MapReduce Service

Best Practices

Issue	01
Date	2024-09-10

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Security Declaration

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process.* For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Contents

1 Data Analytics

1.1 Using Spark2x to Analyze IoV Drivers' Driving Behavior

Application Scenarios

The best practices for Huawei Cloud MapReduce Service (MRS) guides you through the basic functions of MRS. This case shows you how to use the Spark2x component of MRS to analyze and collect statistics on driver behaviors and obtain the analysis results.

The raw data in this practice includes information on driver behavior, such as sudden acceleration, sudden braking, neutral coasting, speeding, and fatigue driving. With the Spark2x component, you can analyze and collect statistics on the frequency of these behaviors within a specified time frame.

NOTE

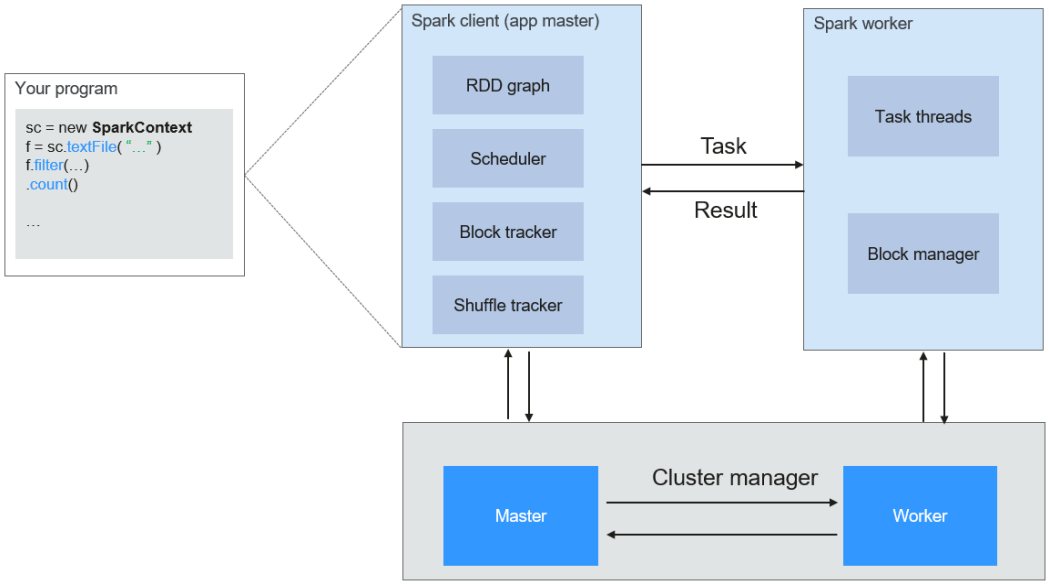
This practice uses MRS 3.1.0 is as an example.You can create a cluster of this version too.

Solution Architecture

Figure 1-1 describes the application running architecture of Spark.

1. An application is running in the cluster as a collection of processes. Driver coordinates the running of applications.
2. To run an application, Driver connects to the cluster manager (such as Standalone, Mesos, and YARN) to apply for the executor resources, and start ExecutorBackend. The cluster manager schedules resources between different applications. Driver schedules DAGs, divides stages, and generates tasks for the application at the same time.
3. Then, Spark sends the codes of the application (the codes transferred to SparkContext, which is defined by JAR or Python) to an executor.
4. After all tasks are finished, the running of the user application is stopped.

Figure 1-1 Spark application running architecture



Procedure

The operation process of this practice is as follows:

1. **Creating an MRS Cluster**: Create an MRS 3.1.0 analysis cluster with Kerberos authentication disabled.
2. **Preparing the Sample Program and Data**: Create an OBS parallel file system and upload the Spark2x sample program and sample data files to the OBS parallel file system.
3. **Creating a Job**: Create and run a SparkSubmit job on the MRS management console.
4. **Viewing the Execution Results**: Obtain the log file from the OBS path and view the execution result.

Creating an MRS Cluster

Step 1 Go to the **Buy Cluster** page.

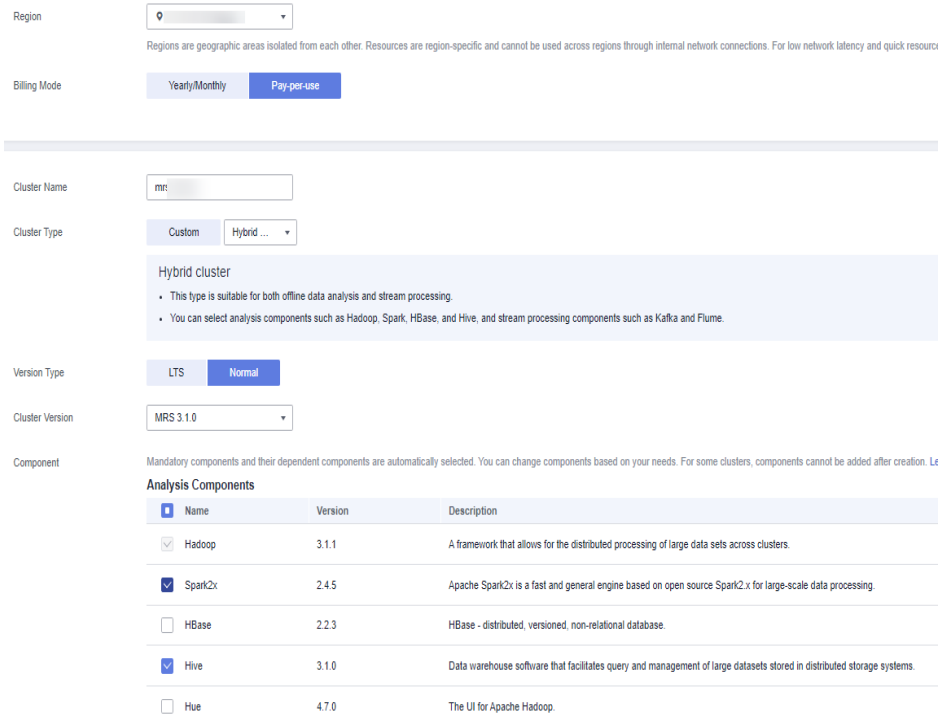
Step 2 Click the **Custom Config** tab.

Configure cluster software information according to **Table 1-1**.

Table 1-1 Software configurations

Parameter	Description	Example Value
Region	Region where the MRS resources belong. MRS clusters in different regions cannot communicate with each other over an intranet. For lower network latency and quick resource access, select the region nearest to you.	**CN-Hong Kong** **NOTE** This document uses **CN-Hong Kong** as an example. If you want to perform operations in other regions, ensure that all operations are performed in the same region.
Billing Mode	Billing mode of the cluster.	**Pay-per-use**
Cluster Name	Name of the MRS cluster.	**mrs_demo**
Cluster Type	Type of the MRS cluster.	**Analysis cluster** (for offline data analysis)
Version Type	Version type of the MRS cluster.	**Normal**
Cluster Version	MRS cluster version.	**MRS 3.1.0** **NOTE** This practice is available for MRS 3.1.0 only.
Component	Components in the MRS cluster.	All components
Metadata	Storage for cluster metadata.	Local

Figure 1-2 Software configurations



Step 3 Click **Next** to configure hardware.

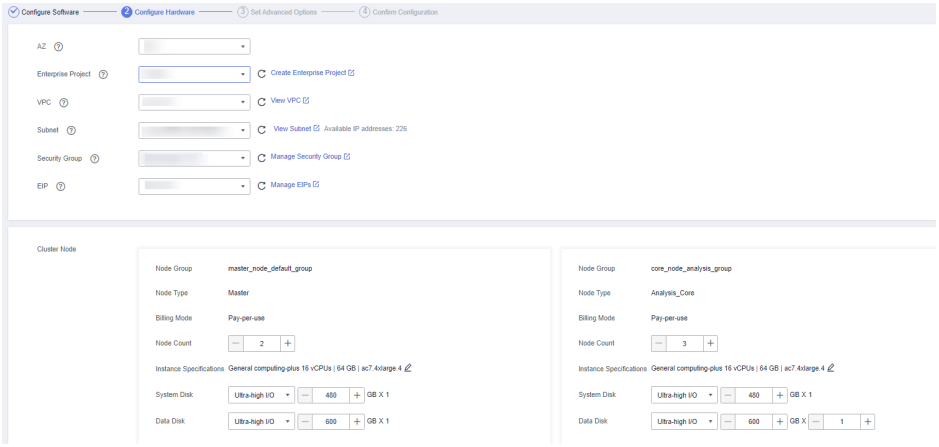
Configure cluster hardware information according to **Table 1-2**.

Table 1-2 Hardware configurations

Parameter	Description	Example Value
AZ	Available AZ associated with the cluster region.	**AZ2**
Enterprise Project	Enterprise project to which the cluster belongs.	default
VPC	VPC where you want to create the cluster. You can click **View VPC** to view the name and ID. If no VPC is available, create one.	xxx
Subnet	Subnet where your cluster belongs. You can access the VPC management console to view the names and IDs of existing subnets in the VPC. If no subnet is created under the VPC, click **Create Subnet** to create one.	xxx

Parameter	Description	Example Value
Security Group	A security group is a set of ECS access rules. It provides access policies for ECSs that have the same security protection requirements and are mutually trusted in a VPC.	Auto create
EIP	An EIP allows you to access the Manager web UI of the cluster.	Bind an EIP.
Cluster Node	Cluster node details.	Default settings

Figure 1-3 Hardware configurations



Step 4 Click **Next**. On the **Set Advanced Options** page, set the following parameters by referring to **Table 1-3** and retain the default settings for other parameters.

Table 1-3 Advanced configurations

Parameter	Description	Example Value
Kerberos Authentication	Whether to enable Kerberos authentication when logging in to Manager.	Disabled
Username	Name of the administrator of MRS Manager. **admin** is used by default.	admin
Password	Password of the MRS Manager administrator.	xxx
Confirm Password	Enter the password of the Manager administrator again.	xxx

Parameter	Description	Example Value
Login Mode	Login method to ECS nodes in the cluster.	Select **Password**.
Username	User for logging in to the ECS. The default value is **root**.	root
Password	Password for logging in to ECSs.	xxx
Confirm Password	Enter the password for logging in to ECSs again.	xxx

Step 5 Click **Next**. On the **Confirm Configuration** page, check the cluster configuration information. If you need to adjust the configuration, click  to go to the corresponding tab page and configure parameters again.

Step 6 Select **Secure Communications** and click **Buy Now**.

Step 7 Click **Back to Cluster List** to view the cluster status.

Cluster creation takes some time. The initial status of the cluster is **Starting**. After the cluster has been created successfully, the cluster status becomes **Running**.

----End

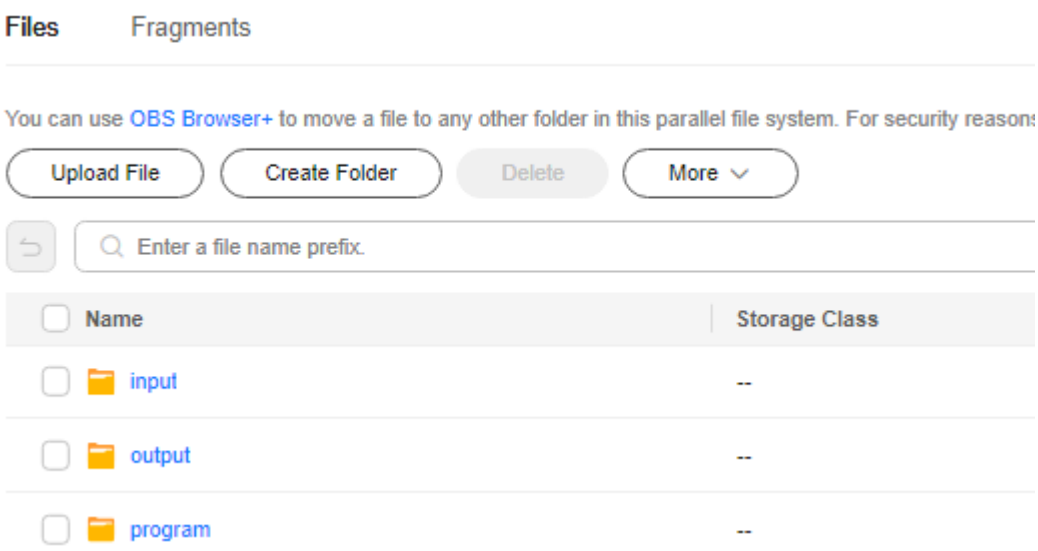
Preparing the Sample Program and Data

Step 1 Create an OBS parallel file system to store the Spark sample program, sample data, job execution results, and logs.

1. Log in to the HUAWEI CLOUD management console.
2. In the **Service List**, choose **Storage** > **Object Storage Service**.
3. In the navigation pane on the left, choose **Parallel File System** and click **Create Parallel File System** to create a file system named **obs-demo-analysis-hwt4**. Retain the default values for parameters such as **Policy**.

Step 2 Click the name of the file system. In the navigation pane on the left, choose **Files**. On the displayed page, click **Create Folder** to create the **program** and **input** folders, as shown in **Figure 1-4**.

Figure 1-4 Creating a folder



Step 3 Download the sample program **driver_behavior.jar** from **https://mrs-obs-ap-southeast-1.obs.ap-southeast-1.myhuaweicloud.com/mrs-demon-samples/demon/driver_behavior.jar** to the local PC.

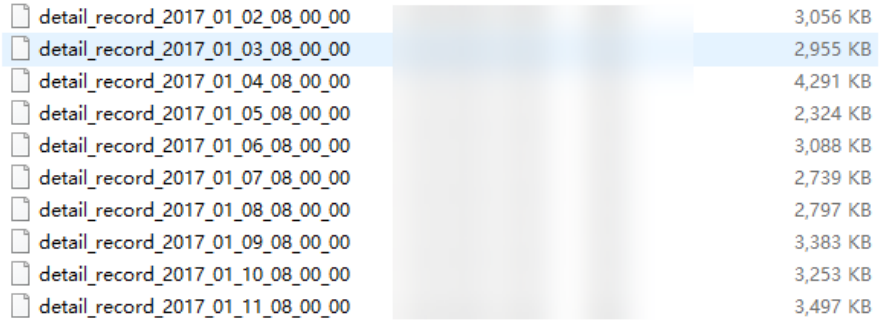
Step 4 Go to the **program** folder. Click **Upload File** and select the local **driver_behavior.jar** sample program.

Step 5 Click **Upload** to upload the sample program to the OBS parallel file system.

Step 6 Obtain Spark sample data from **https://mrs-obs-ap-southeast-1.obs.ap-southeast-1.myhuaweicloud.com/mrs-demon-samples/demon/detail-records.zip**.

Step 7 Decompress the downloaded **detail-records.zip** package to obtain the sample data files.

Figure 1-5 Sample data

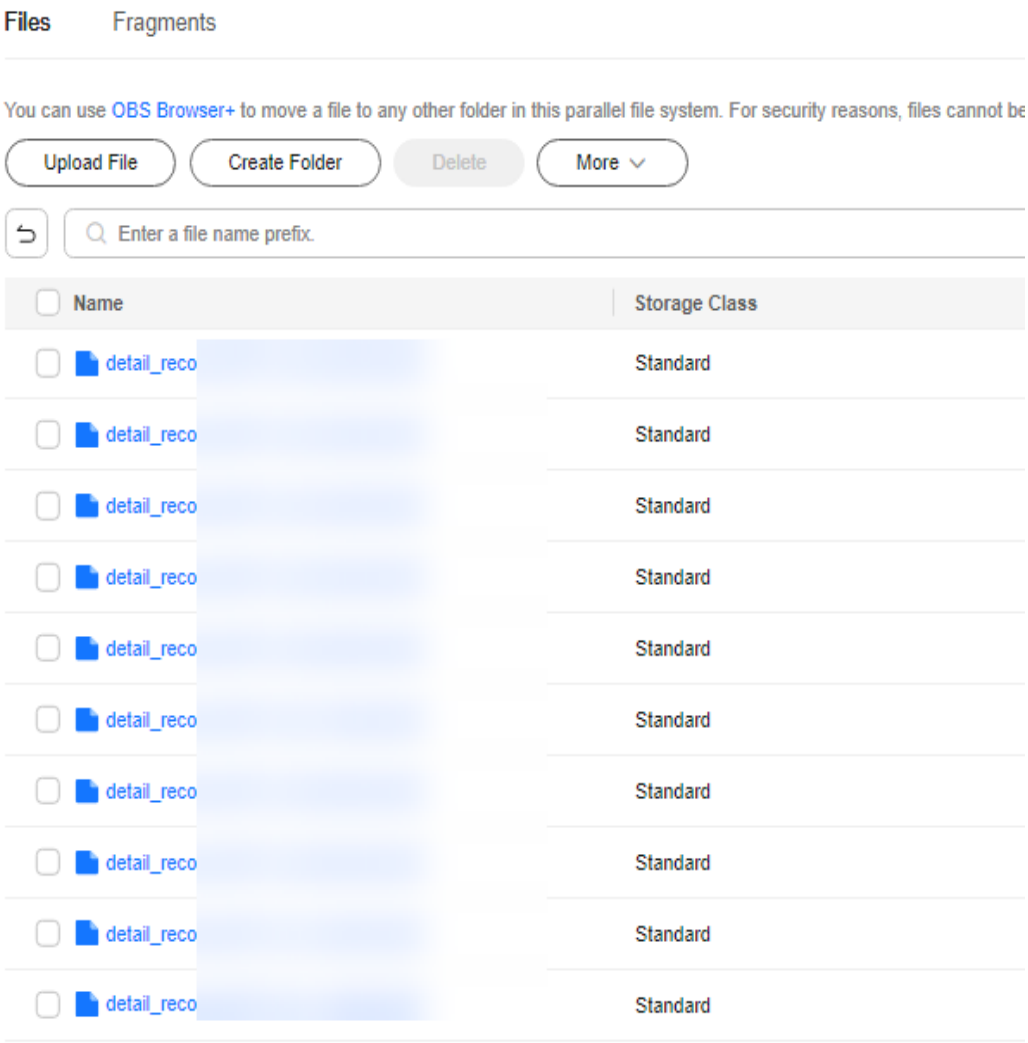


Step 8 Go to the **input** folder. Click **Upload File** and select the local Spark sample data.

Step 9 Click **Upload** to upload the sample data to the OBS parallel file system.

Upload the decompressed data in **Step 7** to the **input** folder.

Figure 1-6 Uploading sample data



----End

Creating a Job

Step 1 Log in to the MRS console, click the **mrs_demo** cluster on the displayed **Active Clusters** page.

Step 2 Click the **Jobs** tab and then **Create** to create a job.

Step 3 Set job parameters.

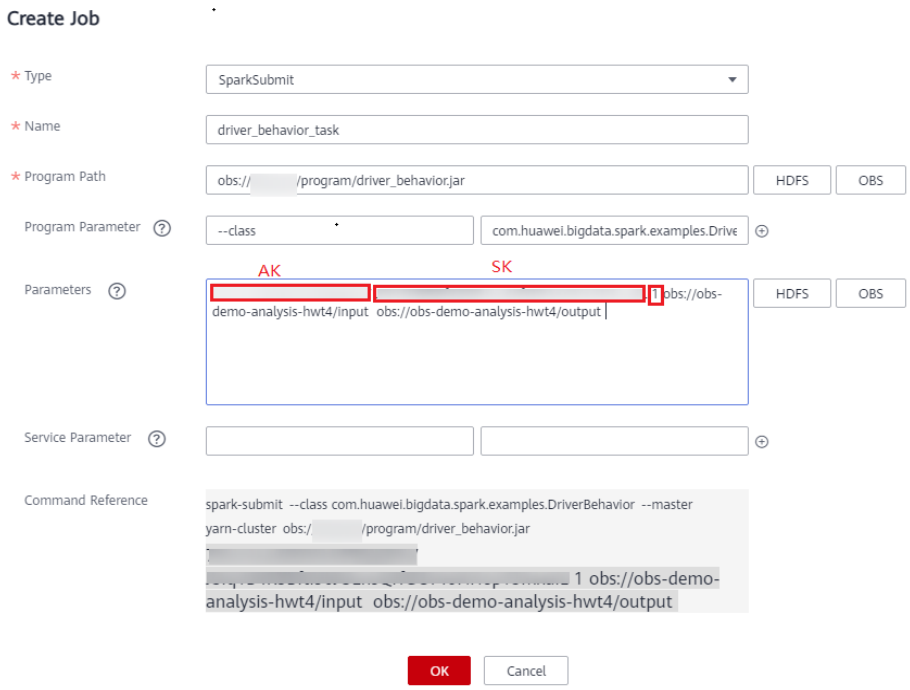
Table 1-4 Configuring job parameters

Parameter	Description	Example Value
Type	Type of the job you want to create.	Select **SparkSubmit**.
Name	Task name.	Enter **driver_behavior_task**.

Parameter	Description	Example Value
Program Path	Path for storing the program package to be executed.	Click **OBS** and select the **driver_behavior.jar** package uploaded in **Preparing the Sample Program and Data**.
Program Parameter	Optimization parameters for resource usage and job execution performance.	Select **--class** in **Parameter**, and enter **com.huawei.bigdata.spark.examples.DriverBehavior** in **Value**.
Parameters	***AK for accessing OBS SK for accessing OBS*** **1** *Input path Output path*. • For details about how to obtain the AK/SK, see the steps described in **NOTE**. • **1** is a fixed input that is used to specify the program function invoked during job execution. • *Input path* is the path you selected for the **Program Path** parameter. • *Output path* should be a directory that does not exist, for example, **obs://obs-demo-analysis-hwt4/output/**. **NOTE** To obtain the AK/SK for accessing OBS, perform the following steps: 1. Log in to the HUAWEI CLOUD management console. 2. Click the username in the upper right corner and choose **My Credentials**. 3. In the navigation pane on the left, choose **Access Keys**. 4. Click **Create Access Key** to add a key. Enter the password and verification code as prompted. The browser automatically downloads the **credentials.csv** file. The file is in CSV format and separated by commas (,). In the file, the middle part is AK and the last part is SK.	***AK information SK information*** **1** *obs://obs-demo-analysis-hwt4/input/ obs://obs-demo-analysis-hwt4/output/*

Parameter	Description	Example Value
Service Parameter	Service parameter modifications of the job to be executed.	This parameter is left blank by default. Retain the default settings.

Figure 1-7 Creating a job



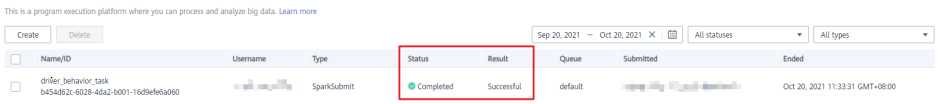
Step 4 Click **OK** to start executing the program.

----End

Viewing the Execution Results

Step 1 Go to the **Jobs** page to view the job execution status.

Figure 1-8 Execution status



Step 2 Wait 1 to 2 minutes and log in to OBS console. Go to the output path of the **obs-demo-analysis-hwt4** file system to view the execution result. Click **Download** in the **Operation** column of the generated CSV file to download the file to your local PC.

Step 3 Open the downloaded CSV file using Excel and classify the data in each column according to the fields defined in the program. The job execution results are obtained.

Figure 1-9 Execution result

Driver ID	License Plate Number	Abrupt Acceleration Times	Abrupt Brake Times	Neutral Sliding Times	Total Neutral Sliding Time	Overspeed Times	Total Overspeed Time	Fatigue Driving Times	Times of Stepping on the Accelerator While Stopping	Oil Leakage Times
shenxian1000004	ADJ750	374	356	297	2810	3126	31494	3767	383	366
xiexiao1000001	AEB132	264	261	248	2525	2324	23434	2720	314	253
xiezhi1000006	A6CU11	255	310	254	2074	2535	23942	2931	312	279
duxu1000009	AT75H8	238	284	247	2632	2301	22338	2814	264	248
hanhui1000002	AZI419	401	444	327	2844	3349	31813	3997	433	371
panxian1000005	AX542C	395	434	330	2930	3531	33946	4307	417	441
zouan1000007	A58M83	360	385	315	2997	3181	31248	3594	389	385
likun1000003	AVM936	341	354	291	3043	3044	28728	3552	347	376
zengpeng1000000	AZQ110	340	344	272	2894	2763	25479	3274	284	337
haowei1000008	A709GB	321	314	255	2659	2639	25522	3204	312	318

----End

1.2 Using Hive to Load HDFS Data and Analyze Book Scores

Application Scenarios

MRS offline processing clusters enable you to analyze and process massive amount of data as well as provide the results for later use.

Offline processing has low requirements on processing time. However, a large amount of data needs to be processed, which occupies a large number of compute and storage resources. Generally, offline processing is implemented through Hive/SparkSQL or MapReduce/Spark2x.

This practice describes how to import and analyze raw data using Hive after you create an MRS cluster and how to implement elastic and low-cost offline big data analysis.

