client is to be installed as the client user.  
Run the following commands to install the client. If you do not have the file operation permission, change the file permission as user **root**.  
**cd /tmp**  
**tar -xvf FusionInsight\_Cluster\_1\_Services\_Client.tar**  
**tar -xvf FusionInsight\_Cluster\_1\_Services\_ClientConfig.tar**  
**cd /tmp/FusionInsight\_Cluster\_1\_Services\_ClientConfig**  
**./install.sh /opt/mrsclient**
- e. Run the following commands to switch to the client directory and configure environment variables:  
**cd /opt/mrsclient**  
**source bigdata\_env**
- f. If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step.  
**kinit MRS cluster user**  
Example: **kinit admin**
- g. Run the client command of a component directly.  
For example, run the **hdfs dfs -ls /** command to view files in the HDFS root directory.

## 29.3.2 Installing a Client (Versions Earlier Than 3.x)

### Scenario

An MRS client is required. The MRS cluster client can be installed on the Master or Core node in the cluster or on a node outside the cluster.

After a cluster of versions earlier than MRS 3.x is created, a client is installed on the active Master node by default. You can directly use the client. The installation directory is **/opt/client**.

For details about how to install a client of MRS 3.x or later, see [Installing a Client \(Version 3.x or Later\)](#).

 **NOTE**

If a client has been installed on the node outside the MRS cluster and the client only needs to be updated, update the client using the user who installed the client, for example, user **root**.

## Prerequisites

- An ECS has been prepared. For details about the OS and its version of the ECS, see [Table 29-2](#).

**Table 29-2** Reference list

OS	Supported Version
EulerOS	<ul style="list-style-type: none"><li>• Available: EulerOS 2.2</li><li>• Available: EulerOS 2.3</li><li>• Available: EulerOS 2.5</li></ul>

For example, a user can select an ECS running the EulerOS.

In addition, sufficient disk space is allocated for the ECS, for example, 40 GB.

- The ECS and the MRS cluster are in the same VPC.
- The security group of the ECS is the same as that of the Master node of the MRS cluster.

If this requirement is not met, modify the ECS security group or configure the inbound and outbound rules of the ECS security group to allow the ECS security group to be accessed by all security groups of MRS cluster nodes.

- To enable users to log in to a Linux ECS using a password (SSH), see *Instances > Logging In to a Linux ECS > Login Using an SSH Password in the Elastic Cloud Server User Guide*.

## Installing a Client on the Core Node

1. Log in to MRS Manager and choose **Services > Download Client** to download the client installation package to the active management node.

 **NOTE**

If only the client configuration file needs to be updated, see method 2 in [Updating a Client \(Versions Earlier Than 3.x\)](#).

2. Use the IP address to search for the active management node, and log in to the active management node using VNC.
3. Log in to the active management node, and run the following command to switch the user:

**sudo su - omm**

4. On the MRS management console, view the IP address on the **Nodes** tab page of the specified cluster.

Record the IP address of the Core node where the client is to be used.

5. On the active management node, run the following command to copy the client installation package to the Core node:

```
scp -p /tmp/MRS-client/MRS_Services_Client.tar IP address of the Core node:/opt/client
```

6. Log in to the Core node as user **root**.  
Master nodes support Cloud-Init. The preset username for Cloud-Init is **root** and the password is the one you set during cluster creation.

7. Run the following commands to install the client:

```
cd /opt/client
tar -xvf MRS_Services_Client.tar
tar -xvf MRS_Services_ClientConfig.tar
cd /opt/client/MRS_Services_ClientConfig
./install.sh Client installation directory
```

For example, run the following command:

```
./install.sh /opt/client
```

8. For details about how to use the client, see [Using an MRS Client](#).

## Using an MRS Client

1. On the node where the client is installed, run the **sudo su - omm** command to switch the user. Run the following command to go to the client directory:

```
cd /opt/client
```