Solution Architecture

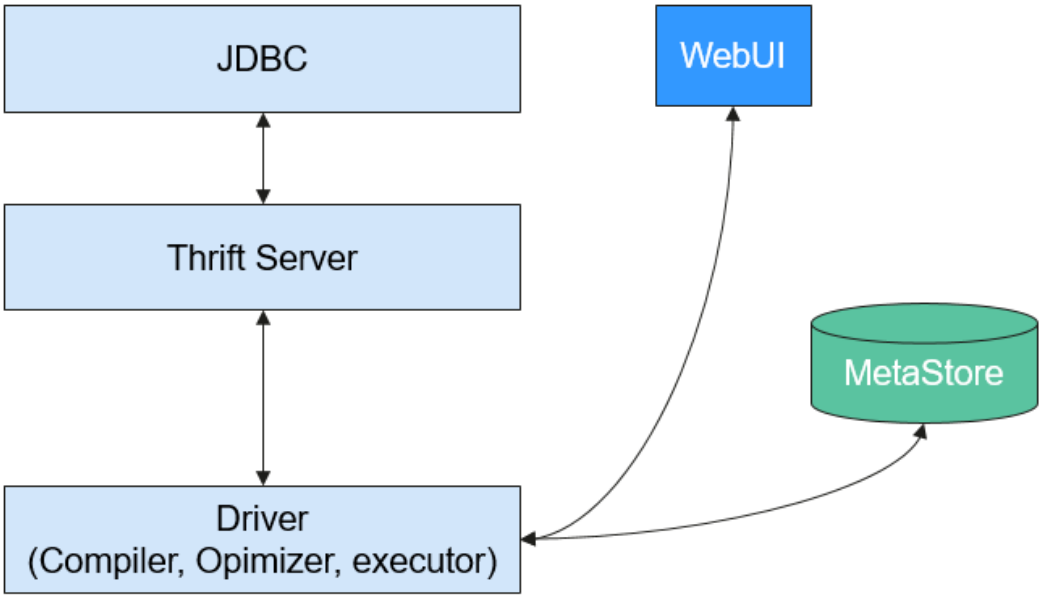
Hive is a data warehouse built on Hadoop. It provides batch computing capability for the big data platform and is able to batch analyze and summarize structured and semi-structured data for data calculation. Hive operates structured data using Hive Query Language (HQL), a SQL-like language. HQL is automatically converted into MapReduce tasks for the query and analysis of massive data in the Hadoop cluster.

Hive is able to:

- Analyze massive structured data and summarizes analysis results.
- Allow complex MapReduce jobs to be compiled in SQL languages.
- Support flexible data storage formats, including JavaScript object notation (JSON), comma separated values (CSV), TextFile, RCFile, SequenceFile, and ORC.

Hive functions as a data warehouse based on HDFS and MapReduce architecture and translates HQL statements into MapReduce jobs or HDFS operations.

Figure 1-10 Hive Architecture



- **Metastore**: reads, writes, and updates metadata such as tables, columns, and partitions. Its lower layer is relational databases.
- **Driver**: manages the lifecycle of HQL execution and participates in the entire Hive job execution.
- **Compiler**: translates HQL statements into a series of interdependent Map or Reduce jobs.
- **Optimizer**: is classified into logical optimizer and physical optimizer to optimize HQL execution plans and MapReduce jobs, respectively.
- **Executor**: runs Map or Reduce jobs based on job dependencies.
- **ThriftServer**: functions as the servers of JDBC, provides Thrift APIs, and integrates with Hive and other applications.
- **Clients**: include the web UI and JDBC APIs and provides APIs for user access.

Procedure

In this practice, we use user comments from a book website as raw data. The data is then imported into a Hive table, where you can run SQL queries to find popular books.

The operation process is as below:

1. **Creating an MRS Offline Query Cluster**
2. **Importing Local Data to HDFS**
3. **Creating a Hive Table**
4. **Importing Raw Data to Hive for Analysis**

Creating an MRS Offline Query Cluster

Step 1 Go to the **Buy Cluster** page.

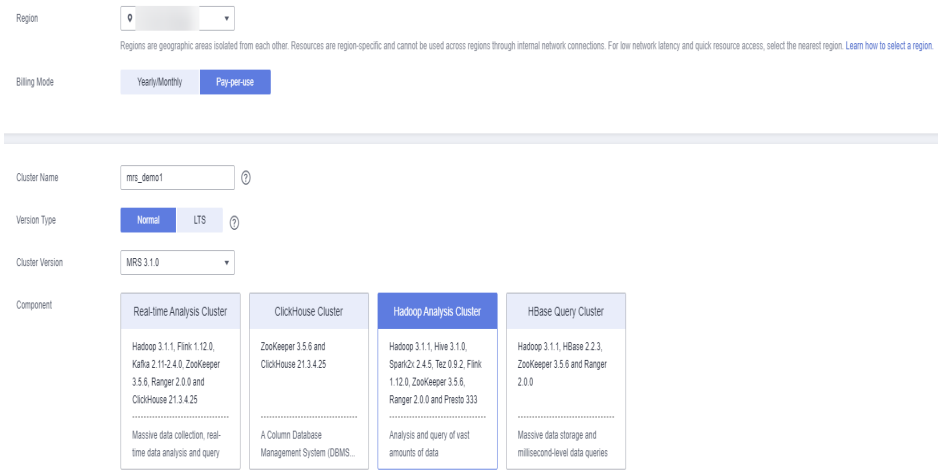
Step 2 Click the **Quick Config** tab and set configuration parameters.

Table 1-5 Software parameters (for reference only)

Parameter	Description	Example Value
Region	Region where the MRS resources belong. MRS clusters in different regions cannot communicate with each other over an intranet. For lower network latency and quick resource access, select the region nearest to you.	CN-Hong Kong
Billing Mode	Billing mode of the cluster.	Pay-per-use
Cluster Name	Name of the MRS cluster.	MRS_demo
Version Type	Version type of the MRS cluster.	Normal
Cluster Version	MRS cluster version.	MRS 3.1.0
Component	Components in the MRS cluster.	Hadoop Analysis Cluster
AZ	Available AZ associated with the cluster region.	AZ1
VPC	VPC where you want to create the cluster. You can click **View VPC** to view the name and ID. If no VPC is available, create one.	vpc-01
Subnet	Subnet where your cluster belongs. You can access the VPC management console to view the names and IDs of existing subnets in the VPC. If no subnet is created under the VPC, click **Create Subnet** to create one.	subnet-01
Enterprise Project	Enterprise project to which the cluster belongs.	default
Kerberos Authentication	Whether to enable Kerberos authentication when logging in to Manager.	Disabled

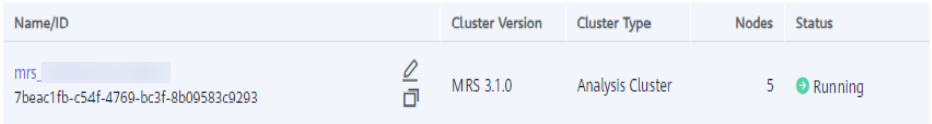
Parameter	Description	Example Value
Username	Name of the administrator of MRS Manager. **admin** is used by default.	admin/root
Password	Password of the MRS Manager administrator.	Set the password for logging in to the cluster management page and ECS node, for example, **Test!@12345**.
Confirm Password	Enter the password of the Manager administrator again.	Enter the password again.
Secure Communications	If the secure communications function is not enabled, MRS clusters cannot be created.	Select **Enable**.

Figure 1-11 Buying a Hadoop analysis cluster



Step 3 Click **Buy Now** and wait until the MRS cluster is created.

Figure 1-12 Cluster purchased



----End

Importing Local Data to HDFS

Step 1 Obtain the book comments file **book_score.txt** from the background of the book website and save it on the local host.

The file contains the following fields: user ID, book ID, book score, and remarks.

Some data is as follows:

```
202001,242,3,Good!
202002,302,3,Test.
202003,377,1,Bad!
220204,51,2,Bad!
202005,346,1,aaa
202006,474,4,None
202007,265,2,Bad!
202008,465,5,Good!
202009,451,3,Bad!
202010,86,3,Bad!
202011,257,2,Bad!
202012,465,4,Good!
202013,465,4,Good!
202014,465,4,Good!
202015,302,5,Good!
202016,302,3,Good!
...
```

Step 2 Log in to OBS Console, click **Parallel File Systems** in the navigation pane. On the displayed page, click **Create Parallel File System**, set the following parameters, and click **Create Now**.

Table 1-6 Parallel file system parameters

Parameter	Description	Example Value
Region	Geographic area where a bucket resides.	CN-Hong Kong
Data Redundancy Policy	• **Multi-AZ storage**: Data is stored in multiple AZs to achieve higher reliability. • **Single-AZ storage**: Data is stored in a single AZ, with lower costs.	Single-AZ storage
File System Name	Name of a file system, which must be globally unique.	mrs-hive
Policy	Read and write permission control for the file system.	Private
Direct Reading	Direct reading allows you to download objects from the Archive storage class without restoring them in advance.	Disable
Enterprise Project	Enterprise project where your bucket belongs, which facilitates unified management.	default
Tags	(Optional) Tags used to identify and classify buckets in OBS.	-

Click the name of the created file system and click **Upload File** in the **Files** tab to upload the data file to the OBS parallel file system.

Step 3 Switch back to the MRS console and click the name of the created MRS cluster. On the **Dashboard** page, click **Synchronize** next to **IAM User Sync**. The synchronization takes about five minutes.

Figure 1-13 Synchronizing IAM users

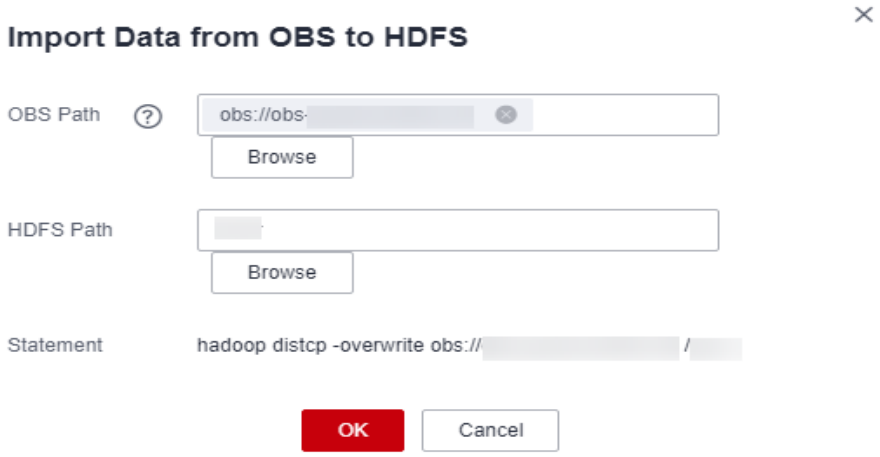
Step 4 Upload the data file to the HDFS.

1. On the **Files** page, click the **HDFS File List** and go to the data storage directory, for example, **/tmp/test**.

   The **/tmp/test** directory is only an example. You can use any directory on the page or create a new one.

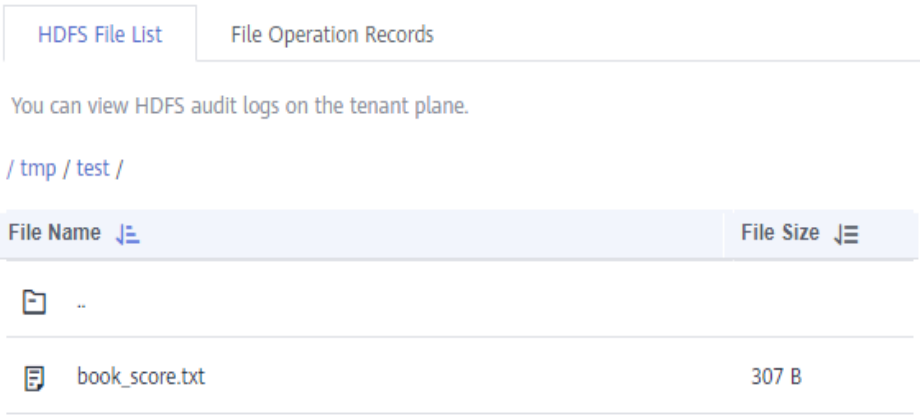
2. Click **Import Data**.
   - **OBS Path**: Select the name of the created OBS parallel file system, find the **book_score.txt** file, select **I confirm that the selected script is secure, and I understand the potential risks and accept the possible exceptions or impacts on the cluster**, and click **OK**.
   - **HDFS Path**: Select the **/tmp/test** directory and click **OK**.

   Figure 1-14 Importing data from OBS to HDFS



3. Click **OK**. After the data is imported, the data file has been uploaded to HDFS of the MRS cluster.

Figure 1-15 Data imported



----End

Creating a Hive Table

Step 1 Download the cluster client, and install it, for example, in the **/opt/client** directory of the active master node. For details, see **Installing a Client**.

You can also use the cluster client provided in the **/opt/Bigdata/client** directory of the master node.

Step 2 Bind an EIP to the active Master node and allow port 22 in the security group. Log in to the active Master node as user **root**, go to the directory where the client is located, and load variables.

cd /opt/client

source bigdata_env

Step 3 Run the **beeline -n'hdfs'** command to go to the Hive Beeline page.

Run the following command to create a Hive table whose fields match the raw data fields:

create table bookscore (userid int,bookid int,score int,remarks string) row format delimited fields terminated by ','stored as textfile;

Step 4 Run the following command to check whether the table is successfully created:

show tables;

```
+------------+
| tab_name  |
+------------+
| bookscore  |
+------------+
```

----End

Importing Raw Data to Hive for Analysis

Step 1 Run the following command on Hive Beeline to import the raw data that has been imported to HDFS to the Hive table:

load data inpath '/tmp/test/book_score.txt' into table bookscore;

Step 2 After data is imported, run the following command to view content in the Hive table:

select * from bookscore;

```
+-------------------+-------------------+-------------------+--------------------+
| bookscore.userid  | bookscore.bookid  | bookscore.score  | bookscore.remarks  |
+-------------------+-------------------+-------------------+--------------------+
| 202001            | 242               | 3                | Good!              |
| 202002            | 302               | 3                | Test.              |
| 202003            | 377               | 1                | Bad!               |
| 220204            | 51                | 2                | Bad!               |
| 202005            | 346               | 1                | aaa                |
| 202006            | 474               | 4                | None               |
| 202007            | 265               | 2                | Bad!               |
| 202008            | 465               | 5                | Good!              |
| 202009            | 451               | 3                | Bad!               |
| 202010            | 86                | 3                | Bad!               |
| 202011            | 257               | 2                | Bad!               |
| 202012            | 465               | 4                | Good!              |
| 202013            | 465               | 4                | Good!              |
| 202014            | 465               | 4                | Good!              |
| 202015            | 302               | 5                | Good!              |
| 202016            | 302               | 3                | Good!              |
...
```

Run the following command to count the number of rows in the table:

select count(*) from bookscore;

```
+------+
| _c0  |
+------+
| 32   |
+------+
```

Step 3 Run the following command to filter the top 3 books with the highest scores in the raw data after the MapReduce task is complete:

select bookid,sum(score) as summarize from bookscore group by bookid order by summarize desc limit 3;

Finally, the following information is displayed:

```
...
INFO  : 2021-10-14 19:53:42,427 Stage-2 map = 0%,  reduce = 0%
INFO  : 2021-10-14 19:53:49,572 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 2.15 sec
INFO  : 2021-10-14 19:53:56,713 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 4.19 sec
INFO  : MapReduce Total cumulative CPU time: 4 seconds 190 msec
INFO  : Ended Job = job_1634197207682_0025
INFO  : MapReduce Jobs Launched:
INFO  : Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 4.24 sec   HDFS Read: 7872 HDFS Write: 322
SUCCESS
INFO  : Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 4.19 sec   HDFS Read: 5965 HDFS Write: 143
SUCCESS
INFO  : Total MapReduce CPU Time Spent: 8 seconds 430 msec
INFO  : Completed executing
command(queryId=omm_20211014195310_cf669633-5b58-4bd5-9837-73286ea83409); Time taken: 47.388
seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+---------+------------+
| bookid  | summarize  |
+---------+------------+
| 465     | 170        |
| 302     | 110        |
```



```
| 474     | 88         |
+---------+------------+
3 rows selected (47.469 seconds)
```

The books whose IDs are 456, 302, and 474 are the top 3 books with the highest scores.

----End

1.3 Using Hive to Load OBS Data and Analyze Enterprise Employee Information

Application Scenarios

MRS Hadoop analysis cluster provides Hive and Spark for storing, computing, and querying massive amounts of offline as well as distributed data.

This practice describes how to import and analyze raw data stored in OBS using Hive after you create an MRS cluster and how to implement elastic and low-cost big data analysis based on storage-compute decoupling.

In this practice, the raw data of employee information includes the following two tables:

Table 1-7 Employee information

ID	Name	Salary Currency	Salary	Tax Category	Work Place	Hire Date
1	Wang	R	8000.01	personal income tax&0.05	China:Shenzhen	2014
3	Tom	D	12000.02	personal income tax&0.09	America:NewYork	2014
4	Jack	D	24000.03	personal income tax&0.09	America:Manhattan	2015
6	Linda	D	36000.04	personal income tax&0.09	America:NewYork	2014
8	Zhang	R	9000.05	personal income tax&0.05	China:Shanghai	2014

Table 1-8 Employee contact information

ID	Mobile Number	Email Addresses
1	135 XXXX XXXX	xxxx@example.com
3	159 XXXX XXXX	xxxxx@example.com.cn
4	186 XXXX XXXX	xxxx@example.org
6	189 XXXX XXXX	xxxx@example.cn
8	134 XXXX XXXX	xxxx@example.cn

You can perform the following analysis through a data application:

- Query contact information of employees whose salaries are paid in USD.
- Query the IDs and names of employees who were hired in 2014, and load the query results to a new table.
- Collect the number of employee information records.
- Query information about employees whose email addresses end with "cn".

Solution Architecture

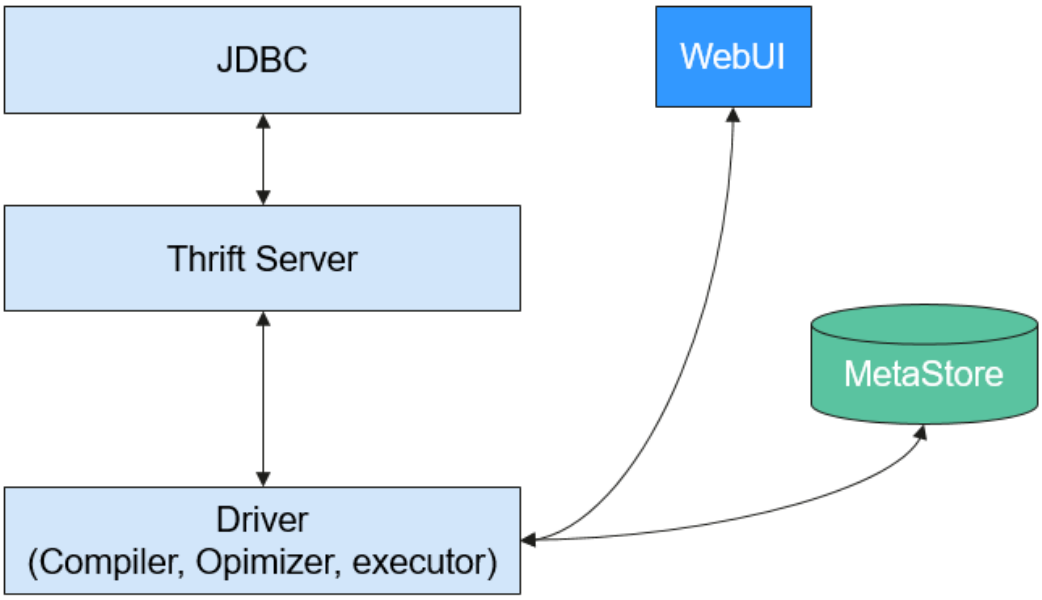
Hive is a data warehouse built on Hadoop. It provides batch computing capability for the big data platform and is able to batch analyze and summarize structured and semi-structured data for data calculation. Hive operates structured data using Hive Query Language (HQL), a SQL-like language. HQL is automatically converted into MapReduce tasks for the query and analysis of massive data in the Hadoop cluster.

Hive is able to:

- Analyze massive structured data and summarizes analysis results.
- Allow complex MapReduce jobs to be compiled in SQL languages.
- Support flexible data storage formats, including JavaScript object notation (JSON), comma separated values (CSV), TextFile, RCFile, SequenceFile, and ORC.

Hive functions as a data warehouse based on HDFS and MapReduce architecture and translates HQL statements into MapReduce jobs or HDFS operations.

Figure 1-16 Hive Architecture



- **Metastore**: reads, writes, and updates metadata such as tables, columns, and partitions. Its lower layer is relational databases.
- **Driver**: manages the lifecycle of HQL execution and participates in the entire Hive job execution.
- **Compiler**: translates HQL statements into a series of interdependent Map or Reduce jobs.
- **Optimizer**: is classified into logical optimizer and physical optimizer to optimize HQL execution plans and MapReduce jobs, respectively.
- **Executor**: runs Map or Reduce jobs based on job dependencies.
- **ThriftServer**: functions as the servers of JDBC, provides Thrift APIs, and integrates with Hive and other applications.
- **Clients**: include the web UI and JDBC APIs and provides APIs for user access.

Procedure

This practice describes how to develop a Hive data analysis application and how to run HQL statements to access Hive data stored in OBS after you connect to Hive through the client. For example, manage and query enterprise employee information. If you need to develop and build your application based on the sample code project provided by MRS, see **Application Development Overview**.

The operation process is as below:

1. **Creating an MRS Offline Query Cluster**
2. **Creating an OBS Agency and Binding It to an MRS Cluster**
3. **Creating a Hive Table and Loading Data from OBS**
4. **Analyzing Data Using HQL**

Creating an MRS Offline Query Cluster

Step 1 Go to the **Buy Cluster** page.

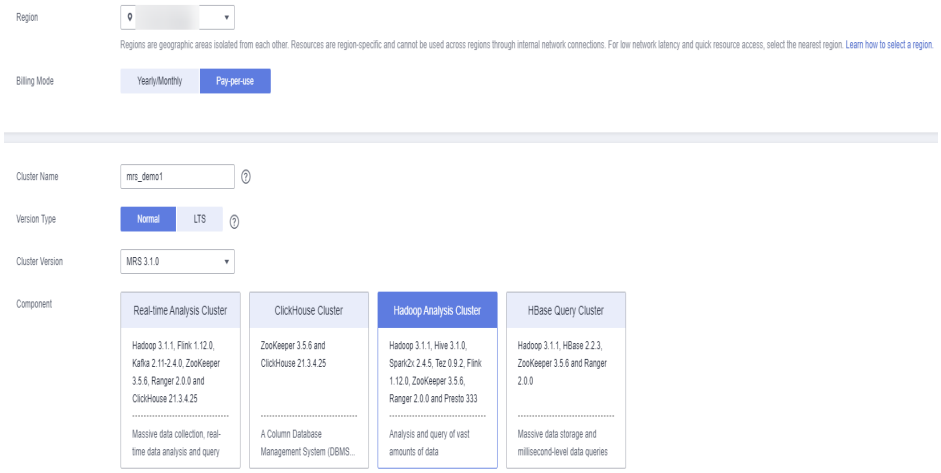
Step 2 Click the **Quick Config** tab and set configuration parameters.

Table 1-9 Software parameters (for reference only)

Parameter	Description	Example Value
Region	Region where the MRS resources belong. MRS clusters in different regions cannot communicate with each other over an intranet. For lower network latency and quick resource access, select the region nearest to you.	CN-Hong Kong
Billing Mode	Billing mode of the cluster.	Pay-per-use
Cluster Name	Name of the MRS cluster.	MRS_demo
Version Type	Version type of the MRS cluster.	Normal
Cluster Version	MRS cluster version.	MRS 3.1.0
Component	Components in the MRS cluster.	Hadoop Analysis Cluster
AZ	Available AZ associated with the cluster region.	AZ1
VPC	VPC where you want to create the cluster. You can click **View VPC** to view the name and ID. If no VPC is available, create one.	vpc-01
Subnet	Subnet where your cluster belongs. You can access the VPC management console to view the names and IDs of existing subnets in the VPC. If no subnet is created under the VPC, click **Create Subnet** to create one.	subnet-01
Enterprise Project	Enterprise project to which the cluster belongs.	default

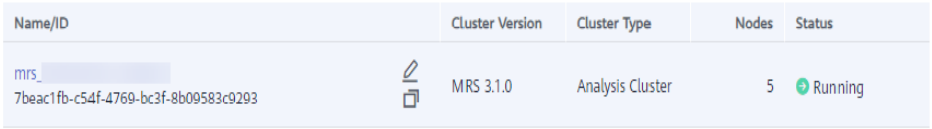
Parameter	Description	Example Value
Kerberos Authentication	Whether to enable Kerberos authentication when logging in to Manager.	Disabled
Username	Name of the administrator of MRS Manager. **admin** is used by default.	admin/root
Password	Password of the MRS Manager administrator.	Set the password for logging in to the cluster management page and ECS node, for example, **Test!@12345**.
Confirm Password	Enter the password of the Manager administrator again.	Enter the password again.
Secure Communications	If the secure communications function is not enabled, MRS clusters cannot be created.	Select **Enable**.

Figure 1-17 Buying a Hadoop analysis cluster



Step 3 Click **Buy Now** and wait until the MRS cluster is created.

Figure 1-18 Cluster created



----End

Creating an OBS Agency and Binding It to an MRS Cluster

NOTE

- MRS presets **MRS_ECS_DEFAULT_AGENCY** in the agency list of IAM so that you can select this agency when creating a custom cluster. This agency has the **OBSOperateAccess** permissions and the **CESFullAccess** (only available for users who have enabled fine-grained policies), **CES Administrator**, and **KMS Administrator** permissions in the region where the cluster resides.
- If you want to use a custom agency, perform the following steps to create an agency. (To create or modify an agency, you must have the **Security Administrator** permission.)

Step 1 Log in to the HUAWEI CLOUD management console.

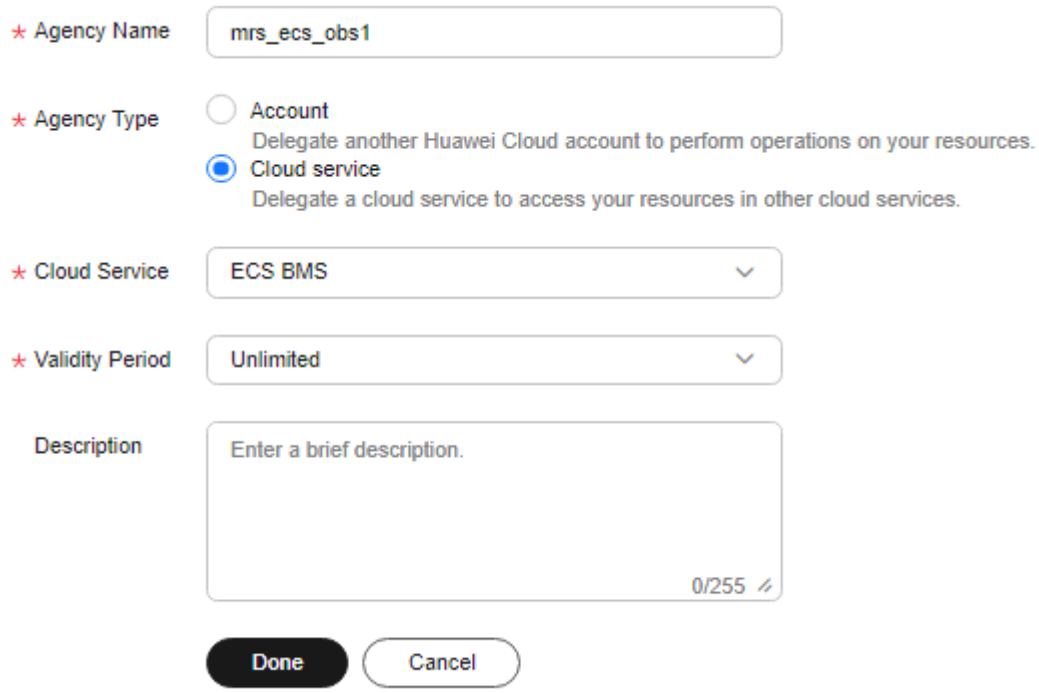
Step 2 Choose **Service List** > **Management & Governance** > **Identity and Access Management**.

Step 3 In the navigation pane on the left, choose **Agencies**. On the displayed page, click **Create Agency**.

Step 4 Set **Agency Name**, select **Cloud service** for **Agency Type**, and select **ECS BMS** for **Cloud Service** to authorize ECS or BMS to invoke OBS.

Step 5 Set **Validity Period** to **Unlimited** and click **Next**.

Figure 1-19 Creating an agency



Step 6 On the displayed page, search for the **OBS OperateAccess** policy and select it.

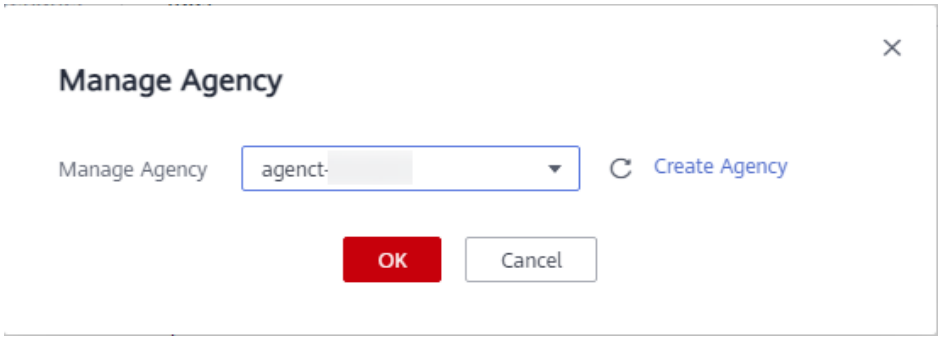
Step 7 Click **Next**. On the displayed page, select the desired scope for permissions you selected. By default, **All resources** is selected. Click **Show More** and select **Global resources**.

Step 8 In the dialog box that is displayed, click **OK** to start authorization. After the message "**Authorization successful.**" is displayed, click **Finish**. The agency is successfully created.

Step 9 Switch back to the MRS console and click the name of the created MRS cluster. On the **Dashboard** page, click **Manage Agency**, select the created OBS agency, and click **OK**.

Figure 1-20 Dashboard tab page of the MRS cluster

Figure 1-21 Binding an agency to an MRS cluster



----End

Creating a Hive Table and Loading Data from OBS

Step 1 Choose **Service List** > **Object Storage Service**. In the navigation pane on the left, choose **Parallel File Systems** and click **Create Parallel File System**, set the following parameters, and click **Create Now**.

Table 1-10 Parallel file system parameters

Parameter	Description	Example Value
Region	Region where the parallel file system is deployed.	CN-Hong Kong
File System Name	Name of the parallel file system.	hiveobs

Parameter	Description	Example Value
Data Redundancy Policy	• **Multi-AZ storage**: Data is stored in multiple AZs to achieve higher reliability. • **Single-AZ storage**: Data is stored in a single AZ, with lower costs.	Single-AZ storage
Policy	Read and write policy of the parallel file system.	Private
Direct Reading	Direct reading allows you to download files from the Archive storage class without restoring them in advance.	Disable
Enterprise Project	Enterprise project where the parallel file system belongs, which facilitates unified management.	default
Tags	(Optional) Tags are used to identify and classify parallel file systems in OBS.	-

Step 2 Download the MRS cluster client, and install it, for example, in the **/opt/client** directory of the active master node. For details, see **Installing a Client**.

You can also use the cluster client provided in the **/opt/Bigdata/client** directory of the master node.

Step 3 Bind an EIP to the active master node and enable port 22 in the security group. Then, log in to the active master node as user **root**, go to the directory where the client is located, and load variables.

cd /opt/client

source bigdata_env

Step 4 Run the **beeline** command to go to the Hive Beeline page.

Run the following command to create an employee information data table **employees_info** that matches the raw data fields:

```
create external table if not exists employees_info
(
id INT,
name STRING,
usd_flag STRING,
salary DOUBLE,
deductions MAP<STRING, DOUBLE>,
```



```
address STRING,
entrytime STRING
)
row format delimited fields terminated by ',' map keys terminated by '&'
stored as textfile
location 'obs://hiveobs/employees_info';
```

Run the following command to create an employee contact information table **employees_contact** that matches the raw data fields:

```
create external table if not exists employees_contact
(
id INT,
phone STRING,
email STRING
)
row format delimited fields terminated by ','
stored as textfile
location 'obs://hiveobs/employees_contact';
```

Step 5 Run the following command to check whether the table is successfully created:

```
show tables;
```

```
+--------------------------+
|         tab_name         |
+--------------------------+
| employees_contact        |
| employees_info           |
+--------------------------+
```

Step 6 Import data to the corresponding OBS table directory.

By default, a folder is created in the specified storage space for a Hive internal table. The table can read data that matches the table structure as long as the file is stored in the folder.

Log in to the OBS Console. On the **Files** page of the created file system, upload the local raw data to the **employees_info** and **employees_contact** folders.

The following is an example of the raw data format:

info.txt:

```
1,Wang,R,8000.01,personal income tax&0.05,China:Shenzhen,2014
3,Tom,D,12000.02,personal income tax&0.09,America:NewYork,2014
4,Jack,D,24000.03,personal income tax&0.09,America:Manhattan,2015
6,Linda,D,36000.04,personal income tax&0.09,America:NewYork,2014
8,Zhang,R,9000.05,personal income tax&0.05,China:Shanghai,2014
```

contact.txt:

```
1,135 XXXX XXXX,xxxx@xx.com
3,159 XXXX XXXX,xxxx@xx.com.cn
```



```
4,189 XXXX XXXX,xxxx@xx.org
6,189 XXXX XXXX,xxxx@xx.cn
8,134 XXXX XXXX,xxxx@xxx.cn
```

Step 7 Run the following command on the Hive Beeline client to check whether the source data is correctly loaded:

select * from employees_info;

```
+-------------------+---------------------+-------------------------+-----------------------
+-------------------------------+-------------------------+--------------------------+
| employees_info.id  | employees_info.name  | employees_info.usd_flag  | employees_info.salary  |
employees_info.deductions  | employees_info.address  | employees_info.entrytime  |
+-------------------+---------------------+-------------------------+-----------------------
+-------------------------------+-------------------------+--------------------------+
| 1                 | Wang                | R                       | 8000.01                | {"personal income tax":0.05}  |
China:Shenzhen          | 2014                    |
| 3                 | Tom                 | D                       | 12000.02               | {"personal income tax":0.09}  |
America:NewYork         | 2014                    |
| 4                 | Jack                | D                       | 24000.03               | {"personal income tax":0.09}  |
America:Manhattan       | 2015                    |
| 6                 | Linda               | D                       | 36000.04               | {"personal income tax":0.09}  |
America:NewYork         | 2014                    |
| 8                 | Zhang               | R                       | 9000.05                | {"personal income tax":0.05}  |
China:Shanghai          | 2014                    |
+-------------------+---------------------+-------------------------+-----------------------
+-------------------------------+-------------------------+--------------------------+
```

select * from employees_contact;

```
+-----------------------+--------------------------+--------------------------+
| employees_contact.id  | employees_contact.phone  | employees_contact.email  |
+-----------------------+--------------------------+--------------------------+
| 1                     | 135 XXXX XXXX            | xxxx@xx.com              |
| 3                     | 159 XXXX XXXX            | xxxx@xx.com.cn           |
| 4                     | 186 XXXX XXXX            | xxxx@xx.org              |
| 6                     | 189 XXXX XXXX            | xxxx@xx.cn               |
| 8                     | 134 XXXX XXXX            | xxxx@xxx.cn              |
+-----------------------+--------------------------+--------------------------+
```

----End

Analyzing Data Using HQL

On the Hive Beeline client, run the HQL statements to analyze the raw data.

Step 1 Query contact information of employees whose salaries are paid in USD.

Run the following command to create a data table for data cleansing:

create table employees_info_v2 as select id, name, regexp_replace(usd_flag, '\s+','') as usd_flag, salary, deductions, address, entrytime from employees_info;

After the Map task is complete, run the following command:

select a.* from employees_info_v2 a inner join employees_contact b on a.id = b.id where a.usd_flag='D';

```
INFO  : MapReduce Jobs Launched:
INFO  : Stage-Stage-3: Map: 1   Cumulative CPU: 2.95 sec   HDFS Read: 8483 HDFS Write: 317 SUCCESS
INFO  : Total MapReduce CPU Time Spent: 2 seconds 950 msec
INFO  : Completed executing command(queryId=omm_20211022162303_c26d4f1b-
a577-4d6c-919c-6cb96095b24b); Time taken: 26.259 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+-------+---------+-------------+-----------+-------------------------------+--------------------+--------------+
```



```
| a.id  | a.name  | a.usd_flag  | a.salary  |         a.deductions          |     a.address      | a.entrytime  |
+-------+---------+-------------+-----------+-------------------------------+--------------------+--------------+
| 3     | Tom     | D           | 12000.02  | {"personal income tax":0.09}  | America:NewYork    | 2014         |
| 4     | Jack    | D           | 24000.03  | {"personal income tax":0.09}  | America:Manhattan  | 2015         |
| 6     | Linda   | D           | 36000.04  | {"personal income tax":0.09}  | America:NewYork    | 2014         |
+-------+---------+-------------+-----------+-------------------------------+--------------------+--------------+
3 rows selected (26.439 seconds)
```

Step 2 Query the IDs and names of employees who were hired in 2014, and load the query results to the partition with the hire date of 2014 in the **employees_info_extended** table.

Run the following to create a table:

create table if not exists employees_info_extended (id int, name string, usd_flag string, salary double, deductions map<string, double>, address string) partitioned by (entrytime string) stored as textfile;

Run the following command to write data into the table:

insert into employees_info_extended partition(entrytime='2014') select id,name,usd_flag,salary,deductions,address from employees_info_v2 where entrytime = '2014';

After data is extracted, run the following command to query the data:

select * from employees_info_extended;

```
+-----------------------------+-------------------------------+---------------------------------
+---------------------------------+-------------------------------------+---------------------------------
+-----------------------------------+
| employees_info_extended.id  | employees_info_extended.name  | employees_info_extended.usd_flag  |
employees_info_extended.salary  | employees_info_extended.deductions  |
employees_info_extended.address  | employees_info_extended.entrytime  |
+-----------------------------+-------------------------------+---------------------------------
+---------------------------------+-------------------------------------+---------------------------------
+-----------------------------------+
| 1                           | Wang                          | R                           | 8000.01                     |
{"personal income tax":0.05}       | China:Shenzhen                | 2014                          |
| 3                           | Tom                           | D                           | 12000.02                    |
{"personal income tax":0.09}       | America:NewYork               | 2014                          |
| 6                           | Linda                         | D                           | 36000.04                    |
{"personal income tax":0.09}       | America:NewYork               | 2014                          |
| 8                           | Zhang                         | R                           | 9000.05                     |
{"personal income tax":0.05}       | China:Shanghai                | 2014                          |
+-----------------------------+-------------------------------+---------------------------------
+---------------------------------+-------------------------------------+---------------------------------
+-----------------------------------+
```

Step 3 Run the following command to collect the number of employee information records:

select count(1) from employees_info_v2;

```
+------+
| _c0  |
+------+
| 5    |
+------+
```

Step 4 Run the following command to query information about employees whose email addresses end with "cn":

select a.*, b.email from employees_info_v2 a inner join employees_contact b on a.id = b.id where b.email rlike '.*cn$';


```
+-------+---------+-------------+-----------+-------------------------------+------------------+--------------
+-----------------+
| a.id  | a.name  | a.usd_flag  | a.salary  |        a.deductions           |    a.address     | a.entrytime  |
b.email     |
+-------+---------+-------------+-----------+-------------------------------+------------------+--------------
+-----------------+
| 3     | Tom     | D           | 12000.02  | {"personal income tax":0.09}  | America:NewYork  | 2014         |
xxxx@xx.com.cn  |
| 6     | Linda   | D           | 36000.04  | {"personal income tax":0.09}  | America:NewYork  | 2014         |
xxxx@xx.cn      |
| 8     | Zhang   | R           | 9000.05   | {"personal income tax":0.05}  | China:Shanghai   | 2014         |
xxxx@xxx.cn     |
+-------+---------+-------------+-----------+-------------------------------+------------------+--------------
+-----------------+
```

----End

1.4 Using Flink Jobs to Process OBS Data

Application Scenarios

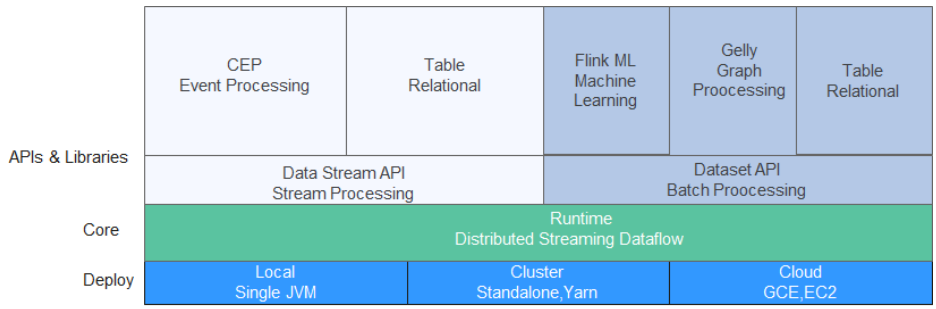
MRS supports decoupled storage and compute in scenarios where a large storage capacity is required and compute resources need to be scaled on demand. This allows you to store your data in OBS and use an MRS cluster only for data computing.

This practice instructs you on how to run Flink jobs in an MRS cluster to process data stored in OBS.

Solution Architecture

Flink is a unified computing framework that supports both batch processing and stream processing. It provides a stream data processing engine that supports data distribution and parallel computing. Flink features stream processing and is a top open-source stream processing engine in the industry.

Flink provides high-concurrency pipeline data processing, millisecond-level latency, and high reliability, making it suitable for low-latency data processing.

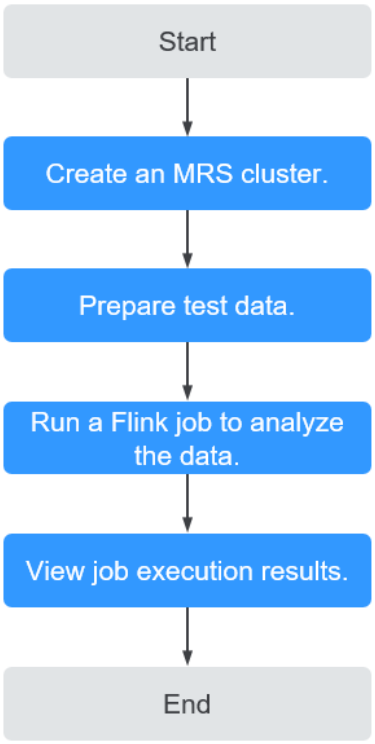


In this example, the Flink WordCount job program built in the MRS cluster is used to analyze the source data stored in the OBS file system and compute the frequency of words in the source data. For details about the program code, see **https://github.com/apache/flink/tree/master/flink-examples/flink-examples-batch/src/main/java/org/apache/flink/examples/java/wordcount**.

You can also obtain **MRS sample code project** and develop other Flink stream job programs by referring to **Flink Development Guide**.

Procedure

The operation process is as follows:



Step 1: Creating an MRS Cluster

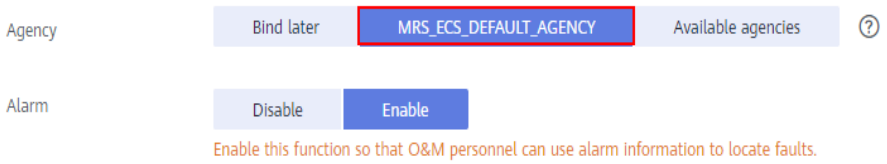
Create and purchase an MRS cluster that contains the Flink component. For details, see **Buying a Custom Cluster**.

NOTE

In this practice, an MRS 3.1.0 cluster with Kerberos authentication disabled is used as an example.

In this example, before you analyze data stored in OBS, bind an IAM agency to the MRS cluster so that cluster components can connect to the OBS file system and have operation permissions on file system directories.

You can select the default **MRS_ECS_DEFAULT_AGENCY** agency or create a custom agency that has the permission to operate the OBS file system.



After the cluster is purchased, install the cluster client on any node of the cluster as user **omm**. For details, see **Installing and Using the Cluster Client**.

Assume that the client is installed in **/opt/client**.

Step 2: Preparing Test Data

Before you create a Flink job for data analysis, prepare test data to be analyzed and upload the data to OBS.

Step 1 Create a file named **mrs_flink_test.txt** on your local PC. For example, the file content is as follows:

```
This is a test demo for MRS Flink. Flink is a unified computing framework that supports both batch processing and stream processing. It provides a stream data processing engine that supports data distribution and parallel computing.
```

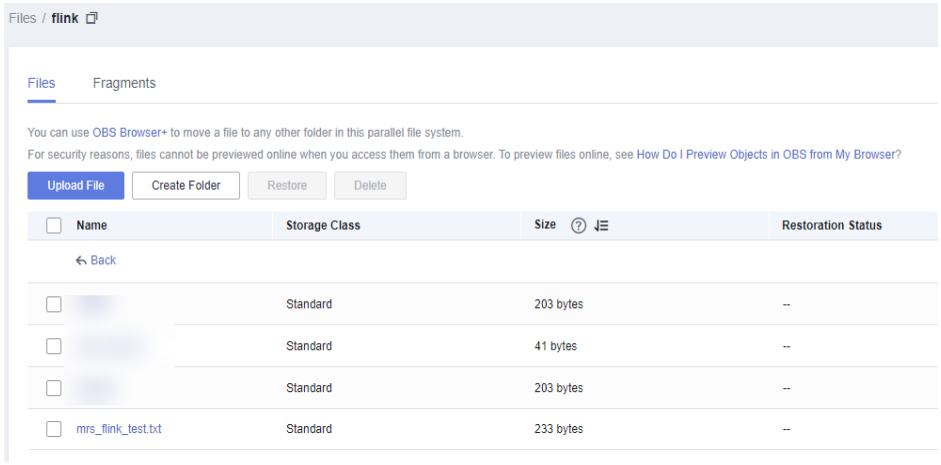
Step 2 Choose **Service List** > **Storage** > **Object Storage Service**.

Step 3 On the OBS management console that is displayed, choose **Parallel File Systems** in the navigation pane on the left. On the page displayed, click **Create Parallel File System** and set required parameters to create a parallel file system. After the system is created, upload the test data to it.

For example, if the created file system is named **mrs-demo-data**, click the system name, and click the **Files** tab. On this tab page, click **Create Folder** to create a folder named **flink** and upload the test data to the folder.

In this example, the complete path of the test data is **obs://mrs-demo-data/flink/mrs_flink_test.txt**.

Figure 1-22 Uploading test data



Step 4 (Optional) Uploading Data Analysis Applications

You can upload the JAR files of the Flink applications developed by yourself to OBS or HDFS of the MRS cluster.

In this example, the Flink WordCount sample program built in the MRS cluster is used. You can obtain the sample program from the MRS cluster client installation directory, that is, **/opt/client/Flink/flink/examples/batch/WordCount.jar**.

Upload **WordCount.jar** to the **mrs-demo-data/program** directory.

----End

Step 3: Creating and Running a Flink Job

Method 1: Submit a job online on the console.

Step 1 Log in to the MRS management console and click the cluster name to go to the cluster details page.

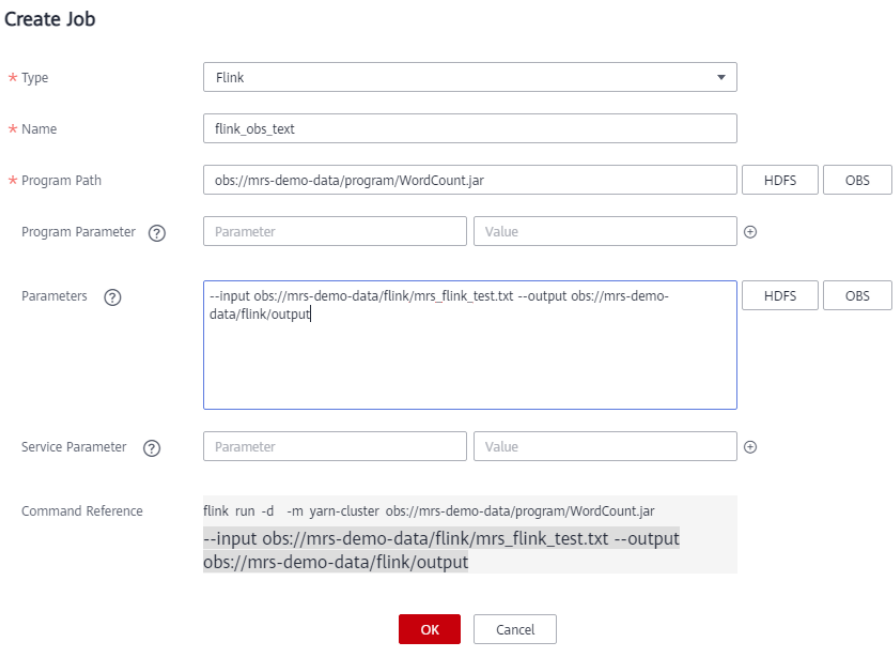
Step 2 On the **Dashboard** tab page, click **Synchronize** next to **IAM User Sync** to synchronize IAM users.

Step 3 Click the **Jobs** tab.

Step 4 Click **Create**. In the **Create Job** dialog box that is displayed, set the following parameters to create a Flink job.

- **Type**: Select **Flink**.
- **Name**: Customize a job name, for example, **flink_obs_test**.
- **Program Path**: In this example, the WordCount program of the Flink client is used.
- **Program Parameter**: Use the default value.
- **Parameters**: Enter the input and output parameters of the application. The **input** parameter indicates the test data to be analyzed, and the **output** parameter indicates the result output file.

  In this example, set this parameter **to --input obs://mrs-demo-data/flink/mrs_flink_test.txt --output obs://mrs-demo-data/flink/output**.
- **Service Parameter**: Use the default values. For details about how to manually configure job parameters, see **Running a Flink Job**.



Step 5 Confirm the job configuration information and click **OK**.

----End

Method 2: Submit a job using the cluster client.

Step 1 Log in to the node where the cluster client is installed as user **root** and go to the client installation directory.

su - omm

cd /opt/client

source bigdata_env

Step 2 Run the following command to check whether the cluster can access OBS:

hdfs dfs -ls obs://mrs-demo-data/flink

Step 3 Submit a Flink job and specify the source file data for consumption.

flink run -m yarn-cluster /opt/client/Flink/flink/examples/batch/WordCount.jar --input obs://mrs-demo-data/flink/mrs_flink_test.txt --output obs://mrs-demo/data/flink/output2

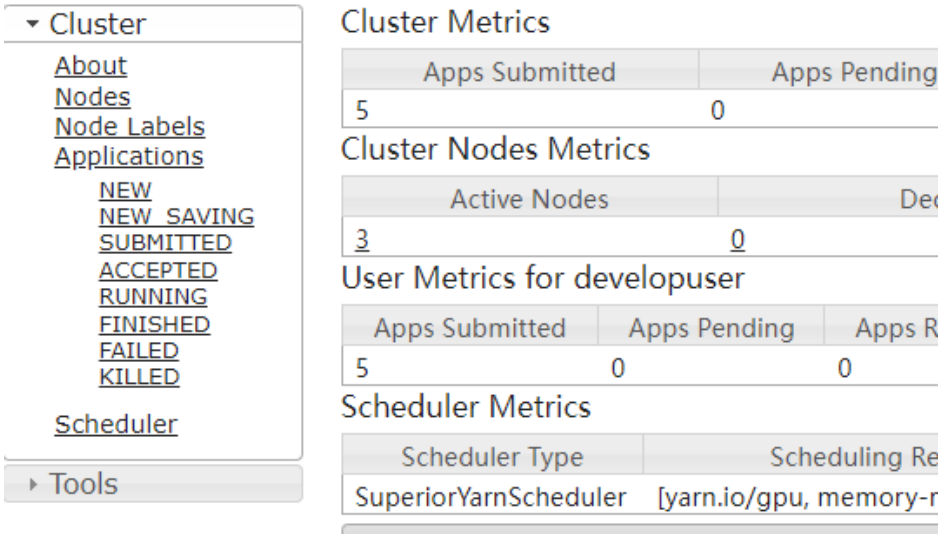
```
...
Cluster started: Yarn cluster with application id application_1654672374562_0011
Job has been submitted with JobID a89b561de5d0298cb2ba01fbc30338bc
Program execution finished
Job with JobID a89b561de5d0298cb2ba01fbc30338bc has finished.
Job Runtime: 1200 ms
```

----End

Step 4: Viewing Job Execution Results

Step 1 After the job is submitted, log in to FusionInsight Manager of the MRS cluster and choose **Cluster** > **Services** > **Yarn**.