2. Run the following command to configure environment variables:

```
source bigdata_env
```

3. If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit MRS cluster user
```

Example: **kinit admin**

### NOTE

User **admin** is created by default for MRS clusters with Kerberos authentication enabled and is used for administrators to maintain the clusters.

4. Run the client command of a component directly.

For example, run the **hdfs dfs -ls /** command to view files in the HDFS root directory.

## Installing a Client on a Node Outside the Cluster

**Step 1** Create an ECS that meets the requirements in the prerequisites.

**Step 2** Log in to MRS Manager. For details, see [Accessing MRS Manager \(Versions Earlier Than MRS 3.x\)](#). Then, choose **Services**.

**Step 3** Click **Download Client**.

**Step 4** In **Client Type**, select **All client files**.

**Step 5** In **Download To**, select **Remote host**.

**Step 6** Set **Host IP Address** to the IP address of the ECS, **Host Port** to **22**, and **Save Path** to **/tmp**.

- If the default port **22** for logging in to an ECS using SSH has been changed, set **Host Port** to the new port.
- **Save Path** contains a maximum of 256 characters.

**Step 7** Set **Login User** to **root**.

If other users are used, ensure that the users have read, write, and execute permission on the save path.

**Step 8** Select **Password** or **SSH Private Key** for **Login Mode**.

- **Password**: Enter the password of user **root** set during cluster creation.
- **SSH Private Key**: Select and upload the key file used for creating the cluster.

**Step 9** Click **OK** to generate a client file.

If the following information is displayed, the client package is saved. Click **Close**. Obtain the client file from the save path on the remote host that is set when the client is downloaded.

Client files downloaded to the remote host successfully.

If the following information is displayed, check the username, password, and security group configurations of the remote host. Ensure that the username and password are correct and an inbound rule of the SSH (22) port has been added to the security group of the remote host. And then, go to [Step 2](#) to download the client again.

Failed to connect to the server. Please check the network connection or parameter settings.

 **NOTE**

Generating a client will occupy a large number of disk I/Os. You are advised not to download a client when the cluster is being installed, started, and patched, or in other unstable states.

**Step 10** Log in to the ECS using VNC. For details, see **Instance > Logging In to a Linux > Logging In to a Linux** in the *Elastic Cloud Server User Guide*

All images support Cloud-Init. The preset username for Cloud-Init is **root** and the password is the one you set during cluster creation. It is recommended that you change the password upon the first login.

**Step 11** Perform NTP time synchronization to synchronize the time of nodes outside the cluster with the time of the MRS cluster.

1. Check whether the NTP service is installed. If it is not installed, run the **yum install ntp -y** command to install it.
2. Run the **vim /etc/ntp.conf** command to edit the NTP client configuration file, add the IP address of the Master node in the MRS cluster, and comment out the IP addresses of other servers.

```
server master1_ip prefer
server master2_ip
```

Figure 29-2 Adding the master node IP addresses

```
Use public servers from the pool.ntp.org project.
Please consider joining the pool (http://www.pool.ntp.org/join.html).
#server 0.centos.pool.ntp.org iburst
#server 1.centos.pool.ntp.org iburst
#server 2.centos.pool.ntp.org iburst
#server 3.centos.pool.ntp.org iburst
#server 4.centos.pool.ntp.org iburst
server 10.9.2.38 prefer
server 10.9.2.39
#broadcast 192.168.1.255 autokey # broadcast server
#broadcastclient # broadcast client
#broadcast autokey # multicast server
#multicastclient # multicast client
#manycastserver # manycast server
#manycastclient autokey # manycast client
```

3. Run the **service ntpd stop** command to stop the NTP service.
4. Run the **/usr/sbin/ntpdate IP address of the active Master node** command to manually synchronize the time.
5. Run the **service ntpd start** or **systemctl restart ntpd** command to start the NTP service.
6. Run the **ntpstat** command to check the time synchronization result:

**Step 12** On the ECS, switch to user **root** and copy the installation package in **Save Path** in **Step 6** to the **/opt** directory. For example, if **Save Path** is set to **/tmp**, run the following commands:

```
sudo su - root
cp /tmp/MRS_Services_Client.tar /opt
```

**Step 13** Run the following command in the **/opt** directory to decompress the package and obtain the verification file and the configuration package of the client:

```
tar -xvf MRS_Services_Client.tar
```

**Step 14** Run the following command to verify the configuration file package of the client:

```
sha256sum -c MRS_Services_ClientConfig.tar.sha256
```

The command output is as follows:

```
MRS_Services_ClientConfig.tar: OK
```

**Step 15** Run the following command to decompress **MRS\_Services\_ClientConfig.tar**:

```
tar -xvf MRS_Services_ClientConfig.tar
```

**Step 16** Run the following command to install the client to a new directory, for example, **/opt/Bigdata/client**. A directory is automatically generated during the client installation.

```
sh /opt/MRS_Services_ClientConfig/install.sh /opt/Bigdata/client
```

If the following information is displayed, the client has been successfully installed:

```
Components client installation is complete.
```

**Step 17** Check whether the IP address of the ECS node is connected to the IP address of the cluster Master node.

For example, run the following command: **ping** *Master node IP address*.

- If yes, go to [Step 18](#).
- If no, check whether the VPC and security group are correct and whether the ECS and the MRS cluster are in the same VPC and security group, and go to [Step 18](#).

**Step 18** Run the following command to configure environment variables:

```
source /opt/Bigdata/client/bigdata_env
```

**Step 19** If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit MRS cluster user
```

Example: **kinit admin**

**Step 20** Run the client command of a component.

For example, run the following command to query the HDFS directory:

```
hdfs dfs -ls /
```

----End

### 29.3.3 Updating a Client (Version 3.x or Later)

A cluster provides a client for you to connect to a server, view task results, or manage data. If you modify service configuration parameters on Manager and restart the service, you need to download and install the client again or use the configuration file to update the client.

#### Updating the Client Configuration

**Method 1:**

**Step 1** Log in to FusionInsight Manager. For details, see [Accessing FusionInsight Manager \(MRS 3.x or Later\)](#). Click the name of the cluster to be operated in the **Cluster** drop-down list.

**Step 2** Choose **More > Download Client > Configuration Files Only**.

The generated compressed file contains the configuration files of all services.

**Step 3** Determine whether to generate a configuration file on the cluster node.

- If yes, select **Save to Path**, and click **OK** to generate the client file. By default, the client file is generated in **/tmp/FusionInsight-Client** on the active management node. You can also store the client file in other directories, and user **omm** has the read, write, and execute permissions on the directories. Then go to [Step 4](#).
- If no, click **OK**, specify a local save path, and download the complete client. Wait until the download is complete and go to [Step 4](#).

**Step 4** Use WinSCP to save the compressed file to the client installation directory, for example, **/opt/hadoopclient**, as the client installation user.



**Step 5** Decompress the software package.

Run the following commands to go to the directory where the client is installed, and decompress the file to a local directory. For example, the downloaded client file is **FusionInsight\_Cluster\_1\_Services\_Client.tar**.

```
cd /opt/hadoopclient
```

```
tar -xvf FusionInsight_Cluster_1_Services_Client.tar
```

**Step 6** Verify the software package.

Run the following command to verify the decompressed file and check whether the command output is consistent with the information in the **sha256** file.

```
sha256sum -c
FusionInsight_Cluster_1_Services_ClientConfig_ConfigFiles.tar.sha256
```

```
FusionInsight_Cluster_1_Services_ClientConfig_ConfigFiles.tar: OK
```

**Step 7** Decompress the package to obtain the configuration file.

```
tar -xvf FusionInsight_Cluster_1_Services_ClientConfig_ConfigFiles.tar
```

**Step 8** Run the following command in the client installation directory to update the client using the configuration file:

```
sh refreshConfig.sh Client installation directory Directory where the configuration file is located
```