Step 2 Click the link next to **ResourceManager WebUI** to access the native Yarn web UI. On the **All Applications** page that is displayed, choose **Applications** on the left, and view the job running status and run logs.



Step 3 After the job execution is complete, you can view the data analysis result in the specified result output file in the OBS file system.

Download the output file to your local PC and open the file to view the analysis result.

```
a 3
and 2
batch 1
both 1
computing 2
data 2
demo 1
distribution 1
engine 1
flink 2
for 1
framework 1
is 2
it 1
mrs 1
parallel 1
processing 3
provides 1
stream 2
supports 2
test 1
that 2
this 1
unified 1
```

If you do not specify the output directory when submitting a job using the cluster client CLI, you can view the data analysis result on the job running page.

```
Job with JobID xxx has finished.
Job Runtime: xxx ms
Accumulator Results:
- e6209f96ffa423974f8c7043821814e9 (java.util.ArrayList) [31 elements]

(a,3)
(and,2)
(batch,1)
(both,1)
(computing,2)
(data,2)
(demo,1)
(distribution,1)
(engine,1)
(flink,2)
(for,1)
(framework,1)
(is,2)
(it,1)
(mrs,1)
(parallel,1)
(processing,3)
(provides,1)
(stream,2)
(supports,2)
(test,1)
(that,2)
```



```
(this,1)
(unified,1)
```

----End

1.5 Consuming Kafka Data Using Spark Streaming Jobs

Application Scenarios

Use an MRS cluster to run Spark Streaming jobs to consume Kafka data.

Assume that Kafka receives one word record every second in a service. The Spark applications developed based on service requirements implements the function of accumulating the total number of records of each word in real time.

Spark Streaming sample projects store data in and send data to Kafka.

Solution Architecture

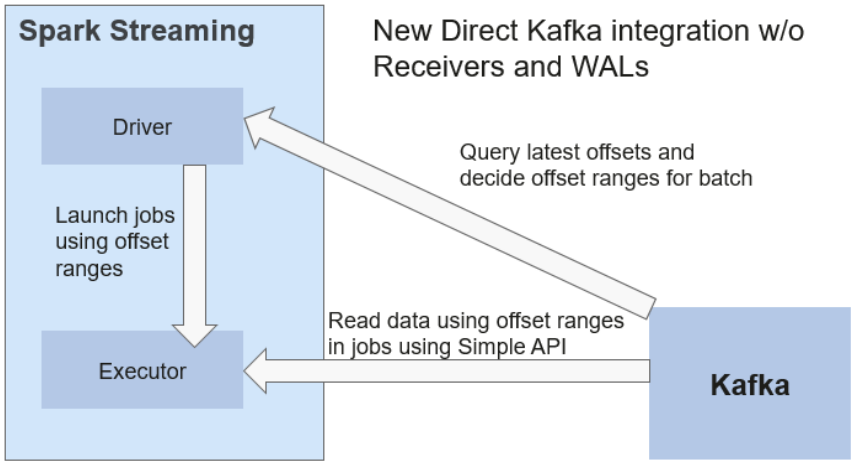
Spark is a distributed batch processing framework. It provides analysis and mining and iterative memory computing capabilities and supports application development in multiple programming languages, including Scala, Java, and Python. Spark applies to the following scenarios:

- Data processing: Spark can process data quickly and has fault tolerance and scalability.
- Iterative computation: Spark supports iterative computation to meet the requirements of multi-step data processing logic.
- Data mining: Based on massive data, Spark can handle complex data mining and analysis and support multiple data mining and machine learning algorithms.
- Streaming processing: Spark supports streaming processing with a seconds-level delay and supports multiple external data sources.
- Query analysis: Spark supports standard SQL query analysis, provides the DSL (DataFrame), and supports multiple external inputs.

Spark Streaming is a real-time computing framework built on the Spark, which expands the capability for processing massive streaming data. Spark supports two data processing approaches: Direct Streaming and Receiver.

In Direct Streaming approach, Direct API is used to process data. Take Kafka Direct API as an example. Direct API provides offset location that each batch range will read from, which is much simpler than starting a receiver to continuously receive data from Kafka and written data to WALs. Then, each batch job is running and the corresponding offset data is ready in Kafka. These offset information can be securely stored in the checkpoint file and read by applications that failed to start.

Figure 1-23 Data transmission through Direct Kafka API

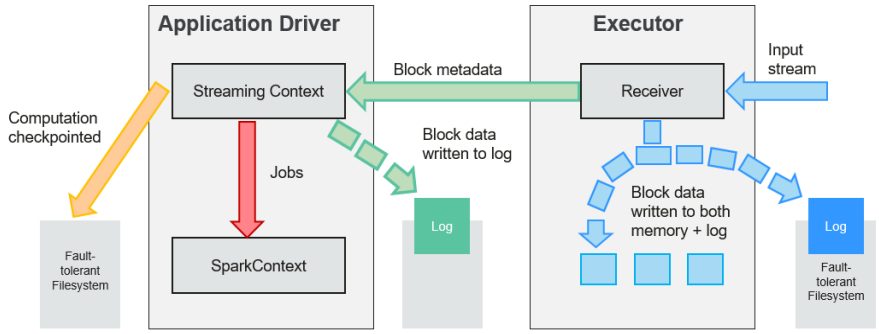


After the failure, Spark Streaming can read data from Kafka again and process the data segment. The processing result is the same no matter Spark Streaming fails or not, because the semantic is processed only once.

Direct API does not need to use the WAL and Receivers, and ensures that each Kafka record is received only once, which is more efficient. In this way, the Spark Streaming and Kafka can be well integrated, making streaming channels be featured with high fault-tolerance, high efficiency, and ease-of-use. Therefore, you are advised to use Direct Streaming to process data.

When a Spark Streaming application starts (that is, when the driver starts), the related StreamingContext (the basis of all streaming functions) uses SparkContext to start the receiver to become a long-term running task. Receiver receives and stores streaming data to the Spark memory for processing. **Figure 1-24** shows the data transfer lifecycle.

Figure 1-24 Data transfer lifecycle



1. Receive data (blue arrow).

   Receiver divides a data stream into a series of blocks and stores them in the executor memory. In addition, after WAL is enabled, it writes data to the WAL of the fault-tolerant file system.

2. Notify the driver (green arrow).

   The metadata in the received block is sent to StreamingContext in the driver. The metadata includes:

  - Block reference ID used to locate the data position in the Executor memory.
  - Block data offset information in logs (if the WAL function is enabled).

3. Process data (red arrow).

   For each batch of data, StreamingContext uses block information to generate resilient distributed datasets (RDDs) and jobs. StreamingContext executes jobs by running tasks to process blocks in the executor memory.

4. Periodically set checkpoints (orange arrows).
5. For fault tolerance, StreamingContext periodically sets checkpoints and saves them to external file systems.

Procedure

Huawei Cloud MRS provides sample development projects for Spark in multiple scenarios. The development guideline for the scenario in this practice is as follows:

1. Receive data from Kafka and generate the corresponding DStream.
2. Classify word records.
3. Compute the result and print it.

Step 1: Creating an MRS Cluster

Step 1 Create and purchase an MRS cluster that contains the Spark2x and Kafka components. For details, see **Buying a Custom Cluster**.

NOTE

In this practice, an MRS 3.1.0 cluster with Kerberos authentication disabled is used as an example.

Step 2 After the cluster is purchased, install the cluster client on any node of the cluster. For details, see **Installing and Using the Cluster Client**.

Assume that the client is installed in **/opt/client**.

----End

Step 2: Preparing Applications

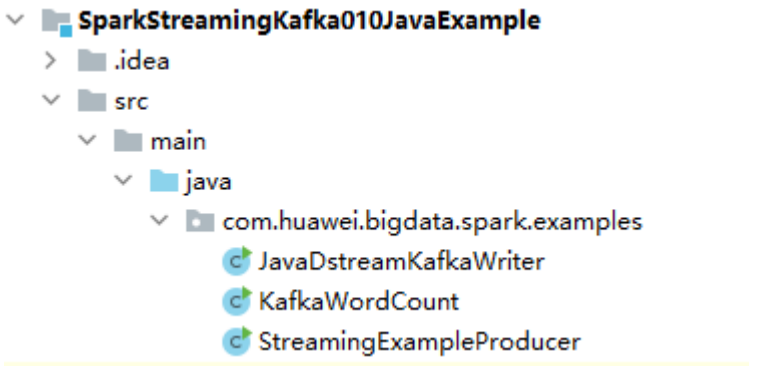
Step 1 Obtain the sample project from Huawei Mirrors.

Download the Maven project source code and configuration files of the sample project, and configure related development tools on the local host. For details, see **Obtaining Sample Projects from Huawei Mirrors**.

Select a sample project based on the cluster version and download the sample project.

For example, to obtain SparkStreamingKafka010JavaExample, visit **https://github.com/huaweicloud/huaweicloud-mrs-example/tree/mrs-3.1.0/src/spark-examples/sparknormal-examples/SparkStreamingKafka010JavaExample**.

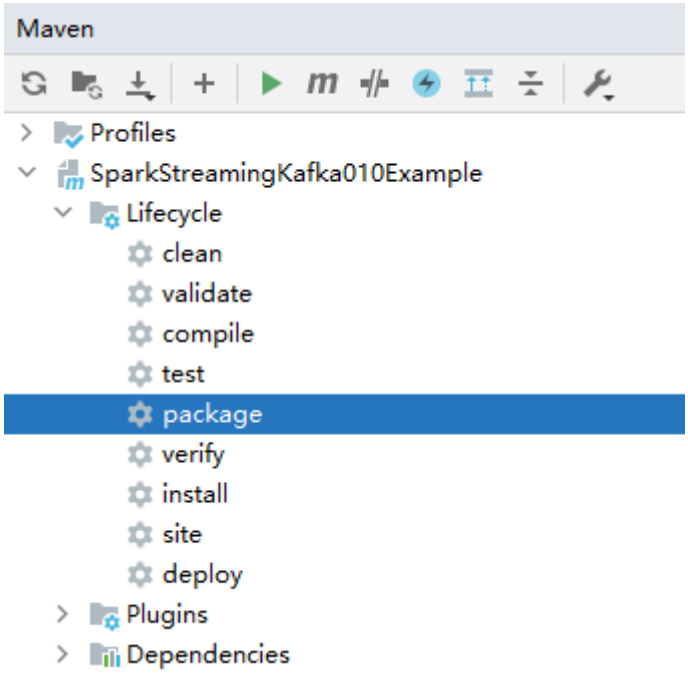
Step 2 Use the IDEA tool to import the sample project and wait for the Maven project to download the dependency package. For details, see **Configuring and Importing Sample Projects**.



In this example project, Streaming is used to call the Kafka API to obtain word records, and word records are classified to obtain the number of records of each word. The key code snippets are as follows:

```
public class StreamingExampleProducer {
    public static void main(String[] args) throws IOException {
        if (args.length < 2) {
            printUsage();
        }
        String brokerList = args[0];
        String topic = args[1];
        String filePath = "/home/data/";    //Path for obtaining the source data
        Properties props = new Properties();
        props.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG, brokerList);
        props.put(ProducerConfig.CLIENT_ID_CONFIG, "DemoProducer");
        props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG, StringSerializer.class.getName());
        props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG, StringSerializer.class.getName());
        Producer<String, String> producer = new KafkaProducer<String, String>(props);

        for (int m = 0; m < Integer.MAX_VALUE / 2; m++) {
            File dir = new File(filePath);
            File[] files = dir.listFiles();
            if (files != null) {
                for (File file : files) {
                    if (file.isDirectory()) {
                        System.out.println(file.getName() + "This is a directory!");
                    } else {
                        BufferedReader reader = null;
                        reader = new BufferedReader(new FileReader(filePath + file.getName()));
                        String tempString = null;
                        while ((tempString = reader.readLine()) != null) {
                            // Blank line judgment
                            if (!tempString.isEmpty()) {
                                producer.send(new ProducerRecord<String, String>(topic, tempString));
                            }
                        }
                        // make sure the streams are closed finally.
                        reader.close();
                    }
                }
            }
            try {
                Thread.sleep(3);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }

    private static void printUsage() {
        System.out.println("Usage: {brokerList} {topic}");
    }
}
```


Step 3 After Maven and SDK parameters are configured on the local host, the sample project automatically loads related dependency packages. After the loading is complete, double-click **package** to obtain the JAR file.



For example, the packaged JAR file is **SparkStreamingKafka010JavaExample-1.0.jar**.

----End

Step 3: Uploading the JAR Package and Source Data

Step 1 Prepare the source data to be sent to Kafka, for example, the following **input_data.txt** file. Upload the file to the **/home/data** directory on the client node.

```
ZhangSan
LiSi
WangwWU
Tom
Jemmmy
LinDa
```

Step 2 Upload the compiled JAR package to a directory, for example, **/opt**, on the client node.

NOTE

If you cannot directly connect to the client node to upload files through the local network, upload the JAR file or source data to OBS, import the file to HDFS on the **Files** tab page of the MRS cluster, and run the **hdfs dfs -get** command on the HDFS client to download the file to the client node.

----End

Step 4: Running the Job and Viewing the Result

Step 1 Log in to the node where the cluster client is installed as user **root**.

cd /opt/client

source bigdata_env

Step 2 Create a Kafka topic for receiving data.

kafka-topics.sh --create --zookeeper *IP address of the quorumpeer instance*:*ZooKeeper client connection port* **/kafka --replication-factor 2 --partitions 3 --topic** *Topic name*

To query the IP address of the quorumpeer instance, log in to FusionInsight Manager of the cluster, choose **Cluster** > **Services** > **ZooKeeper**, and click the **Instance** tab. Use commas (,) to separate multiple IP addresses. You can query the ZooKeeper client connection port by querying the ZooKeeper service configuration parameter **clientPort**. The default value is **2181**.

For example, run the following command:

kafka-topics.sh --create --zookeeper 192.168.0.17:2181/kafka --replication-factor 2 --partitions 2 --topic sparkkafka

```
Created topic sparkkafka.
```

Step 3 After the topic is created, execute the program to send data to Kafka.

java -cp /opt/SparkStreamingKafka010JavaExample-1.0.jar:/opt/client/Spark2x/spark/jars/*:/opt/client/Spark2x/spark/jars/streamingClient010/* com.huawei.bigdata.spark.examples.StreamingExampleProducer *IP address of the Broker instance*:*Kafka connection port Topic name*

To query the IP address of the Kafka Broker instance, log in to FusionInsight Manager of the cluster, choose **Cluster** > **Services** > **Kafka**, and click the **Instance** tab. Use commas (,) to separate multiple IP addresses. You can query the Kafka connection port by querying the Kafka service configuration parameter **port**. The default value is **9092**.

For example, run the following command:

java -cp /opt/SparkStreamingKafka010JavaExample-1.0.jar:/opt/client/Spark2x/spark/jars/*:/opt/client/Spark2x/spark/jars/streamingClient010/* com.huawei.bigdata.spark.examples.StreamingExampleProducer 192.168.0.131:9092 sparkkafka

```
...
    transactional.id = null
    value.serializer = class org.apache.kafka.common.serialization.StringSerializer

2022-06-08 15:43:42 INFO  AppInfoParser:117 - Kafka version: xxx
2022-06-08 15:43:42 INFO  AppInfoParser:118 - Kafka commitId: xxx
2022-06-08 15:43:42 INFO  AppInfoParser:119 - Kafka startTimeMs: xxx
2022-06-08 15:43:42 INFO  Metadata:259 - [Producer clientId=DemoProducer] Cluster ID: d54RYHthSUishVb6nTHP0A
```

Step 4 Open a new client connection window and run the following commands to read data from the Kafka topic:

cd /opt/client/Spark2x/spark

source bigdata_env

bin/spark-submit --master yarn --deploy-mode client --jars $(files=($SPARK_HOME/jars/streamingClient010/*.jar); IFS=,; echo "${files[*]}") --class com.huawei.bigdata.spark.examples.KafkaWordCount /opt/SparkStreamingKafka010JavaExample-1.0.jar *<checkpointDir> <brokers> <topic> <batchTime>*

- **<checkPointDir>** indicates the HDFS path for backing up application results, for example, **/tmp**.
- **<brokers>** indicates the Kafka address for obtaining metadata, in the format of *IP address of the Broker instance*:*Kafka connection port*.
- **<topic>** indicates the topic name read from Kafka.
- **<batchTime>** indicates the interval for Streaming processing in batches, for example, **5**.

For example, run the following commands:

cd /opt/client/Spark2x/spark

source bigdata_env

bin/spark-submit --master yarn --deploy-mode client --jars $(files=($SPARK_HOME/jars/streamingClient010/*.jar); IFS=,; echo "${files[*]}") --class com.huawei.bigdata.spark.examples.KafkaWordCount /opt/SparkStreamingKafka010JavaExample-1.0.jar /tmp 192.168.0.131:9092 sparkkafka 5

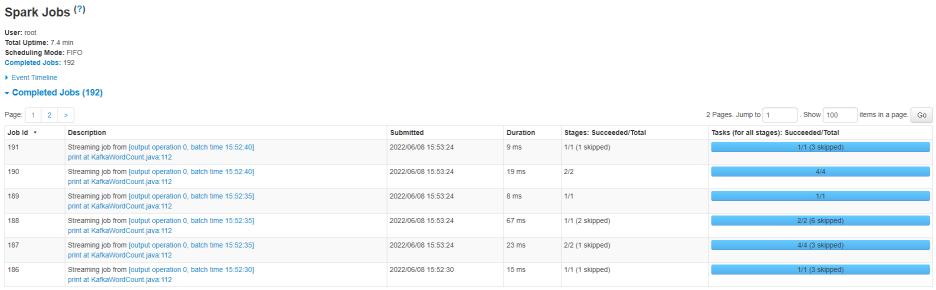
After the program is executed, you can view the data statistics in Kafka.

```
....
-------------------------------------------
Time: 1654674380000 ms
-------------------------------------------
(ZhangSan,6)
(Tom,6)
(LinDa,6)
(WangwWU,6)
(LiSi,6)
(Jemmmy,6)

-------------------------------------------
Time: 1654674385000 ms
-------------------------------------------
(ZhangSan,717)
(Tom,717)
(LinDa,717)
(WangwWU,717)
(LiSi,717)
(Jemmmy,717)

-------------------------------------------
Time: 1654674390000 ms
-------------------------------------------
(ZhangSan,2326)
(Tom,2326)
(LinDa,2326)
(WangwWU,2326)
(LiSi,2326)
(Jemmmy,2326)
 ...
```

Step 5 Log in to FusionInsight Manager and choose **Cluster** > **Services** > **Spark2x**.

Step 6 On the **Dashboard** tab page that is displayed, click the link next to **Spark WebUI** to access the History Server web UI.

Click a job ID to view the status of the Spark Streaming job.



----End

1.6 Using Flume to Collect Log Files from a Specified Directory to HDFS

Application Scenarios

Flume is a distributed, reliable, and highly available system for aggregating massive logs. It can efficiently collect, aggregate, and move massive amounts of log data from different data sources and store the data in a centralized data storage system. Data senders can be customized in the system to collect data. In addition, Flume provides the capability of simply processing data and writing data to data receivers (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. he client-side and service-side FlumeAgents work independently and provide the same functions.

The Flume client needs to be installed separately. It can be used to import data directly to components such as HDFS and Kafka of a cluster.

In this practice, the Flume component of a custom MRS cluster is used to automatically collect new files generated in the log directory of a specified node and store the files to HDFS.

Solution Architecture

A Flume-NG consists of agents. Each agent consists of three components (source, channel, and sink). A source is used for receiving data. A channel is used for transmitting data. A sink is used for sending data to the next end.

Figure 1-25 Flume-NG architecture

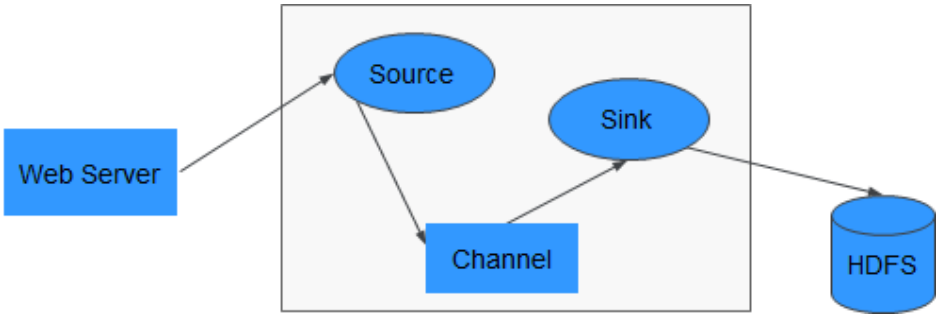


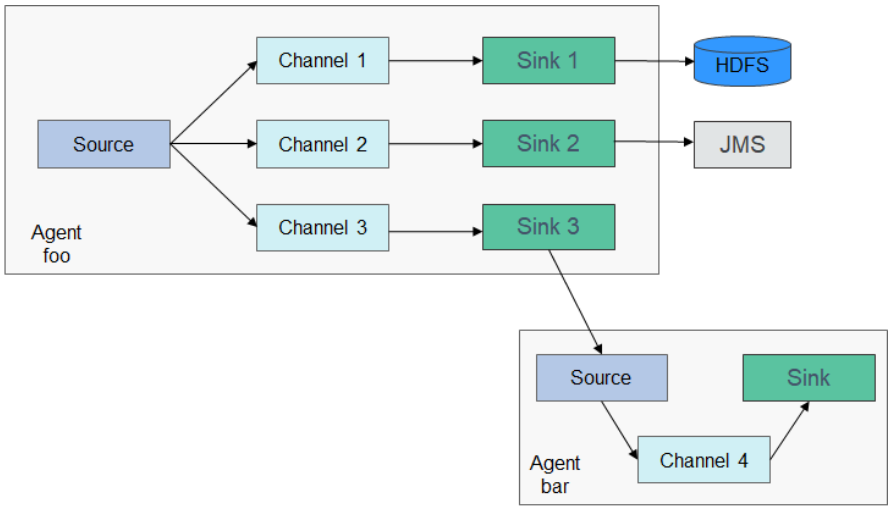
Table 1-11 Module description

Name	Description
Source	A source receives data or generates data by using a special mechanism, and places the data in batches in one or more channels. The source can work in data-driven or polling mode. Typical source types are as follows: • Sources that are integrated with the system, such as Syslog and Netcat • Sources that automatically generate events, such as Exec and SEQ • IPC sources that are used for communication between agents, such as Avro A Source must associate with at least one channel.
Channel	A channel is used to buffer data between a source and a sink. The channel caches data from the source and deletes that data after the sink sends the data to the next-hop channel or final destination. Different channels provide different persistence levels. • Memory channel: non-persistency • File channel: Write-Ahead Logging (WAL)-based persistence • JDBC channel: persistency implemented based on the embedded database The channel supports the transaction feature to ensure simple sequential operations. A channel can work with sources and sinks of any quantity.

Name	Description
Sink	A sink sends data to the next-hop channel or final destination. Once completed, the transmitted data is removed from the channel. Typical sink types are as follows: • Sinks that send storage data to the final destination, such as HDFS and HBase • Sinks that are consumed automatically, such as Null Sink • IPC sinks used for communication between Agents, such as Avro A sink must be associated with a specific channel.

As shown in **Figure 1-26**, a Flume client can have multiple sources, channels, and sinks.

Figure 1-26 Flume structure



Step 1: Creating an MRS Cluster

Step 1 Create and purchase an MRS cluster that contains the Flume and HDFS components. For details, see **Buying a Custom Cluster**.

NOTE

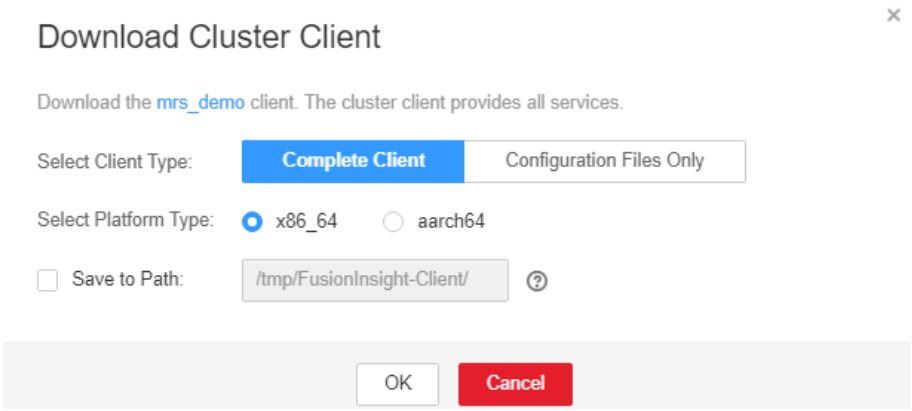
In this practice, an MRS 3.1.0 cluster with Kerberos authentication disabled is used as an example.

Step 2 After the cluster is purchased, log in to FusionInsight Manager of the cluster, download the cluster client, and decompress it.

The Flume client needs to be installed separately. You need to download the cluster client installation package to the node where the Flume client is to be installed and decompress the package.

1. On the **Homepage** page of FusionInsight Manager, click ··· next to the cluster name and click **Download Client** to download the cluster client.
2. On the **Download Cluster Client** page, enter the cluster client download information.

   Figure 1-27 Downloading the cluster client

   - Set **Select Client Type** to **Complete Client**.
   - Set **Select Platform Type** to the architecture of the node to install the client. **x86_64** is used as an example.
   - Select **Save to Path** and enter the download path, for example, **/tmp/FusionInsight-Client/**. Ensure that user **omm** has the operation permission on the path.
3. After the client software package is downloaded, log in to the active OMS node of the cluster as user **root** and copy the installation package to a specified node.

   By default, the client software package is downloaded to the active OMS node of the cluster. You can view the node marked with ★ on the host page of FusionInsight Manager. If you need to install the client software package on another node in the cluster, run the following command to transfer the software package to the target node.

   cd /tmp/FusionInsight-Client/

   scp -p FusionInsight_Cluster_1_Services_Client.tar *IP address of the node where the Flume client is to be installed*:**/tmp**
4. Log in to the node where the Flume client is to be installed as user **root**, go to the directory where the client software package is stored, and run the following commands to decompress the software package:

   tar -xvf FusionInsight_Cluster_1_Services_Client.tar

   tar -xvf FusionInsight_Cluster_1_Services_ClientConfig.tar

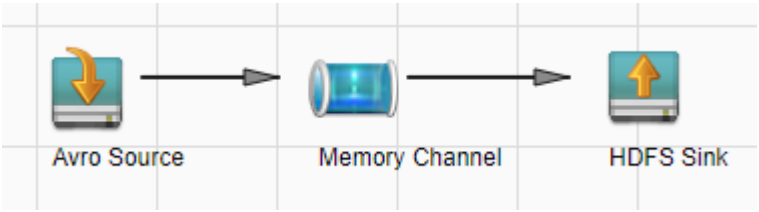
----End

Step 2: Generating the Flume Configuration File

Step 1 Log in to FusionInsight Manager and choose **Cluster** > **Services**. On the page that is displayed, choose **Flume**. On the displayed page, click the **Configuration Tool** tab.

Step 2 Configure and export the **properties.properties** file.

Set **Agent Name** to **server**, select **Avro Source**, **Memory Channel**, and **HDFS Sink**, and connect them.

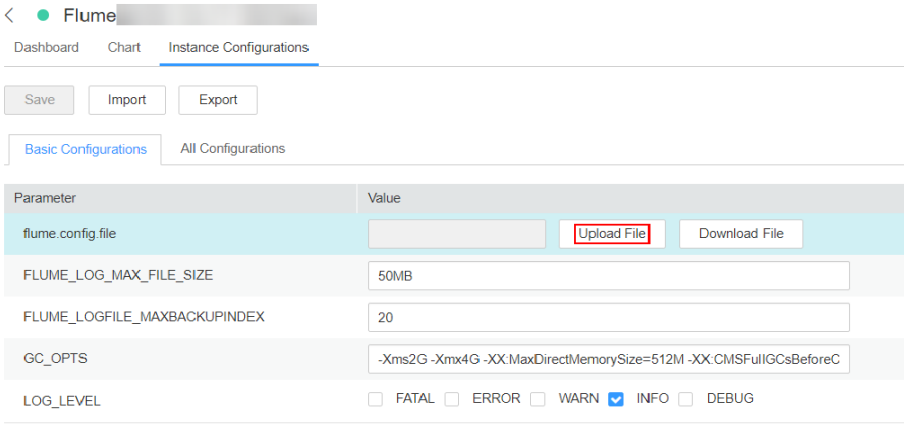


Double-click the module icon and set the parameters according to the following table. Retain the default values for the parameters not listed.

Type	Parameter	Description	Example Value
Avro Source	Name	Module name, which is customizable	test_source_1
	bind	IP address of the node where the Flume role resides. You can choose **Cluster** > **Services** > **Flume** > **Instances** to view the IP address of any Flume role instance.	192.168.10.192
	port	Connection port. The port number starts from 21154.	21154
Memory Channel	Name	Module name, which is customizable	test_channel_1
HDFS Sink	Name	Module name, which is customizable	test_sink_1
	hdfs.path	HDFS directory to which log files are written	hdfs://hacluster/flume/test
	hdfs.filePrefix	Prefix of the file name written to HDFS	over_%{basename}

Step 3 Click **Export** to download the **properties.properties** file to your local PC.

Step 4 On FusionInsight Manager, choose **Cluster** > **Services** > **Flume**, click the **Instance** tab, and click the Flume role in the row of the node where the configuration file is to be uploaded. The **Instance Configurations** tab page is displayed.



Step 5 Click **Upload File** and upload the **properties.properties** file.

Click **Save**. Then click **OK**.

Step 6 Choose **Cluster** > **Services** > **Flume**. On the page that is displayed, click the **Configuration Tool** tab.

Set **Agent Name** to **client**, select **SpoolDir Source**, **Memory Channel**, and **Avro Sink**, and connect them.

Double-click the module icon and set the parameters according to the following table. (Retain the default values for the parameters not listed.)

Type	Parameter	Description	Example Value
SpoolDir Source	Name	Module name, which is customizable	test_source_1
	spoolDir	Directory where logs need to be collected. The Flume running user must have the read and write permissions on the directory, and the permissions must be verified by storing files in the directory.	/var/log/Bigdata/audit/test
Memory Channel	Name	Module name, which is customizable	test_channel_1
HDFS Sink	Name	Module name, which is customizable	test_sink_1
	hostname	IP address of the node where the Flume role to be connected resides	192.168.10.192

Type	Parameter	Description	Example Value
	port	Connection port. The port number starts from 21154.	21154

Step 7 Click **Export** to download the **properties.properties** file to your local PC.

Step 8 Rename the **properties.properties** file as **client.properties.properties**, and upload the file to the *Path where the cluster client installation package is decompressed*/**Flume/FlumeClient/flume/conf** directory on the Flume client node.

----End

Step 3: Installing the Flume Client

Step 1 Log in to the node where the Flume client is to be installed as user **root**.

Step 2 Go to the path where the client installation package is decompressed. For example, the client installation package has been uploaded to **/tmp** and then decompressed.

Step 3 Run the following commands to install the Flume client. In the command, **/opt/FlumeClient** indicates the custom Flume client installation path.

cd /tmp/FusionInsight-Client/FusionInsight_Cluster_1_Services_ClientConfig/Flume/FlumeClient

./install.sh -d /opt/FlumeClient -c flume/conf/client.properties.properties

```
CST ... [flume-client install]: install flume client successfully.
```

----End

Step 4: Viewing Log Collection Results

Step 1 After the Flume client is installed, write new log files to the log collection directory to check whether logs are transmitted.

For example, create several log files in the **/var/log/Bigdata/audit/test** directory.

cd /var/log/Bigdata/audit/test

vi log1.txt

```
Test log file 1!!!
```

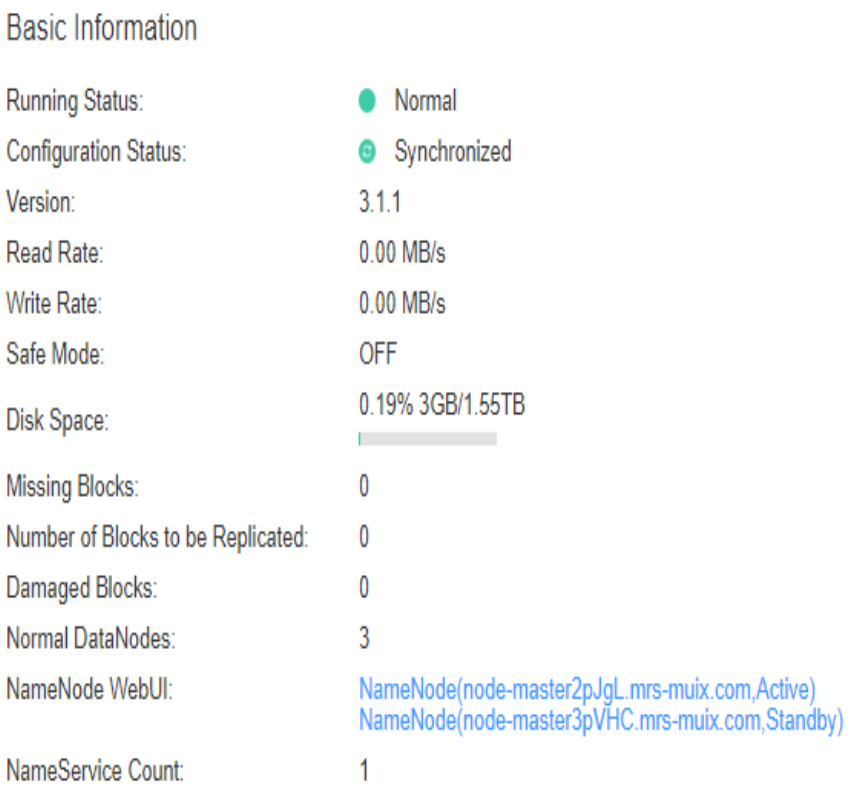
vi log2.txt

```
Test log file 2!!!
```

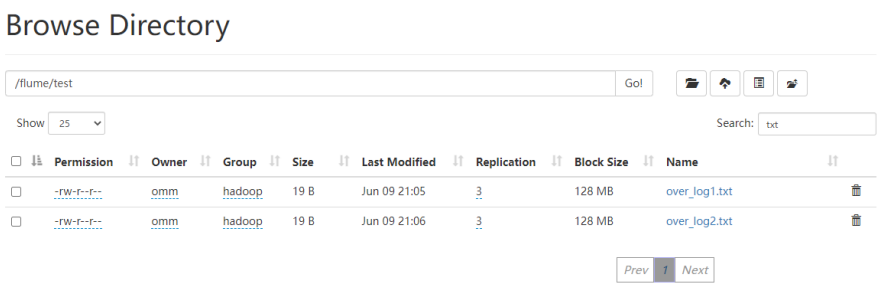
Step 2 After the log files are written, run the **ll** command to view the file list. If the suffix **.COMPLETED** is automatically added to the file names, the log files have been collected.

```
-rw-------. 1 root root      75 Jun  9 19:59 log1.txt.COMPLETED
-rw-------. 1 root root      75 Jun  9 19:59 log2.txt.COMPLETED
```


Step 3 Log in to FusionInsight Manager and choose **Cluster** > **Services** > **HDFS**. On the Dashboard tab page that is displayed, click the **NameNode(***Node name***,Active)** link next to **NameNode WebUI** to access the HDFS web UI.



Step 4 Choose **Utilities** > **Browse the file system** and check whether data is generated in the **/flume/test** directory in HDFS.



As shown above, log files are generated in the directory, and the prefix **over_** is added to the file names.

Download the log file **over_log1.txt** and check whether its content is the same as that of the log file **log1.txt**.

```
Test log file 1!!!
```

----End

1.7 Kafka-based WordCount Data Flow Statistics Case

Application Scenarios

Use an MRS cluster to run Kafka programs to process data.

Kafka Streams is a lightweight stream processing framework provided by Apache Kafka, where the input and output data are stored in Kafka clusters.

The following uses WordCount as an example.

Solution Architecture

Kafka is a distributed message publish-subscribe system. With features similar to JMS, Kafka processes active streaming data.

Kafka applies to many scenarios, such as message queuing, behavior tracing, O&M data monitoring, log collection, stream processing, event tracing, and log persistence.

Kafka has the following features:

- High throughput
- Message persistence to disks
- Scalable distributed system
- High fault tolerance

Procedure

Huawei Cloud MRS provides sample development projects for Kafka in multiple scenarios. The development guideline for the scenario in this practice is as follows:

1. Create two topics on the Kafka client to serve as the input and output topics.
2. Develop a Kafka Streams to implement the word count function. The system collects statistics on the number of words in each message by reading the message in the input topic, consumes data from the output topic, and provides the statistical result in the form of a key-value pair.

Step 1: Creating an MRS Cluster

Step 1 Create and purchase an MRS cluster that contains the Kafka component. For details, see **Buying a Custom Cluster**.

NOTE

In this practice, an MRS 3.1.0 cluster, with Hadoop and Kafka installed and with Kerberos authentication disabled, is used as an example.

Step 2 After the cluster is purchased, install the cluster client on any node of the cluster. For details, see **Installing and Using the Cluster Client**.

For example, install the client in the **/opt/client** directory on the active management node.

Step 3 After the client is installed, create the **lib** directory on the client to store related JAR files.

Copy the Kafka JAR files in the directory decompressed during client installation to **lib**.

For example, if the download path of the client software package is **/tmp/FusionInsight-Client** on the active management node, run the following commands:

mkdir /opt/client/lib

cd /tmp/FusionInsight-Client/FusionInsight_Cluster_1_Services_ClientConfig

scp Kafka/install_files/kafka/libs/* /opt/client/lib

----End

Step 2: Preparing Applications

Step 1 Obtain the sample project from Huawei Mirrors.

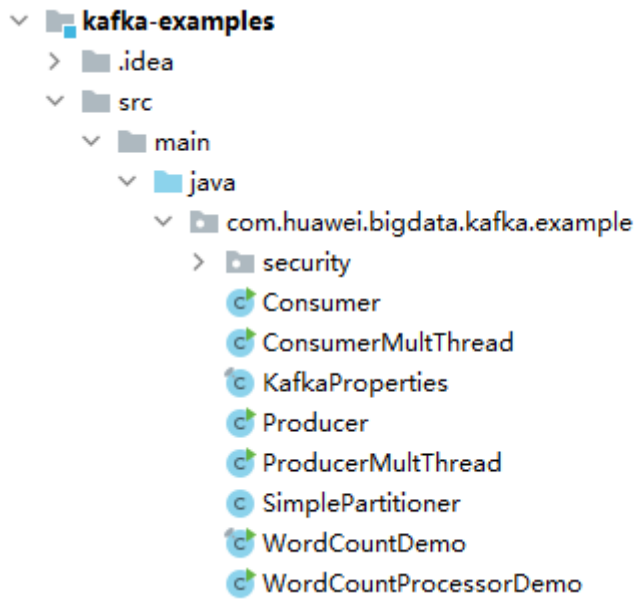
Download the Maven project source code and configuration files of the sample project, and configure related development tools on the local host. For details, see **Obtaining Sample Projects from Huawei Mirrors**.

Select a sample project based on the cluster version and download the sample project.

For example, to obtain WordCountDemo, visit **https://github.com/huaweicloud/huaweicloud-mrs-example/tree/mrs-3.1.0/src/kafka-examples**.

Step 2 Use IntelliJ IDEA to import the sample project locally and wait for the Maven project to download related dependency packages.

After Maven and SDK parameters are configured on the local host, the sample project automatically loads related dependency packages. For details, see **Configuring and Importing a Sample Project**.



In this sample program WordCountDemo, Kafka APIs are called to obtain word records, and word records are classified to obtain the number of records of each word. The key code snippets are as follows:

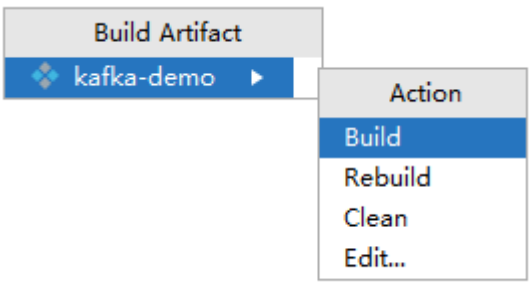
```
...
    static Properties getStreamsConfig() {
        final Properties props = new Properties();
        KafkaProperties kafkaProc = KafkaProperties.getInstance();
        // Broker address list. Configure this parameter based on site requirements.
        props.put(BOOTSTRAP_SERVERS, kafkaProc.getValues(BOOTSTRAP_SERVERS, "node-group-1kLFk.mrs-rbmq.com:9092"));
        props.put(SASL_KERBEROS_SERVICE_NAME, "kafka");
        props.put(KERBEROS_DOMAIN_NAME, kafkaProc.getValues(KERBEROS_DOMAIN_NAME, "hadoop.hadoop.com"));
        props.put(APPLICATION_ID, kafkaProc.getValues(APPLICATION_ID, "streams-wordcount"));
        // Protocol type. The value can be SASL_PLAINTEXT or PLAINTEXT.
        props.put(SECURITY_PROTOCOL, kafkaProc.getValues(SECURITY_PROTOCOL, "PLAINTEXT"));
        props.put(CACHE_MAX_BYTES_BUFFERING, 0);
        props.put(DEFAULT_KEY_SERDE, Serdes.String().getClass().getName());
        props.put(DEFAULT_VALUE_SERDE, Serdes.String().getClass().getName());
        props.put(ConsumerConfig.AUTO_OFFSET_RESET_CONFIG, "earliest");
        return props;
    }
    static void createWordCountStream(final StreamsBuilder builder) {
        // Receives input records from the input topic.
        final KStream<String, String> source = builder.stream(INPUT_TOPIC_NAME);
        // Aggregates calculation results of the key-value pair.
        final KTable<String, Long> counts = source
                .flatMapValues(value -> Arrays.asList(value.toLowerCase(Locale.getDefault()).split(REGEX_STRING)))
                .groupBy((key, value) -> value)
                .count();
        // Outputs the key-value pairs from the output topic.
        counts.toStream().to(OUTPUT_TOPIC_NAME, Produced.with(Serdes.String(), Serdes.Long()));
    }
...
```

NOTE

- Set **BOOTSTRAP_SERVERS** to the host names and port numbers of Kafka broker nodes based on site requirements. For details about the broker information in **Commissioning an Application in Linux**, log in to FusionInsight Manager, choose **Cluster** > **Services** > **Kafka**, and click the **Instance** tab.
- **SECURITY_PROTOCOL** indicates the protocol type for connecting to Kafka. In this example, set this parameter to **PLAINTEXT**.

Step 3 After confirming that the parameters in **WordCountDemo.java** are correct, compile the project and package it to obtain the JAR file.

For details about how to compile a JAR file, see **Commissioning an Application in Linux**.



For example, the packaged JAR file is **kafka-demo.jar**.

----End

Step 3: Uploading the JAR File and Source Data

Upload the compiled JAR file to a directory, for example, **/opt/client/lib**, on the client node.

NOTE

If you cannot directly connect to the client node to upload files through the local network, upload the JAR file or source data to OBS, import the file to HDFS on the **Files** tab page of the MRS cluster, and run the **hdfs dfs -get** command on the HDFS client to download the file to the client node.

Step 4: Running the Job and Viewing the Result

Step 1 Log in to the node where the cluster client is installed as user **root**.

cd /opt/client

source bigdata_env

Step 2 Create an input topic and an output topic. Ensure that the topic names are the same as those specified in the sample code. Set the cleanup policy of the output topic to **compact**.

kafka-topics.sh --create --zookeeper *IP address of the quorumpeer instance*:*ZooKeeper client connection port* **/kafka --replication-factor 1 --partitions 1 --topic** *Topic name*

To query the IP address of the quorumpeer instance, log in to FusionInsight Manager of the cluster, choose **Cluster** > **Services** > **ZooKeeper**, and click the **Instance** tab. Use commas (,) to separate multiple IP addresses. You can query the ZooKeeper client connection port by querying the ZooKeeper service configuration parameter **clientPort**. The default value is **2181**.

For example, run the following commands:

kafka-topics.sh --create --zookeeper 192.168.0.17:2181/kafka --replication-factor 1 --partitions 1 --topic streams-wordcount-input

kafka-topics.sh --create --zookeeper 192.168.0.17:2181/kafka --replication-factor 1 --partitions 1 --topic streams-wordcount-output --config cleanup.policy=compact

Step 3 After the topics are created, run the following command to run the program:

java -cp .:/opt/client/lib/* com.huawei.bigdata.kafka.example.WordCountDemo

Step 4 Open a new client connection window and run the following commands to use **kafka-console-producer.sh** to write messages to the input topic:

cd /opt/client

source bigdata_env

kafka-console-producer.sh --broker-list *Broker instance IP address*:*Kafka connection port (For example, 192.168.0.13:9092)* **--topic streams-wordcount-input --producer.config /opt/client/Kafka/kafka/config/producer.properties**

Step 5 Open a new client connection window and run the following commands to use **kafka-console-consumer.sh** to consume data from the output topic and view the statistics result:

cd /opt/client

source bigdata_env

kafka-console-consumer.sh --topic streams-wordcount-output --bootstrap-server *Broker instance IP address*:*Kafka connection port* **--consumer.config /opt/client/Kafka/kafka/config/consumer.properties --from-beginning --property print.key=true --property print.value=true --property key.deserializer=org.apache.kafka.common.serialization.StringDeserializer --property value.deserializer=org.apache.kafka.common.serialization.LongDeserializer --formatter kafka.tools.DefaultMessageFormatter**

Write a message to the input topic.

```
>This is Kafka Streams test
>test starting
>now Kafka Streams is running
>test end
```

The message is output as follows:

```
this    1
is      1
kafka   1
streams 1
test    1
test    2
starting 1
now     1
kafka   2
streams 2
is      2
running 1
test    3
end     1
```

----End

2 Data Migration

2.1 Data Migration Solution

2.1.1 Making Preparations

This section describes how to migrate HDFS, HBase, and Hive data to an MRS cluster in different scenarios. During data migration, data may be overwritten, lost, or damaged. This document is for reference only. Please cooperate with Huawei Cloud technical personnel to formulate and implement a specific data migration solution.

Make preparations on a source cluster before data migration to prevent the source cluster from generating new data during data migration, thereby avoiding data inconsistency between the source and destination clusters after data migration. Before data migration is complete, the destination cluster must be in the initial state and cannot run any other services except data migration jobs.

Stopping Cluster Services and the Related Services

- If the Kafka service is involved in your cluster, stop all jobs that generate data in Kafka. Wait until the Kafka consumption tasks have consumed the inventory data in Kafka, and then perform the next step.
- Stop all services and jobs related to HDFS, HBase, and Hive, and stop the HBase and Hive services.

Establishing a Data Transmission Channel

- If the source cluster and destination cluster are deployed in different VPCs in the same region, create a network connection between the two VPCs to establish a data transmission channel at the network layer. For details, see **VPC Peering Connection Overview**.
- If the source cluster and destination cluster are deployed in the same VPC but belong to different security groups, add security group rules to each security group on the VPC management console. In the security rules, **Protocol** is set to **ANY**, **Transfer Direction** is set to **Inbound**, and **Source** is set to **Security Group** (the security group of the peer cluster).

  - To add an inbound rule to the security group of the source cluster, select the security group of the destination cluster in **Source**.
  - To add an inbound rule to the security group of the destination cluster, select the security group of the source cluster in **Source**.
- If the source and destination clusters are deployed in the same security group of the same VPC and Kerberos authentication is enabled for both clusters, you need to configure mutual trust between the two clusters. For details, see **Configuring Cross-Manager Mutual Trust Between Clusters**.

2.1.2 Exporting Metadata

To ensure that the data properties and permissions of the source cluster are consistent with those of the destination cluster, metadata of the source cluster needs to be exported to restore metadata after data migration.

The metadata to be exported includes the owner, group, and permission information of the HDFS files and Hive table description.

Exporting HDFS Metadata

The metadata information to be exported includes the permission, owner, and group information of files and folders. You can run the following HDFS client command to export:

```
$HADOOP_HOME/bin/hdfs dfs -ls -R <migrating_path> > /tmp/hdfs_meta.txt
```

- *$HADOOP_HOME*: installation directory of the Hadoop client in the source cluster
- *<migrating_path>*: HDFS data directory to be migrated
- */tmp/hdfs_meta.txt*: local path for storing the exported metadata

NOTE

If the source cluster can communicate with the destination cluster and you run the **hadoop distcp** command as an administrator to copy data, you can add the **-p** parameter to enable DistCp to restore the metadata of the corresponding file in the destination cluster while copying data. In this case, you can skip this step.

Exporting Hive Metadata

Hive table data is stored in HDFS. Table data and the metadata of the table data is centrally migrated in directories by HDFS in a unified manner. Metadata of Hive tables can be stored in different types of relational databases (such as MySQL, PostgreSQL, and Oracle) based on cluster configurations.

The exported metadata of the Hive tables in this document is the Hive table description stored in the relational database.

The mainstream big data release editions in the industry support Sqoop installation. For on-premises big data clusters of the community version, you can download the Sqoop of the community version for installation. Use Sqoop to decouple the metadata to be exported and the relational database, then export Hive metadata to HDFS, and migrate it together with the table data for restoration.

The following uses Account A (you) and Account B (another user) as an example:

Step 1 Download the Sqoop tool from the source cluster and install it.

For details, see **http://sqoop.apache.org/**.

Step 2 Download the JDBC driver of the relational database to the **${Sqoop_Home}/lib** directory.

Step 3 Run the following command to export all Hive metadata tables:

All exported data is stored in the **/user/<*user_name*>/<*table_name*>** directory on HDFS.

```
$Sqoop_Home/bin/sqoop import --connect jdbc:<driver_type>://<ip>:<port>/<database> --table <table_name> --username <user> -password <passwd> -m 1
```

- *$Sqoop_Home*: Sqoop installation directory
- <*driver_type*>: Database type
- *<ip>*: IP address of the database in the source cluster
- *<port>*: Port number of the database in the source cluster
- *<table_name>*: Name of the table to be exported
- <*user*>: Username
- <*passwd*>: User password

NOTE

Commands carrying authentication passwords pose security risks. Disable historical command recording before running such commands to prevent information leakage.

----End

2.1.3 Copying Data

Based on the regions of and network connectivity between the source cluster and destination cluster, data copy scenarios are classified as follows:

Same Region

If the source cluster and destination cluster are in the same region, follow instructions in **Establishing a Data Transmission Channel** to configure the network and set up a network transmission channel. Use the DistCp tool to run the following command to copy the HDFS, HBase, Hive data files and Hive metadata backup files from the source cluster to the destination cluster.

```
$HADOOP_HOME/bin/hadoop distcp <src> <dist> -p
```

- **$HADOOP_HOME**: installation directory of the Hadoop client in the destination cluster
- **<src>**: HDFS directory of the source cluster
- **<dist>**: HDFS directory of the destination cluster

Different Regions

If the source cluster and target cluster are in different regions, use the DistCp tool to copy the source cluster data to OBS, and use the OBS cross-region replication

function to copy the data to OBS in the region where the target cluster resides. For details, see **Cross-Region Replication**. If DistCp is used, permission, owner, and group information cannot be set for files on OBS. In this case, you need to export and copy the HDFS metadata while exporting data to prevent the loss of HDFS file property information.

Migrating Data from an Offline Cluster to a Cloud

You can use the following way to migrate data from an offline cluster to the cloud.

- Direct Connect

  Create a **Direct Connect** between the source cluster and target cluster, enable the network between the offline cluster egress gateway and the online VPC, and use DistCp to copy the data by referring to **Same Region**.