For example, run the following command:

```
sh refreshConfig.sh /opt/hadoopclient /opt/hadoopclient/
FusionInsight_Cluster_1_Services_ClientConfig_ConfigFiles
```

If the following information is displayed, the configurations have been updated successfully.

```
Succeed to refresh components client config.
```

```
----End
```

**Method 2:**

**Step 1** Log in to the client installation node as user **root**.

**Step 2** Go to the client installation directory, for example, **/opt/hadoopclient** and run the following commands to update the configuration file:

```
cd /opt/hadoopclient
```

```
sh autoRefreshConfig.sh
```

**Step 3** Enter the username and password of the FusionInsight Manager administrator and the floating IP address of FusionInsight Manager.

**Step 4** Enter the names of the components whose configuration needs to be updated. Use commas (,) to separate the component names. Press **Enter** to update the configurations of all components if necessary.

If the following information is displayed, the configurations have been updated successfully.

Succeed to refresh components client config.

----End

## 29.3.4 Updating a Client (Versions Earlier Than 3.x)

### NOTE

This section applies to clusters of versions earlier than MRS 3.x. For MRS 3.x or later, see [Updating a Client \(Version 3.x or Later\)](#).

## Updating a Client Configuration File

### Scenario

An MRS cluster provides a client for you to connect to a server, view task results, or manage data. Before using an MRS client, you need to download and update the client configuration file if service configuration parameters are modified and a service is restarted or the service is merely restarted on MRS Manager.

During cluster creation, the original client is stored in the **/opt/client** directory on all nodes in the cluster by default. After the cluster is created, only the client of a Master node can be directly used. To use the client of a Core node, you need to update the client configuration file first.

### Procedure

#### Method 1:

**Step 1** Log in to MRS Manager. For details, see [Accessing MRS Manager \(Versions Earlier Than MRS 3.x\)](#). Then, choose **Services**.

**Step 2** Click **Download Client**.

Set **Client Type** to **Only configuration files**, **Download To** to **Server**, and click **OK** to generate the client configuration file. The generated file is saved in the **/tmp/MRS-client** directory on the active management node by default. You can customize the file path.

**Step 3** Query and log in to the active Master node.

**Step 4** If you use the client in the cluster, run the following command to switch to user **omm**. If you use the client outside the cluster, switch to user **root**.

```
sudo su - omm
```

**Step 5** Run the following command to switch to the client directory, for example, **/opt/Bigdata/client**:

```
cd /opt/Bigdata/client
```

**Step 6** Run the following command to update client configurations:

```
sh refreshConfig.sh Client installation directory Full path of the client configuration file package
```

For example, run the following command:

```
sh refreshConfig.sh /opt/Bigdata/client /tmp/MRS-client/
MRS_Services_Client.tar
```

If the following information is displayed, the configurations have been updated successfully.

```
ReFresh components client config is complete.
Succeed to refresh components client config.
```

----End

### Method 2:

**Step 1** After the cluster is installed, run the following command to switch to user **omm**. If you use the client outside the cluster, switch to user **root**.

```
sudo su - omm
```

**Step 2** Run the following command to switch to the client directory, for example, **/opt/Bigdata/client**:

```
cd /opt/Bigdata/client
```

**Step 3** Run the following command and enter the name of an MRS Manager user with the download permission and its password (for example, the username is **admin** and the password is the one set during cluster creation) as prompted to update client configurations.

```
sh autoRefreshConfig.sh
```

**Step 4** After the command is executed, the following information is displayed, where **XXX** indicates the name of the component installed in the cluster. To update client configurations of all components, press **Enter**. To update client configurations of some components, enter the component names and separate them with commas (,).

```
Components "xxx" have been installed in the cluster. Please input the comma-separated names of the
components for which you want to update client configurations. If you press Enter without inputting any
component name, the client configurations of all components will be updated:
```

If the following information is displayed, the configurations have been updated successfully.

```
Succeed to refresh components client config.
```

If the following information is displayed, the username or password is incorrect.

```
login manager failed,Incorrect username or password.
```

### NOTE

- This script automatically connects to the cluster and invokes the **refreshConfig.sh** script to download and update the client configuration file.
- By default, the client uses the floating IP address specified by **wsom=xxx** in the **Version** file in the installation directory to update the client configurations. To update the configuration file of another cluster, modify the value of **wsom=xxx** in the **Version** file to the floating IP address of the corresponding cluster before performing this step.