2.1.4 Restoring Data

HDFS File Property Restoration

Based on the exported permission information, run the HDFS commands in the background of the destination cluster to restore the file permission and owner and group information.

```
$HADOOP_HOME/bin/hdfs dfs -chmod <MODE> <path>
$HADOOP_HOME/bin/hdfs dfs -chown <OWNER> <path>
```

Hive Metadata Restoration

Install Sqoop and run the Sqoop command in the destination cluster to import the exported Hive metadata to DBService in the MRS cluster.

```
$Sqoop_Home/bin/sqoop export --connect jdbc:postgresql://<ip>:20051/hivemeta --table <table_name> --username hive -password <passwd> --export-dir <export_from>
```

- *$Sqoop_Home*: Sqoop installation directory in the destination cluster
- *<ip>*: IP address of the database in the destination cluster
- *<table_name>*: Name of the table to be restored
- *<passwd>*: Password of user **hive**
- *<export_from>*: HDFS address of the metadata in the destination cluster

NOTE

Commands carrying authentication passwords pose security risks. Disable historical command recording before running such commands to prevent information leakage.

HBase Table Reconstruction

Restart the HBase service of the destination cluster to make data migration take effect. During the restart, HBase loads the data in the current HDFS and regenerates metadata. After the restart is complete, run the following command on the Master node client to load the HBase table data:

```
$HBase_Home/bin/hbase hbck -fixMeta -fixAssignments
```


After the command is executed, run the following command repeatedly to check the health status of the HBase cluster until the health status is normal:

```
hbase hbck
```

2.2 Information Collection Before Data Migrated to MRS

Offline big data migration is flexible. Before the migration, you need to understand the detailed information about the existing cluster to better make migration decisions.

Service Information Survey

1. Architecture of the big data platform and services.
2. The data flow diagram (including peak and average traffic) of the big data platform and services is used to identify the data access source of the platform. Data inflow mode of the big data platform (real-time data reporting and batch data extraction) and data flow direction of the analysis platform. Data flow direction between components on the platform. For example, which component is used to collect data, how the collected data flows to the next-layer component, which component is used to store data, and the workflow during data processing.
3. Job Type: Hive SQL, Spark SQL, and Spark Python. For details about whether to use the third-party package of MRS, see the **MRS Application Development Sample**.
4. Dispatching system: The dispatching system needs to interconnect with the MRS cluster.
5. After the migration, the service interruption duration is allowed during service cutover to identify the platform service priority. Identify the services that cannot be interrupted during the migration, the services that can be interrupted for a short time, and the acceptable migration duration of the entire service migration to decide the migration sequence.
6. The deployment requirements of a client.
7. Service execution time segment and peak time segment.
8. Number and function division of big data clusters, and service model of the analysis platform, including the services that each cluster or component is responsible for and types of data that each cluster or component processes. For example, the components used to process real-time and offline data, data format type, and compression algorithm.

Collecting Basic Cluster Information

Table 2-1 Basic cluster information

Parameter	Value	Description
Cluster name	-	-

Parameter	Value	Description
Cluster version	-	Versions of clusters such as MRS, CDM, and FusionInsight.
Node quantity and specifications	-	Mandatory. Know the number of existing cluster nodes and node specifications. If the cluster uses heterogeneous hardware, enter multiple specifications and the number of nodes. For details, see **Table 2-2**. Example: NameNode and ResourceManager are deployed on two 32 U 64 GB servers. The HiveServer is deployed on two 32 U 64 GB servers. DataNodes and NodeManagers are deployed on 20 16 U 32 GB servers.
Enable Kerberos authentication	-	Mandatory: Yes or No
Permission control and description	-	Mandatory. Investigate the components and configurations for which ACL permission control is enabled. Involved components: Yarn, Hive, Impala and HBase. Use Ranger, Sentry, or open-source component permission capabilities to control permissions.
Region/AZ	-	Cloud resource parameters
VPC	-	Cloud resource parameters
Subnet	-	Cloud resource parameters
Security group	-	Cloud resource parameters

Table 2-2 Hardware information survey table

Node Group	CPU and Memory	Disk and Network (by Node Group)		HDFS			Yarn	
-	-	Disk information (data disk size, disk I/O, current disk usage, and I/O status)	Network (NIC bandwidth, network read/write speed, and peak value)	Name Node	DataNode	Joural Node	Node Manager	ResourceManager
master 1	(16 U 64 GB)	-	-	1	-	1	-	1
master 2	(16 U 64 GB)	-	-	1	-	1	-	1
master 3	(16 U 64 GB)	-	-	-	-	1	-	-
Core-group 1	(32 U 128 GB)*Quantity	-	-	-	1	-	1	-
Core-group 1	(32 U 129 GB)	-	-	-	-	-	-	-
Core-group 1	(32 U 130 GB)	-	-	-	-	-	-	-

Big data component information

Compare the big data component information with the version information of the planned new big data cluster to identify the possible impact of version differences on the migration process and the impact on service compatibility after the migration.

Table 2-3 Big data component information

Component	Source Cluster Version	Target Cluster Version (Example: MRS 1.9.2)	Description
HDFS/OBS (or other file storage systems)	Hadoop 2.8.3	Hadoop 2.8.3	-
Hive	1.2.1	2.3.3	Database for storing metadata: MySQL
HBase	1.3.1	1.3.1	-
Spark	2.2.2	2.2.2	-
Kafka	1.1.0	1.1.0	-
Oozie	2.x	Self-built	-
mysql	5.7.1	RDS	-
Flink	1.7	1.7	-
...	...	...	-

Statistics on Inventory Data to Be Migrated and Data Volume

If HDFS is used as the file storage system, you can run the **hadoop fs -du -h /user/test** command to collect statistics on the file size in the path.

Table 2-4 Existing data volume statistics

Component	Path of the Data to Be Migrated	Data Volume	Number of Files or Tables
HDFS/OBS (or other file storage systems)	/user/helloworld	xx	Total: *xxxx* files Number of files smaller than 2 MB: *xxx*
Hive	/user/hive/warehouse/	xx	Number of tables: *xxx*
HBase	/hbase	xx	Number of tables: *xx* Number of regions: *xx*

Statistics on New Data Volume Every Day

Evaluate the data growth rate (by day or hour) based on the amount of new data generated every day. After the first full data migration, you can periodically migrate new data in the old cluster until the final service migration is complete.

Table 2-5 New data volume statistics

Component	Path of the Data to Be Migrated	New Data Volume
HDFS/OBS (or other file storage systems)	/user/helloworld	xx
Hive	/user/hive/warehouse/	xx
HBase	/hbase	xx

Network Egress Bandwidth Capability

- Maximum network bandwidth and private line bandwidth that can be used for data migration (adjustable or not)
- Period during which a data migration job can be executed every day

Collecting Streaming Kafka Cluster Information

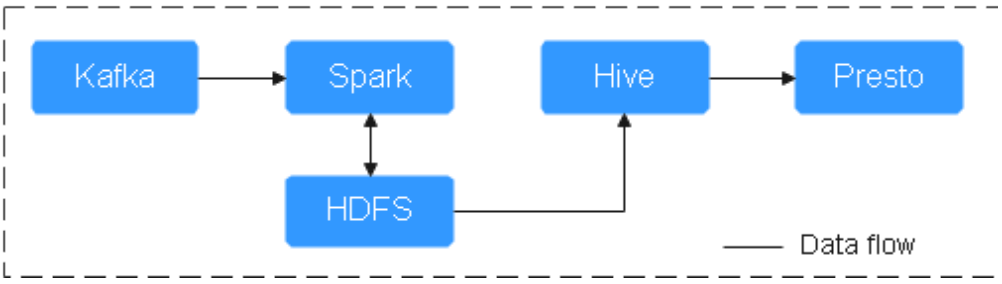
Table 2-6 Streaming Kafka cluster information

Item	Description
Number and names of Kafka topics	-
Local data temporary storage duration of Kafka. If the configuration of each topic is different, collect the data by topic.	-
Number of copies and partitions of each topic. (The default value is **2**. More copies indicate more reliable data and more disk space is consumed.) If the configuration of each topic is different, collect data by topic.	-
Kafka production and consumption traffic, which is refined to the topic level.	-
Configure acks in ACK of the Kafka client	-

Data Migration Model Example

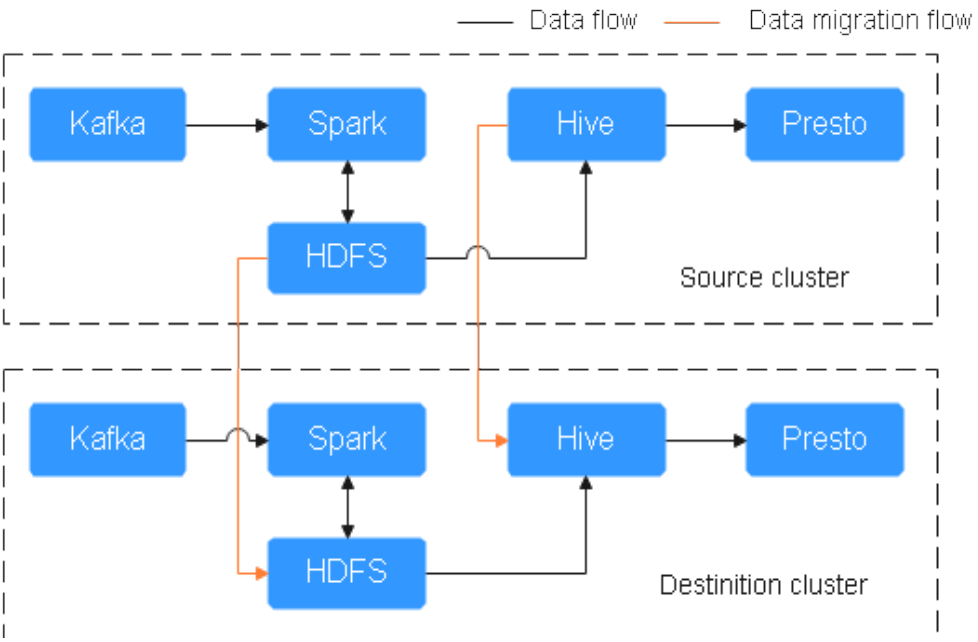
- In the customer service system diagram of an offline analysis platform, Spark Streaming consumes Kafka data and saves the data to HDFS. After small files are merged in HDFS, Hive Load loads the data to Hive tables. Operation personnel can use Presto to query Hive data.

  Figure 2-1 Service diagram of the source cluster



- For offline big data platforms, including HDFS and Hive, service programs of Kafka, Spark Streaming, HDFS, Hive, and Presto must be deployed on the destination cluster.

  Figure 2-2 Migration diagram

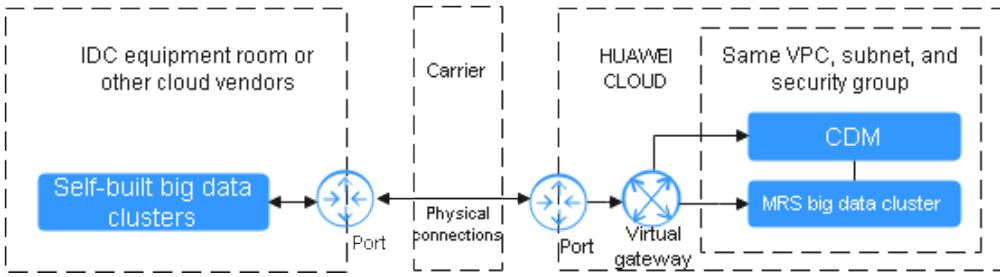


2.3 Preparing the Network Before Data Migration to MRS

During big data migration, ensure that the network connection between the source cluster and the destination cluster is normal. For example, when you run the Hadoop **distcp** command to copy data across clusters, all DataNodes must be connected. Based on different migration scenarios, you need to use different methods to connect the network between the two clusters.

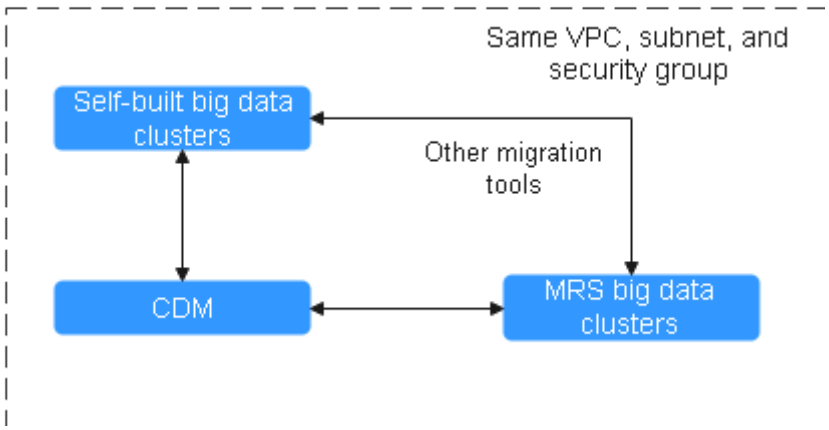
- You can migrate data from your local data center to an MRS cluster on HUAWEI CLOUD and use Direct Connect to establish a dedicated connection between your local data center and a VPC on the cloud. You can create a **Direct Connect** service on HUAWEI CLOUD or use a third-party Direct Connect service to connect to HUAWEI CLOUD.

  Figure 2-3 Offline data center migration

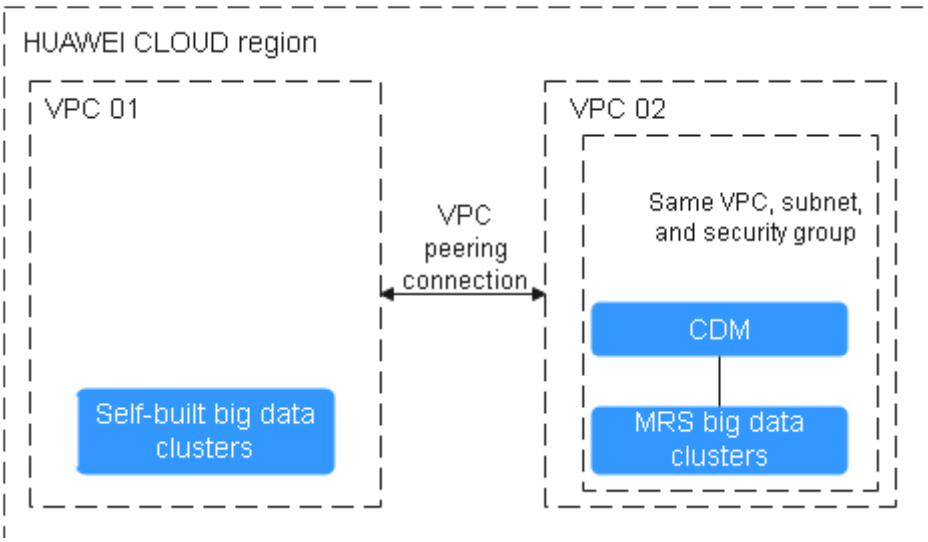
- If you want to migrate your created big data clusters (or MRS clusters of earlier versions) to HUAWEI CLOUD MRS clusters, you need to switch to the same region and VPC subnet. In this way, your created clusters and MRS clusters can use the same security group, VPC, and subnet to ensure network connectivity.

  Figure 2-4 Online migration in the same region and VPC

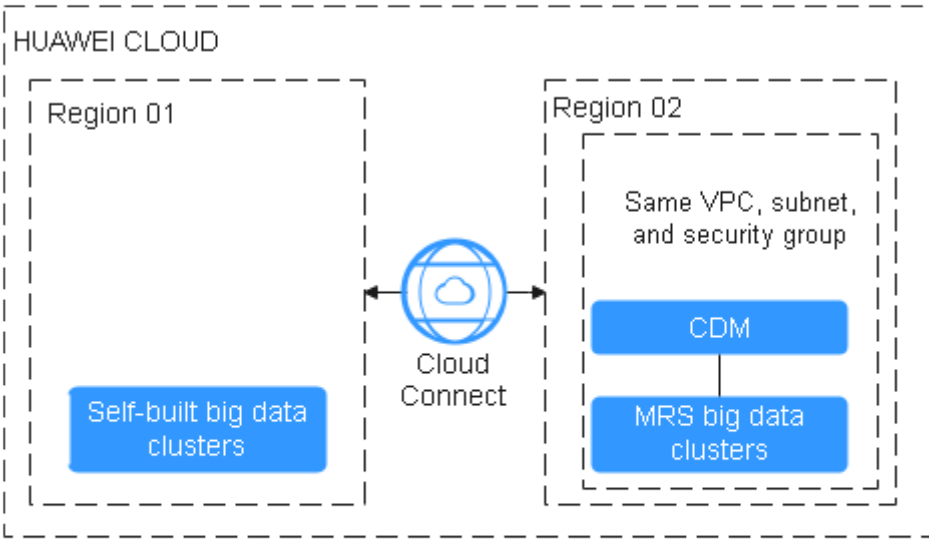
- The customer builds a big data cluster (or an MRS cluster of an earlier version) on HUAWEI CLOUD and needs to migrate it to an MRS cluster on HUAWEI CLOUD. The two clusters are in the same region but use different VPC subnets. You need to use a **VPC peering connection** to configure network connectivity.

Figure 2-5 Online migration between different VPCs in the same region



- You need to migrate your big data clusters (or MRS clusters of earlier versions) created on HUAWEI CLOUD to MRS clusters on HUAWEI CLOUD. In different regions, you can use **Cloud Connect (CC)** to build cross-region VPC network connections.

Figure 2-6 Online migration between different regions



2.4 Migrating Data from Hadoop to MRS

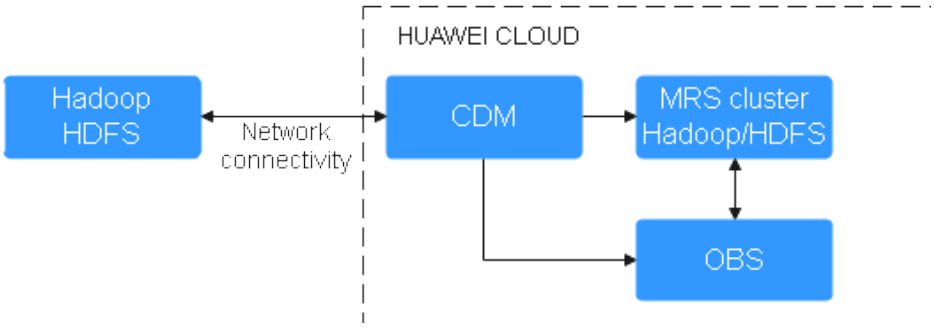
Scenario

This section describes how to migrate data from offline IDCs or public cloud Hadoop clusters to Huawei Cloud MRS. The data volume can be tens of TBs or less.

This section uses **Huawei Cloud CDM** 2.9.1.200 as an example to describe how to migrate data.

For details about the data sources supported by CDM, see Supported Data Sources. If the data source is Apache HDFS, the recommended version is 2.8.*X* or 3.1.*X*. Before performing the migration, ensure that the data source support migration.

Figure 2-7 Hadoop data migration



Solution Advantages

- Easy-to-use: The wizard-based development interface frees you from programming but helps you develop migration tasks by simple configurations in minutes.
- High migration efficiency: The performance of data migration and transmission is enhanced based on the distributed computing framework. Data write performance of specific data sources is optimized to improve data migration efficiency.
- Real-time monitoring: During the migration, automatic real-time monitoring, alarms, and notifications can be performed.

Impact on the System

Migrating large volumes of data has high requirements on network communication. When a migration task is executed, other services may be affected. You are advised to migrate data during off-peak hours.

Procedure

Step 1 Log in to the CDM management console.

Step 2 **Create a CDM cluster.** The security group, VPC, and subnet of the CDM cluster must be the same as those of the destination cluster to ensure that the CDM cluster can communicate with the MRS cluster.

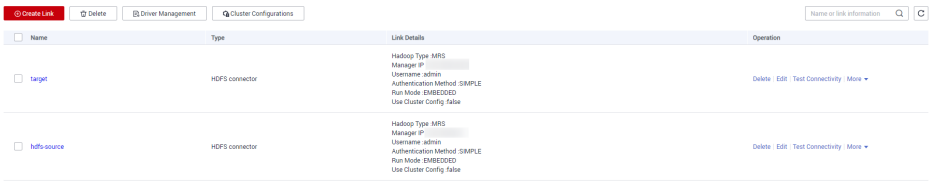
Step 3 On the **Cluster Management** page, locate the row containing the desired cluster and click **Job Management** in the **Operation** column.

Step 4 On the **Links** tab page, click **Create Link**.

Step 5 Add two HDFS links to the source cluster and destination cluster, respectively. For details, see **Creating Links**.

Select a link type based on the actual cluster. For an MRS cluster, select **MRS HDFS**. For a self-built cluster, select **Apache HDFS**.

Figure 2-8 HDFS link



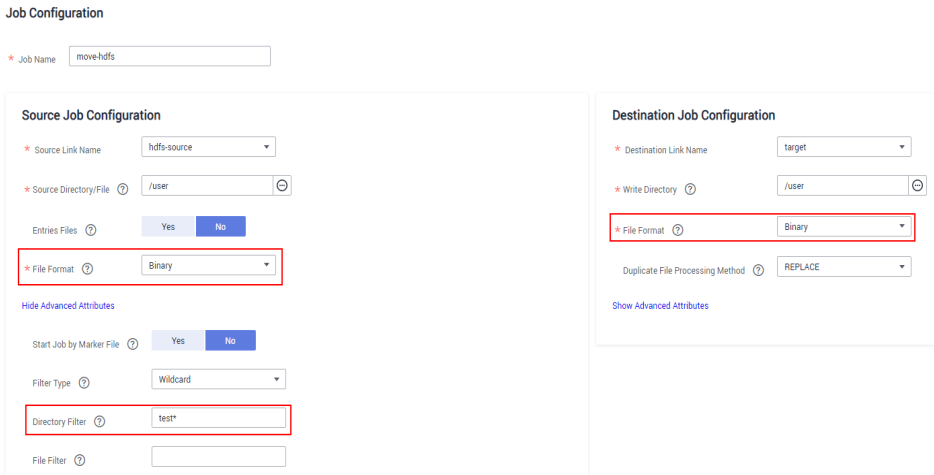
Step 6 On the **Table/File Migration** tab page, click **Create Job**.

Step 7 Select the source and destination links.

- **Job Name**: Enter a custom job name, which contains 1 to 256 characters consisting of letters, underscores (_), and digits.
- **Source Link Name**: Select the HDFS link of the source cluster. Data is exported from this link when the job is running.
- **Destination Link Name**: Select the HDFS link of the destination cluster. Data is imported to this link when the job is running.

Step 8 Configure source job parameters by referring to **From HDFS**. You can set **Directory Filter** and **File Filter** to specify the directories and files to be migrated. For example, if **Path Filter** is set to **test***, files in the **/user/test*** folder will be migrated. In this scenario, **File Format** is fixed to **Binary**.

Figure 2-9 Configuring job parameters

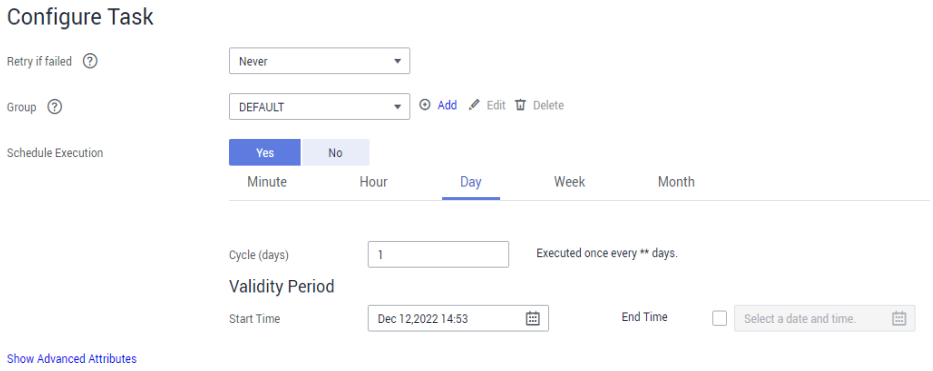


Step 9 Configure destination job parameters by referring to **To HDFS**.

Step 10 Click **Next**. The task configuration page is displayed.

- If you need to periodically migrate new data to the destination cluster, configure a scheduled task on this page. Alternatively, you can configure a scheduled task later by referring to **Step 14**.
- If no new data needs to be migrated periodically, skip the configurations on this page and click **Save**.

Figure 2-10 Task configuration



Step 11 Choose **Job Management** and click the **Table/File Migration** tab. Click **Run** in the **Operation** column of the job to be executed to start migrating HDFS data. Wait until the job execution is complete.

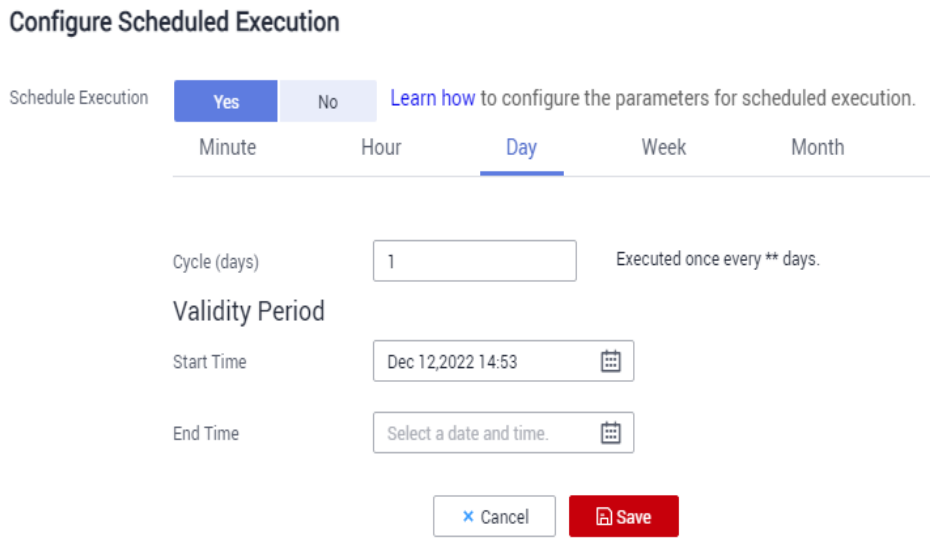
Step 12 Log in to the active management node of the destination cluster.

Step 13 Run the **hdfs dfs -ls -h /user/** command to view the migrated files in the destination cluster.

Step 14 (Optional) If new data in the source cluster needs to be periodically migrated to the destination cluster, configure a scheduled task for incremental data migration until all services are migrated to the destination cluster.

1. On the **Cluster Management** page of the CDM console, choose **Job Management** and click the **Table/File Migration** tab.
2. In the **Operation** column of the migration job, click **More** and select **Configure Scheduled Execution**.
3. Enable the scheduled job execution function, set the execution cycle based on service requirements and the end time of the validity period to the time after all services are migrated to the new cluster.

Figure 2-11 Scheduling job execution



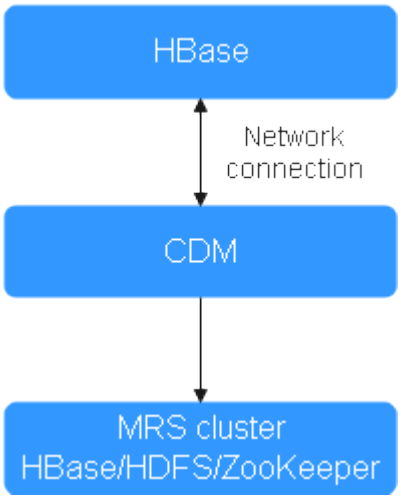
----End

2.5 Migrating Data from HBase to MRS

Scenario

This section describes how to migrate data from offline IDCs or public cloud HBase clusters to Huawei Cloud MRS. The data volume can be tens of TBs or less. This section uses **Huawei Cloud CDM** 2.9.1.200 as an example to describe how to migrate data.

Figure 2-12 HBase data migration



HBase stores data in HDFS, including HFile and WAL files. The **hbase.rootdir** configuration item specifies the HDFS path. By default, data is stored in the **/hbase** folder on MRS.

Some mechanisms and tool commands of HBase can also be used to migrate data. For example, you can migrate data by exporting snapshots, exporting/importing data, and CopyTable. For details, see the Apache official website.

This document describes how to migrate HBase data using Huawei Cloud CDM.

For details about the data sources supported by CDM, see Supported Data Sources. If the data source is Apache HBase, the recommended version is 2.1.*X* or 1.3.*X*. Before performing the migration, ensure that the data source supports migration.

Solution Advantages

Scenario-based migration migrates snapshots and then restores table data to speed up migration.

Impact on the System

Migrating large volumes of data has high requirements on network communication. When a migration task is executed, other services may be affected. You are advised to migrate data during off-peak hours.

Full Data Migration

Step 1 Log in to the CDM management console.

Step 2 **Create a CDM cluster.** The security group, VPC, and subnet of the CDM cluster must be the same as those of the destination cluster to ensure that the CDM cluster can communicate with the MRS cluster.

Step 3 On the **Cluster Management** page, locate the row containing the desired cluster and click **Job Management** in the **Operation** column.

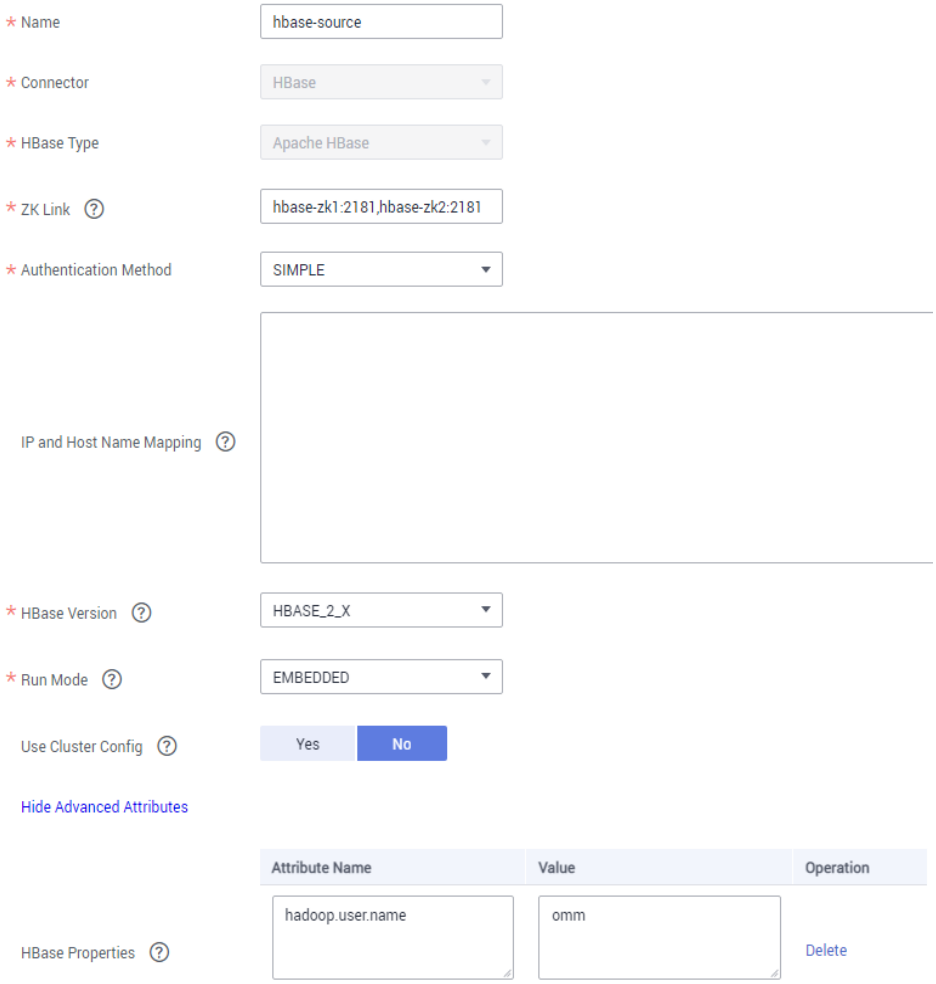
Step 4 On the **Links** tab page, click **Create Link**.

Step 5 Create a link to the source cluster by referring to **Creating a CDM Link**. Select a connector type based on the actual cluster, for example, **Apache HBase**.

NOTE

(Optional) Use a user with high permissions to migrate HBase. For example, click **Show Advanced Attributes** and add user **hadoop.user.name** = *Username* (for example, **omm**).

Figure 2-13 Link to the source cluster



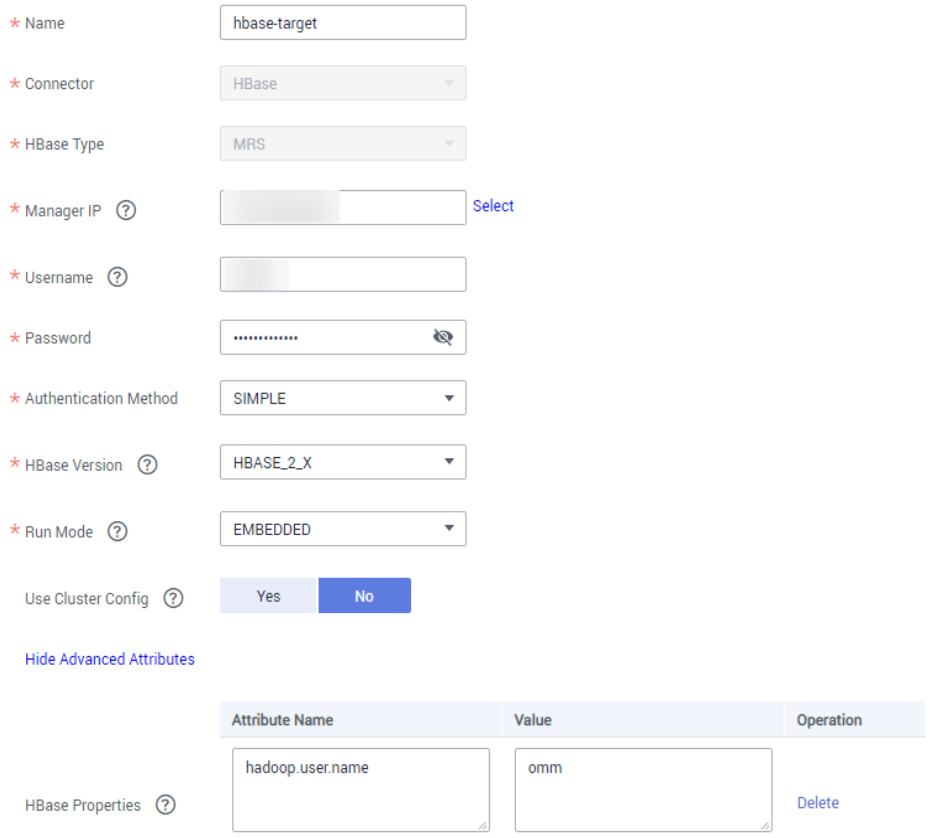
Step 6 On the **Links** tab page, click **Create Link**.

Step 7 Create a link to the destination cluster by referring to **Creating a CDM Link**. Select a connector type based on the actual cluster, for example, **MRS HBase**.

NOTE

(Optional) Use a user with high permissions to migrate HBase. For example, click **Show Advanced Attributes** and add user **hadoop.user.name** = *Username* (for example, **omm**).

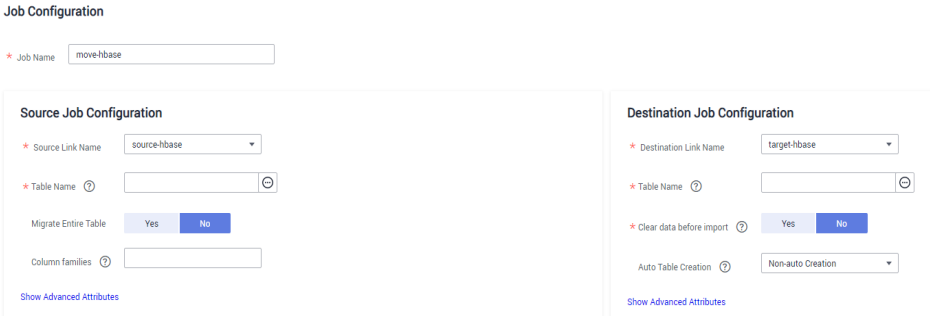
Figure 2-14 Link to the destination cluster



Step 8 Choose **Job Management** and click the **Table/File Migration** tab. Then, click **Create Job**.

Step 9 In the job creation dialog box, configure the job name, source job parameters, and destination job parameters, select the data table to be migrated, and click **Next**.

Figure 2-15 HBase job configuration



Step 10 Configure the mapping between the source fields and destination fields and click **Next**.

Step 11 On the task configuration page that is displayed, click **Save** without any modification.

Step 12 Choose **Job Management** and click **Table/File Migration**. Locate the row containing the job to run and click **Run** in the **Operation** column to start migrating HBase data.

Step 13 After the migration is complete, you can run the same query statement in the source and destination clusters to compare the query results.

Example:

- Query the number of records in the BTable table on the source and destination clusters to check whether the number of data records is the same. Add the **--endtime** parameter to eliminate the impact of data updates on the source cluster during the migration.

  Hbase org.apache.hadoop.hbase.mapreduce.RowCounter *BTable* **--endtime=1587973835000**

  Figure 2-16 Querying the number of records in the BTable table

```
2020-04-27 16:15:09,500 INFO  [main] mapreduce.Job:  map 56% reduce 0%
2020-04-27 16:15:17,528 INFO  [main] mapreduce.Job:  map 67% reduce 0%
2020-04-27 16:15:25,566 INFO  [main] mapreduce.Job:  map 89% reduce 0%
2020-04-27 16:15:30,584 INFO  [main] mapreduce.Job:  map 100% reduce 0%
2020-04-27 16:15:30,592 INFO  [main] mapreduce.Job: Job job_1587471561730_0063 completed successfully
2020-04-27 16:15:30,653 INFO  [main] mapreduce.Job: Counters: 46
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=2163033
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=2474
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=9
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=0
        Job Counters
                Killed map tasks=3
                Launched map tasks=12
                Data-local map tasks=10
                Rack-local map tasks=2
                Total time spent by all maps in occupied slots (ms)=1442868
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=360717
                Total vcore-milliseconds taken by all map tasks=360717
                Total megabyte-milliseconds taken by all map tasks=738748416
        Map-Reduce Framework
                Map input records=20646264
                Map output records=0
                Input split bytes=2474
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=1674
                CPU time spent (ms)=86120
                Physical memory (bytes) snapshot=3175682048
                Virtual memory (bytes) snapshot=33582198784
                Total committed heap usage (bytes)=1950875648
        HBase Counters
                BYTES_IN_REMOTE_RESULTS=0
                BYTES_IN_RESULTS=3280268799
                MILLIS_BETWEEN_NEXTS=280535
                NOT_SERVING_REGION_EXCEPTION=0
                NUM_SCANNER_RESTARTS=0
                NUM_SCAN_RESULTS_STALE=0
                REGIONS_SCANNED=9
                REMOTE_RPC_CALLS=0
                REMOTE_RPC_RETRIES=0
                ROWS_FILTERED=0
                ROWS_SCANNED=20646264
                RPC_CALLS=206485
                RPC_RETRIES=0
        org.apache.hadoop.hbase.mapreduce.RowCounter$RowCounterMapper$Counters
                ROWS=20646264
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=0
[root@node-masterlieDB etc]#
```

- Use **scan '** *BTable* **', {TIMERANGE=>[1587973235000, 1587973835000]}** of HBase shell to query data in a specified period for comparison.

----End

Incremental Data Migration

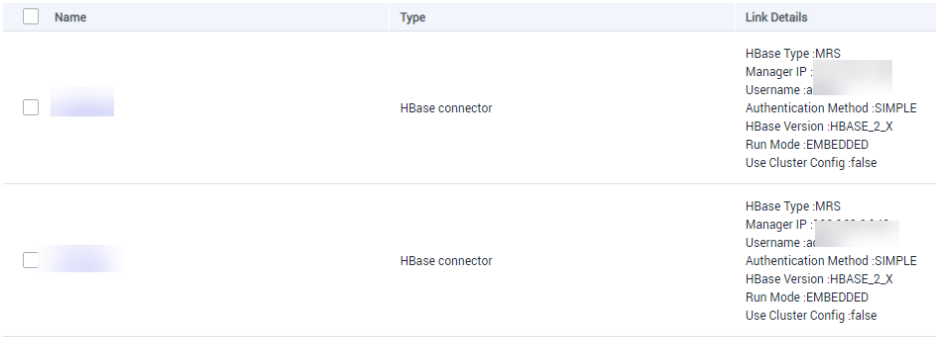
If new data exists in the source cluster before the service cutover, you need to periodically migrate the new data to the destination cluster. Generally, the data volume updated every day is at the GB level. You can use the **Entire DB migration** function of CDM to migrate new HBase data every day.

If the **Entire DB Migration** function of CDM is used, the deleted data in the source HBase cluster cannot be synchronized to the destination cluster.

The HBase connector for scenario migration cannot be shared with that for entire database migration. Therefore, a new HBase connector is required.

Step 1 Repeat **Step 1** to **Step 7** in **Full Data Migration** to create two HBase connectors. Select **MRS HBase** and **Apache HBase** as the connector type for the source cluster and destination cluster, respectively.

Figure 2-17 HBase incremental migration link

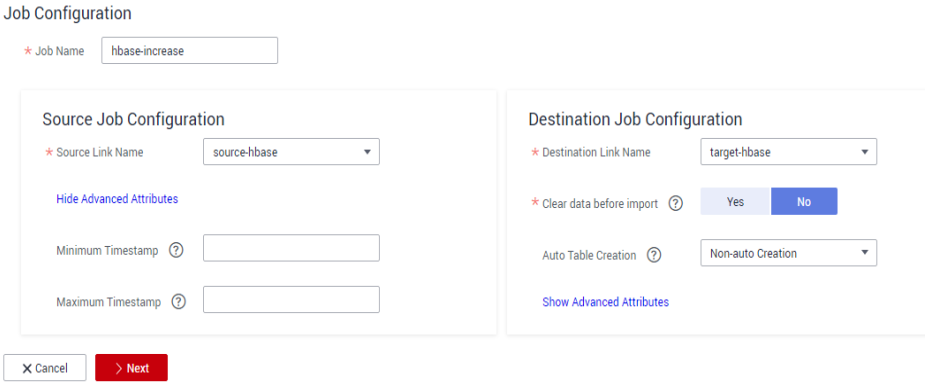


Step 2 Choose **Job Management** > **Entire DB Migration**, and click **Create Job**.

Step 3 On the job parameter configuration page, configure job parameters and click **Next**.

- **Job Name**: Enter a user-defined job name, for example, **hbase-increase**.
- **Source Job Configuration**: Set **Source Link Name** to the name of the link to the source cluster created in **Step 1**, and click **Show Advanced Attributes** to configure the time range for data migration.
- **Destination Job Configuration**: Set **Destination Link Name** to the name of the link to the destination cluster created in **Step 1**. Leave other parameters blank.

Figure 2-18 HBase incremental migration job configuration



Step 4 Select the data table to be migrated, click **Next**, and click **Save**.

Step 5 Choose **Job Management** > **Entire DB Migration** and click **Run** in the **Operation** column of the job to be executed to start HBase incremental data migration.

----End

2.6 Migrating Data from Hive to MRS

Scenario

This section describes how to migrate data from offline IDCs or public cloud Hive clusters to Huawei Cloud MRS. The data volume can be tens of TBs or less. This section uses **Huawei Cloud CDM** 2.9.1.200 as an example to describe how to migrate data.

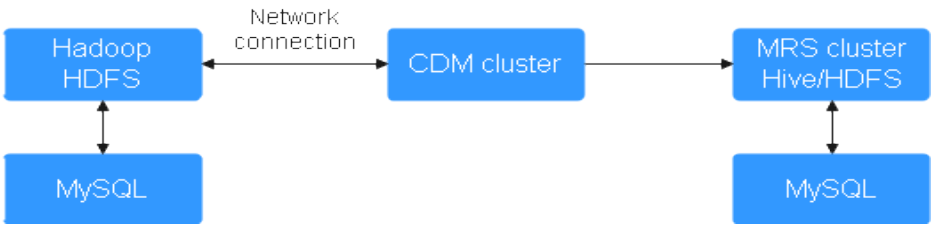
Hive data migration consists of two parts:

- Hive metadata, which is stored in the databases such as MySQL. By default, the metadata of the MRS Hive cluster is stored in MRS DBService (Huawei GaussDB database). You can also use RDS (MySQL) as the external metadata database.
- Hive service data, which is stored in HDFS or OBS

You can use the scenario migration function of Huawei Cloud CDM to migrate Hive data with one click.

For details about the data sources supported by CDM, see Supported Data Sources. If the data source is Apache Hive, the recommended version is 1.2.*X* or 3.1.*X*. Version 2.*x* is not supported. Before performing the migration, ensure that the data source supports migration.

Figure 2-19 Hive data migration



Solution Advantages

Scenario-based migration migrates snapshots and then restores table data to speed up migration.

Impact on the System

Migrating large volumes of data has high requirements on network communication. When a migration task is executed, other services may be affected. You are advised to migrate data during off-peak hours.

Procedure

Step 1 Log in to the CDM management console.

Step 2 **Create a CDM cluster.** The security group, VPC, and subnet of the CDM cluster must be the same as those of the destination cluster to ensure that the CDM cluster can communicate with the MRS cluster.

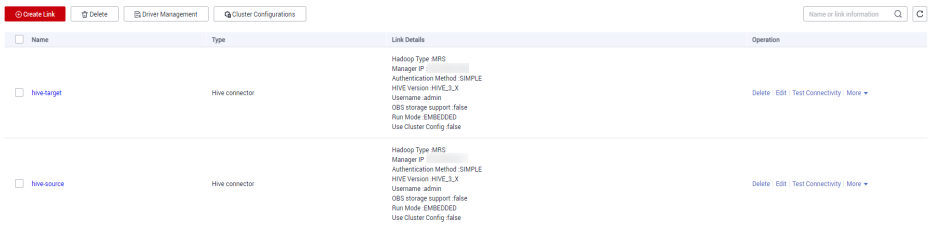
Step 3 On the **Cluster Management** page, locate the row containing the desired cluster and click **Job Management** in the **Operation** column.

Step 4 On the **Links** tab page, click **Create Link**.

Step 5 Create links to the source and destination clusters by referring to **Creating Links**. Select **MRS Hive** as the connector type.

Set the connector type based on the actual cluster. For an MRS cluster, select **MRS Hive**. For a self-built cluster, select **Apache Hive**.

Figure 2-20 Hive link



Step 6 Create a storage database after data migration in the destination cluster.

Step 7 Choose **Job Management** and click the **Table/File Migration** tab. Then, click **Create Job**.

Step 8 In the job configuration dialog box that is displayed, configure the job name, select the data links created in **Step 5** as the source link and destination link, select the names of the database and table to be migrated, and click **Next**.

Figure 2-21 Hive job configuration

Step 9 Configure the mapping between the source fields and destination fields and click **Next**.

Step 10 On the task configuration page that is displayed, click **Save** without any modification.

Step 11 Choose **Job Management** and click **Table/File Migration**. Locate the row containing the job to run and click **Run** in the **Operation** column to start migrating Hive data.

Step 12 After the migration is complete, you can run the same query statement in the source and destination clusters to compare the query results.

For example, query the number of records in the **catalog_sales** table in the destination cluster and source cluster to check whether the number of data records is the same.

select count(*) from *catalog_sales*;

Figure 2-22 Data records of the source cluster

```
1 row selected (0.098 seconds)
0: jdbc:hive2://192.168.0.216:2181,192.168.0.> select count(*) from catalog_sales;
INFO  : Compiling command(queryId=omm_20200424173337_aaf0d972-b100-4f4a-87c4-959fbd6a2c4f): select count(*) from catalog_sales
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:_c0, type:bigint, comment:null)], properties:null)
INFO  : EXPLAIN output for queryid omm_20200424173337_aaf0d972-b100-4f4a-87c4-959fbd6a2c4f : STAGE DEPENDENCIES:
  Stage-0 is a root stage [FETCH]

STAGE PLANS:
  Stage: Stage-0
    Fetch Operator
      limit: 1
      Processor Tree:
        ListSink


INFO  : Completed compiling command(queryId=omm_20200424173337_aaf0d972-b100-4f4a-87c4-959fbd6a2c4f); Time taken: 0.263 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=omm_20200424173337_aaf0d972-b100-4f4a-87c4-959fbd6a2c4f): select count(*) from catalog_sales
INFO  : Completed executing command(queryId=omm_20200424173337_aaf0d972-b100-4f4a-87c4-959fbd6a2c4f); Time taken: 0.001 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+-----------+
|    _c0    |
+-----------+
| 43204059  |
+-----------+
1 row selected (0.275 seconds)
0: jdbc:hive2://192.168.0.216:2181,192.168.0.>
```

Figure 2-23 Data records of the destination cluster

```
INFO  : Completed compiling command(queryId=omm_20200424173329_53ad05b4-e097-44c4-9a8f-9f77e7087888); Time taken: 0.845 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=omm_20200424173329_53ad05b4-e097-44c4-9a8f-9f77e7087888): select count(*) from catalog_sales
INFO  : Query ID = omm_20200424173329_53ad05b4-e097-44c4-9a8f-9f77e7087888
INFO  : Total jobs = 1
INFO  : Launching Job 1 out of 1
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Subscribed to counters: [] for queryId: omm_20200424173329_53ad05b4-e097-44c4-9a8f-9f77e7087888
INFO  : Session is already open
INFO  : Dag name: select count(*) from catalog_sales (Stage-1)
INFO  : Tez session was closed. Reopening...
INFO  : Session re-established.
INFO  : Session re-established.
INFO  : Status: Running (Executing on YARN cluster with App id application_1587628367568_0006)

INFO  : Completed executing command(queryId=omm_20200424173329_53ad05b4-e097-44c4-9a8f-9f77e7087888); Time taken: 22.54 seconds
INFO  : OK
INFO  : Concurrency mode is disabled, not creating a lock manager
+-----------+
|    _c0    |
+-----------+
| 43204059  |
+-----------+
1 row selected (29.898 seconds)
0: jdbc:hive2://192.168.0.186:2181,192.168.0.>
```

Step 13 (Optional) If new data in the source cluster needs to be periodically migrated to the destination cluster, perform the migration based on the data adding mode. Configure a scheduled task to migrate incremental data until all services are migrated to the destination cluster.

- If no table is added or deleted and the data structure of the existing table is not modified and only the Hive table data is modified: you only need to migrate the files stored on HDFS or OBS. For details about how to migrate data, see the description about the new data migration method in **Migrating Data from Hadoop to MRS**.
- If a Hive table is added, choose **Job Management** and click the **Table/File Migration** tab. Click **Edit** in the **Operation** column of the Hive migration job and select the new data table for data migration.

- If a Hive table is deleted or the data structure of an existing table is modified, manually delete the table from the destination cluster or manually update the table structure.

----End

2.7 Using BulkLoad to Import Data to HBase in Batches

Application Scenarios

When batch importing a large amount of data to HBase, you have many choices, for example, calling the **put** method of HBase to insert data or using MapReduce to load data from HDFS. However, the two methods cause high pressure on the RegionServer and consume a large number of CPU and network resources because of frequent flush, compact, and split operations of HBase, thereby resulting in low efficiency.

This practice describes how to import local data to HBase in batches using BulkLoad after you create an MRS cluster. This method greatly improves the write efficiency and reduces the write pressure on RegionServer nodes.

Solution Architecture

HBase provides a data import tool called BulkLoad, which imports and directly writes data to underlying data files and WAL logs, greatly improving data loading speed and efficiency.

BulkLoad uses MapReduce jobs to directly convert data into HFiles that comply with the internal data format of HBase, and then loads the generated StoreFiles to the corresponding nodes in a cluster. This method requires no flush, compact, or split operations, occupies no region resources, and generates little write requests. Fewer CPU and network resources are required.