----End

## Fully Updating the Original Client of the Active Master Node

### Scenario

During cluster creation, the original client is stored in the `/opt/client` directory on all nodes in the cluster by default. The following uses `/opt/Bigdata/client` as an example.

- For a normal MRS cluster, you will use the pre-installed client on a Master node to submit a job on the management console page.
- You can also use the pre-installed client on the Master node to connect to a server, view task results, and manage data.

After installing the patch on the cluster, you need to update the client on the Master node to ensure that the functions of the built-in client are available.

### Procedure

**Step 1** Log in to MRS Manager. For details, see [Accessing MRS Manager \(Versions Earlier Than MRS 3.x\)](#). Then, choose **Services**.

**Step 2** Click **Download Client**.

Set **Client Type** to **All client files**, **Download To** to **Server**, and click **OK** to generate the client configuration file. The generated file is saved in the `/tmp/MRS-client` directory on the active management node by default. You can customize the file path.

**Step 3** Query and log in to the active Master node.

**Step 4** On the ECS, switch to user **root** and copy the installation package to the `/opt` directory.

```
sudo su - root
```

```
cp /tmp/MRS-client/MRS_Services_Client.tar /opt
```

**Step 5** Run the following command in the `/opt` directory to decompress the package and obtain the verification file and the configuration package of the client:

```
tar -xvf MRS_Services_Client.tar
```

**Step 6** Run the following command to verify the configuration file package of the client:

```
sha256sum -c MRS_Services_ClientConfig.tar.sha256
```

The command output is as follows:

```
MRS_Services_ClientConfig.tar: OK
```

**Step 7** Run the following command to decompress `MRS_Services_ClientConfig.tar`:

```
tar -xvf MRS_Services_ClientConfig.tar
```

**Step 8** Run the following command to move the original client to the `/opt/Bigdata/client_bak` directory:

```
mv /opt/Bigdata/client /opt/Bigdata/client_bak
```

**Step 9** Run the following command to install the client in a new directory. The client path must be `/opt/Bigdata/client`.

```
sh /opt/MRS_Services_ClientConfig/install.sh /opt/Bigdata/client
```

If the following information is displayed, the client has been successfully installed:

Components client installation is complete.

**Step 10** Run the following command to modify the user and user group of the **/opt/Bigdata/client** directory:

```
chown omm:wheel /opt/Bigdata/client -R
```

**Step 11** Run the following command to configure environment variables:

```
source /opt/Bigdata/client/bigdata_env
```

**Step 12** If Kerberos authentication is enabled for the current cluster, run the following command to authenticate the user. If Kerberos authentication is disabled for the current cluster, skip this step.

```
kinit MRS cluster user
```

Example: **kinit admin**

**Step 13** Run the client command of a component.

For example, run the following command to query the HDFS directory:

```
hdfs dfs -ls /
```

----End

## Fully Updating the Original Client of the Standby Master Node

**Step 1** Repeat [Step 1](#) to [Step 3](#) to log in to the standby Master node, and run the following command to switch to user **omm**:

```
sudo su - omm
```

**Step 2** Run the following command on the standby master node to copy the downloaded client package from the active master node:

```
scp omm@master1 nodeIP address:/tmp/MRS-client/
MRS_Services_Client.tar /tmp/MRS-client/
```

### NOTE

- In this command, **master1** node is the active master node.
- **/tmp/MRS-client/** is an example target directory of the standby master node.

**Step 3** Repeat [Step 4](#) to [Step 13](#) to update the client of the standby Master node.

----End