Inapplicable scenarios of BulkLoad:

- Large amounts of data needs to be loaded to HBase in the one-off manner.
- When data is loaded to HBase, requirements on reliability are not high and WAL files do not need to be generated.
- When the **put** method is used to load large amounts of data to HBase, data loading and query will be slow.
- The size of an HFile generated after data loading is similar to the size of HDFS blocks.

Creating an MRS Offline Query Cluster

1. Go to the **Buy Cluster** page.
2. Click the **Quick Config** tab and set configuration parameters.

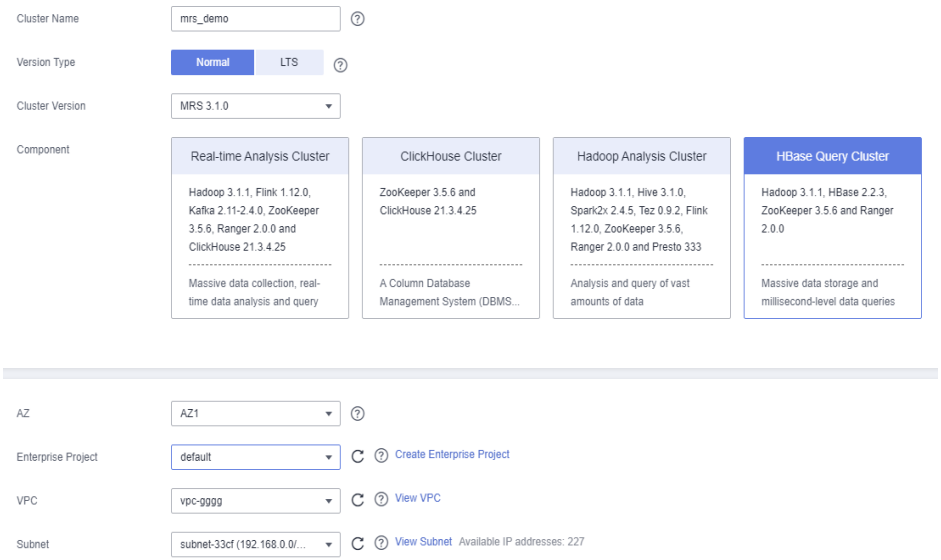
Table 2-7 Software configurations

Parameter	Description	Value
Region	MRS clusters in different regions cannot communicate with each other over an intranet. For lower network latency and quick resource access, select the region nearest to you.	CN-Hong Kong
Billing Mode	MRS provides two billing modes. • Yearly/Monthly • Pay-per-use A prepaid balance will be frozen. For details, see **Billing**.	Pay-per-use
Cluster Name	The cluster name must be unique. A cluster name can contain 1 to 64 characters. Only letters, digits, hyphens (-), and underscores (_) are allowed.	MRS_hbase
Cluster Type	Available cluster types are as follows: • Analysis cluster • Streaming cluster • Hybrid cluster • Custom cluster	Custom
Version Type	Available version types are as follows: • Normal • LTS	Normal
Cluster Version	Available MRS versions	MRS 3.1.0
Component	MRS cluster components. For details about component versions supported by different versions of MRS clusters, see **List of MRS Component Versions**.	HBase Query Cluster

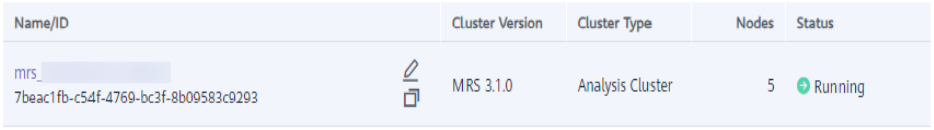
Parameter	Description	Value
AZ	An availability zone (AZ) is a physical area that uses independent power and network resources. AZs are physically isolated but interconnected through the internal network. This improves the availability of applications. You are advised to create clusters in different AZs.	AZ1
Enterprise Project	The **Enterprise Management** console is designed for resource management. It helps you manage cloud-based personnel, resources, permissions, and finance in a hierarchical manner, such as management of companies, departments, and projects.	default
VPC	A Virtual Private Cloud (VPC) is a secure, isolated, and logical network environment.	vpc-01
Subnet	A subnet provides dedicated network resources that are logically isolated from other networks for network security.	subnet-01
Kerberos Authentication	If Kerberos authentication is enabled for a cluster, check whether Kerberos authentication is required. If yes, click **Continue**. If no, click **Back** to disable Kerberos authentication and then create a cluster. After a cluster is purchased, this configuration cannot be modified.	Toggle the slider on.

Parameter	Description	Value
Username	The default value is **root/admin**. User **root** is used to remotely log in to ECS nodes, and user **admin** is used to access the cluster management page.	root/admin
Password	Password for users **root**/**admin**.	Set the password for logging in to the cluster management page and ECS node, for example, **Test!@12345**.
Confirm Password	-	Enter the password again.
Secure Communications	In an MRS cluster, you can provision, manage, and use big data components through the management console. Big data components are deployed in users' VPCs. To allow the MRS console to directly access big data components, you must enable the corresponding security group rules after granting authorization. This authorization process is called secure communications. If the secure communications function is not enabled, MRS clusters cannot be created.	Select **Enable**.

Figure 2-24 Creating an HBase query cluster



3. Click **Buy Now** and wait until the MRS cluster is created.



Importing Local Data to HDFS

1. Prepare a student information file **info.txt** on the local host.

   The fields include student ID, name, birthday, gender, and address. An example file is as follows:

```
20200101245, Zhang xx, 20150324, Male, City 1
20200101246, Li xx, 20150202, Male, City 2
20200101247, Yang xx, 20151101, Female, City 3
20200101248, Chen xx, 20150218, Male, City 4
20200101249, Li xx, 20150801, Female, City 5
20200101250, Wang xx, 20150315, Male, City 6
20200101251, Li xx, 20151201, Male, City 7
20200101252, Sun xx, 20150916, Female, City 8
20200101253, Lin xx, 20150303, Male, City 9
```

2. Log in to OBS Console, click **Parallel File Systems** in the navigation pane. On the displayed page, click **Create Parallel File System**, set the following parameters, and click **Create Now**.

Table 2-8 Parallel file system parameters

Parameter	Value
Region	CN-Hong Kong
File System Name	mrs-hbase
Data Redundancy Policy	Single-AZ storage
Policy	Private

Parameter	Value
Direct Reading	Disable
Enterprise Project	default
Tags	-

Click the name of the created bucket and click **Upload File** in the **Files** tab to upload the data file to the OBS bucket.

3. Switch back to the MRS console and click the name of the created MRS cluster. On the **Dashboard** page, click **Synchronize** next to **IAM User Sync**. The synchronization takes about five minutes.
4. Upload the data file to the HDFS.
   a. On the **Files** page, click the **HDFS File List** and go to the data storage directory, for example, **/tmp/test**.

      The **/tmp/test** directory is only an example. You can use any directory on the page or create a new one.

   b. Click **Import Data**.

      - **OBS Path**: Find the **info.txt** file in the created OBS parallel file system and click **Yes**.
      - **HDFS Path**: Select an HDFS path, for example, **/tmp/test**, and click **Yes**.

   c. Click **OK** and wait until the data file is imported.

Figure 2-25 Importing data

Creating an HBase Table

1. Log in to FusionInsight Manager of the cluster (if no elastic IP address is available, purchase one), create a user named **hbasetest**, and bind it to the user group **supergroup** and role **System_administrator**.

2. Download the cluster client, and install it, for example, in the **/opt/client** directory of the active master node. For details, see **Installing a Client**.

   You can also use the cluster client provided by the Master node. The installation directory is **/opt/Bigdata/client**.

3. Run the following commands to bind an elastic IP address to the active Master node, log in to the active Master node as user **root**, go to the directory where the client is located, and authenticate the user.

   cd /opt/client

   source bigdata_env

   kinit hbasetest

4. Run the **hbase shell** command to go to the HBase shell page.

   Plan the table name, rowkey, column family, and column of the HBase data table based on the imported data. Ensure that the rowkey is pre-split during table creation.

   Run the following command to create the **student_info** table:

   create 'student_info', {NAME => 'base',COMPRESSION => 'SNAPPY', DATA_BLOCK_ENCODING => 'FAST_DIFF'},SPLITS => ['1','2','3','4','5','6','7','8']

   - **NAME => 'base'**: Column family name of the HBase table
   - **COMPRESSION**: Compression mode
   - **DATA_BLOCK_ENCODING**: encoding algorithm
   - **SPLITS**: Region pre-splitting

5. Check whether the table is created and then exit the HBase shell page.

   list

Generating an HFile and Importing It to HBase

1. Create a custom template file, for example, **/opt/configuration_index.xml**. You can obtain the template file example from *Client installation directory*/**HBase/hbase/conf/index_import.xml.template**.

   vi /opt/configuration_index.xml

   An example template file is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
<!--The value of column_num must be consistent with the number of columns in the data file: 5 columns -->
 <import column_num="5" id="first">
  <columns>
   <column type="string" index="1">P_ID</column>
   <column type="string" index="2">P_NAME</column>
   <column type="string" index="3">P_BIRTH</column>
   <column type="string" index="4">P_GENDER</column>
   <column type="string" index="5">P_DISTRICT</column>
  </columns>
<!--reverse(P_BIRTH): Reverse the birth date to avoid hotspotting. -->
```



```
<!--substring(P_NAME,0,1): Filter out the student information based on the last name. -->
<!--substring(P_ID,0,6): Filter out the student information based on the first six digits of a student ID.
-->
   <rowkey>
    reverse(P_BIRTH)+'_'+substring(P_NAME,0,1)+'_'+substring(P_ID,0,6)
   </rowkey>
  <qualifiers>
  <!--The specified family must correspond to the column family of the table. -->
   <normal family="base">
    <qualifier column="P_ID">H_ID</qualifier>
    <qualifier column="P_NAME">H_NAME</qualifier>
    <qualifier column="P_BIRTH">H_BIRTH</qualifier>
    <qualifier column="P_GENDER">H_GENDER</qualifier>
    <qualifier column="P_DISTRICT">H_DISTRICT</qualifier>
   </normal>
  </qualifiers>
 </import>
</configuration>
```

2. Run the following commands to generate an HFile file:

   hbase com.huawei.hadoop.hbase.tools.bulkload.ImportData -Dimport.separator=',' -Dimport.hfile.output=/tmp/test/hfile /opt/configuration_index.xml student_info /tmp/test/info.txt

   - **-Dimport.separator**: indicates a separator.
   - **-Dimport.hfile.output**: indicates the output path of the execution result.
   - **/opt/configuration_index.xml**: indicates a custom template file.
   - **student_info**: indicates the name of the HBase table to be operated.
   - **/tmp/test/info.txt**: indicates the HDFS data directory to which data is to be uploaded in batches.
   - **com.huawei.hadoop.hbase.tools.bulkload.IndexImportData**: indicates **IndexImportData** used to create a secondary index during data import. If no secondary index needs to be created, **ImportData** is used.

   After the MapReduce job is successfully executed, an HFile file is generated in the output path.

   hdfs dfs -ls /tmp/test/hfile

   ```
   Found 2 items
   -rw-r--r--   3 hbasetest hadoop          0 2021-05-14 11:39 /tmp/test/hfile/_SUCCESS
   drwxr-xr-x   - hbasetest hadoop          0 2021-05-14 11:39 /tmp/test/hfile/base
   ```

3. Run the following command to import the HFile to the HBase table:

   hbase org.apache.hadoop.hbase.mapreduce.LoadIncrementalHFiles /tmp/test/hfile student_info

4. Run the following commands to go to the HBase shell page and view the table content:

   hbase shell

   scan 'student_info', {FORMATTER => 'toString'}

   ```
   ROW                                  COLUMN+CELL
    10115102_Yang_202001                 column=base:H_BIRTH, timestamp=2021-05-14T15:28:56.755,
   value=20151101
    10115102_Yang_202001                 column=base:H_DISTRICT,
   timestamp=2021-05-14T15:28:56.755, value=City3
    10115102_Yang_202001                 column=base:H_GENDER,
   timestamp=2021-05-14T15:28:56.755, value=female
    10115102_Yang_202001                 column=base:H_ID, timestamp=2021-05-14T15:28:56.755,
   value=20200101247
    10115102_Yang_202001                 column=base:H_NAME, timestamp=2021-05-14T15:28:56.755,
   value=Yang xx
    10215102_Li_202001                 column=base:H_BIRTH, timestamp=2021-05-14T15:28:56.755,
   ```



```
value=20151201
 10215102_Li_202001                column=base:H_DISTRICT, timestamp=2021-05-14T15:28:56.755,
value=City7
...
```

5. Analyze and process data based on the upper-layer applications of the big data platform after data is imported to the cluster.

2.8 Migrating MySQL Data to MRS Hive with CDM

Application Scenarios

MRS provides enterprise-class on-cloud big data clusters. It contains components such as HDFS, Hive, and Spark, and is tailored to analyze massive amounts of enterprise data.

Hive supports SQL to help users perform extraction, transformation, and loading (ETL) operations on large-scale data sets. Query on large-scale data sets takes a long time. In many scenarios, you can create Hive partitions to reduce the total amount of data to be scanned each time. This significantly improves query performance.

Hive partitions are implemented by using the HDFS subdirectory function. Each subdirectory contains the column names and values of each partition. If there are multiple partitions, there are many HDFS subdirectories. It is not easy to load external data to each partition of the Hive table without using tools.

With the Cloud Data Migration (CDM) service, you can easily load data of the external data sources (relational databases, object storage services, and file system services) to Hive partitioned tables.

This practice demonstrates how to use CDM to import MySQL data to the Hive partitioned table in an MRS cluster.

Suppose that there is a **trip_data** table in the MySQL database. The table stores cycling records such as the start time, end time, start sites, end sites, and rider IDs.

For details about the fields in the **trip_data** table, see **Figure 2-26**.

Figure 2-26 MySQL table fields

Column Name	#	Data Type
TripID	1	int(11)
Duration	2	int(11)
StartDate	3	timestamp
StartStation	4	varchar(64)
StartTerminal	5	int(11)
EndDate	6	timestamp
EndStation	7	varchar(64)
EndTerminal	8	int(11)
Bike	9	int(11)
SubscriberType	10	varchar(32)
ZipCodev	11	varchar(10)

The following describes how to use CDM to import data in the **trip_data** table of the MySQL database to the MRS Hive partitioned table. The procedure includes five steps:

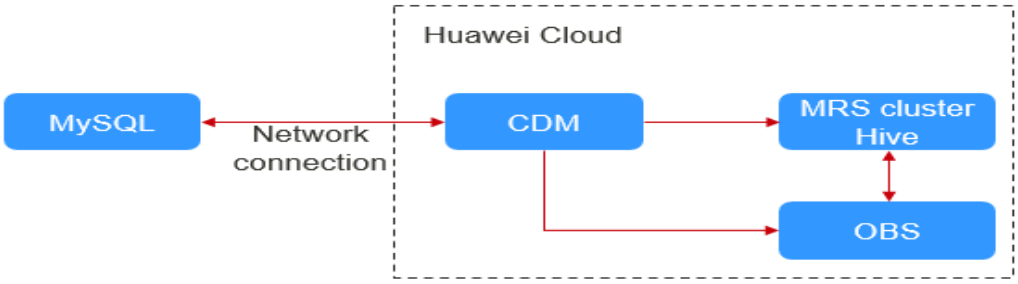
1. **Creating a Hive Partitioned Table on MRS Hive**
2. **Creating a CDM Cluster and Binding an EIP to the Cluster**
3. **Creating a MySQL Link**
4. **Creating a Hive Link**
5. **Creating a Migration Job**

Solution Architecture

Based on the big data migration to the cloud and intelligent data lake solution, CDM provides easy-to-use migration capabilities and capabilities of integrating multiple data sources to the data lake, reducing the complexity of data source migration and integration and effectively improving the data migration and integration efficiency.

Figure 2-27 shows the solution for migrating MySQL data to the MRS cluster with CDM.

Figure 2-27 MySQL data migration



Prerequisites

- You have purchased an MRS cluster that contains the Hive service.
- You have obtained the IP address, port, database name, username, and password for connecting to the MySQL database. In addition, the user must have the read and write permissions on the MySQL database.
- You have uploaded the MySQL database driver by referring to **Managing Drivers**.

Creating a Hive Partitioned Table on MRS Hive

On MRS Hive, run the following SQL statement to create a Hive partitioned table named **trip_data** with three new fields **y**, **ym**, and **ymd** used as partition fields.

The SQL statement is as follows:

```
create table trip_data(TripID int,Duration int,StartDate timestamp,StartStation varchar(64),StartTerminal int,EndDate timestamp,EndStation varchar(64),EndTerminal int,Bike int,SubscriberType varchar(32),ZipCodev varchar(10))partitioned by (y int,ym int,ymd int);
```


NOTE

The **trip_data** table has three partition fields: year, year and month, and year, month, and date of the start time of a ride.

For example, if the start time of a ride is **2018/5/11 9:40**, the record is saved in the **trip_data/2018/201805/20180511** partition.

When the records in the **trip_data** table are summarized, only part of the data needs to be scanned, greatly improving the performance.

Creating a CDM Cluster and Binding an EIP to the Cluster

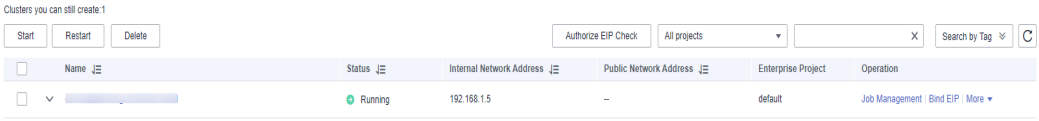
Step 1 If CDM is deployed as an independent service, create a CDM cluster by referring to **Creating a CDM Cluster**. If it is deployed as a component of the DataArts Studio service, create a CDM cluster by referring to **Creating a CDM Cluster**.

The key configurations are as follows:

- The flavor of the CDM cluster is selected based on the amount of data to be migrated. Generally, cdm.medium meets the requirements for most migration scenarios.
- The VPC, subnet, and security group of the CDM cluster must be the same as those of the MRS cluster.

Step 2 After the CDM cluster is created, on the **Cluster Management** page, click **Bind Elastic IP** in the **Operation** column to bind an EIP to the cluster. The CDM cluster uses the EIP to access MySQL.

Figure 2-28 Cluster list



NOTE

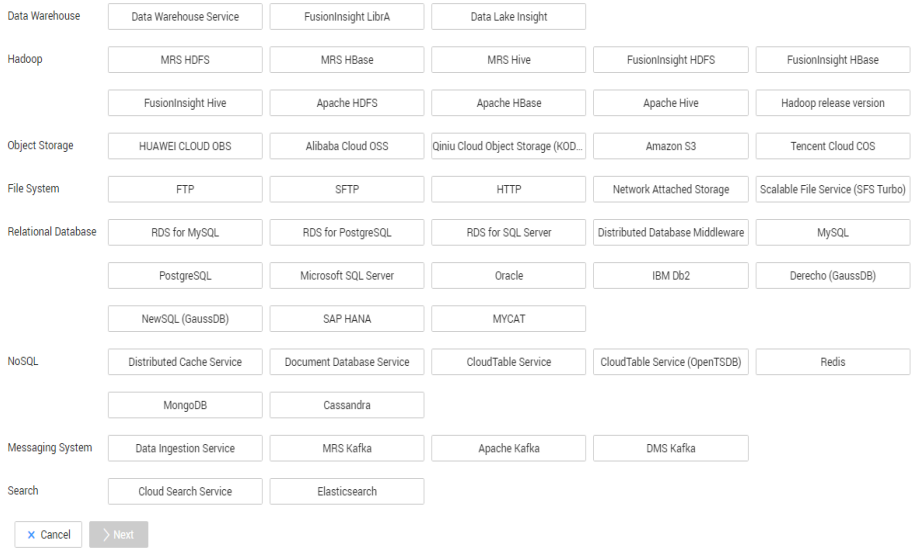
If SSL encryption is configured for the access channel of a local data source, CDM cannot connect to the data source using the EIP.

----End

Creating a MySQL Link

Step 1 On the **Cluster Management** page of CDM, click **Job Management** in the **Operation** column of the CDM cluster. On the displayed page, click the **Links** tab and then **Create Link**.

Figure 2-29 Selecting a connector



Step 2 Select **MySQL** and click **Next**. On the page that is displayed, configure MySQL link parameters.

You can click **Show Advanced Attributes** for more optional parameters. For details, see **Link to Relational Databases**. For this example, retain the default values of the optional parameters and configure the mandatory parameters according to **Table 2-9**.

Table 2-9 MySQL link parameters

Parameter	Description	Example Value
Name	Enter a unique link name.	mysqllink
Database Server	IP address or domain name of the MySQL database	192.168.1.110
Port	MySQL database port	3306
Database Name	Name of the MySQL database	sqoop
Username	User who has the read, write, and delete permissions on the MySQL database	admin
Password	Password of the user	-
Use Agent	Whether to extract data from the data source through an agent	Yes
Agent	Click **Select** and select the created agent.	-

Step 3 Click **Save**. The **Links** page is displayed.

NOTE

If an error occurs during the saving, the security settings of the MySQL database are incorrect. In this case, you need to enable the EIP of the CDM cluster to access the MySQL database.

----End

Creating a Hive Link

Step 1 On the **Links** page, click **Create Link** and select **MRS Hive** to create an MRS Hive link.

Step 2 Click **Next** and configure the MRS Hive link parameters. See **Figure 2-30**.

Figure 2-30 Creating an MRS Hive link

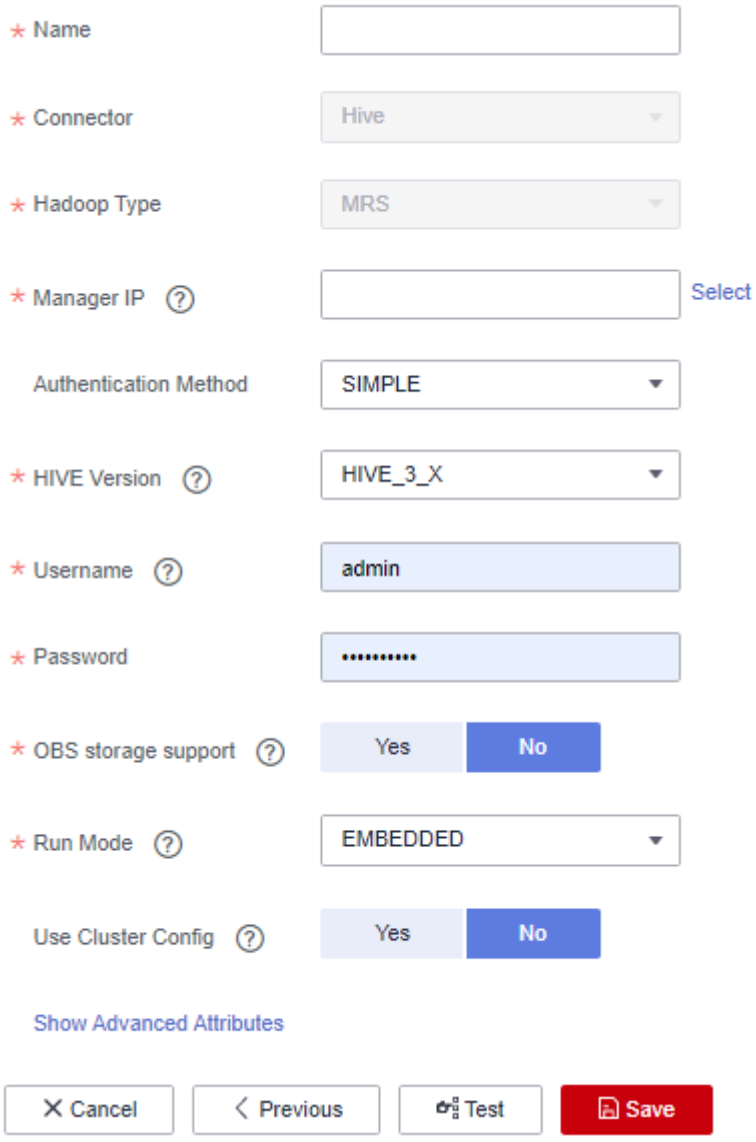


Table 2-10 describes the parameters. You can configure the parameters as required.

Table 2-10 MRS Hive link parameters

Parameter	Description	Example Value
Name	Link name, which should be defined based on the data source type, so it is easier to remember what the link is for	hivelink
Manager IP	Floating IP address of MRS Manager. Click **Select** next to the **Manager IP** text box to select an MRS cluster. CDM automatically fills in the authentication information.	127.0.0.1
Authentication Method	Authentication method used for accessing MRS • **SIMPLE**: Select this for non-security mode. • **KERBEROS**: Select this for security mode.	SIMPLE
HIVE Version	Hive version. Set it to the Hive version on the server.	HIVE_3_X
Username	If **Authentication Method** is set to **KERBEROS**, you must provide the username and password used for logging in to MRS Manager. If you need to create a snapshot when exporting a directory from HDFS, the user configured here must have the administrator permission on HDFS. To create a data link for an MRS security cluster, do not use user **admin**. The **admin** user is the default management page user and cannot be used as the authentication user of the security cluster. You can create an MRS user and set **Username** and **Password** to the username and password of the created MRS user when creating an MRS data link. **NOTE** • If the CDM cluster version is 2.9.0 or later and the MRS cluster version is 3.1.0 or later, the created user must have the permissions of the **Manager_viewer** role to create links on CDM. To perform operations on databases, tables, and data of a component, you also need to add the user group permissions of the component to the user. • If the CDM cluster version is earlier than 2.9.0 or the MRS cluster version is earlier than 3.1.0, the created user must have the permissions of **Manager_administrator** or **System_administrator** to create links on CDM. • A user with only the **Manager_tenant** or **Manager_auditor** permission cannot create links.	cdm

Parameter	Description	Example Value
Password	Password used for logging in to MRS Manager	-
OBS storage support	The server must support OBS storage. When creating a Hive table, you can store the table in OBS.	No
Run Mode	This parameter is used only when the Hive version is **HIVE_3_X**. Possible values are: • **EMBEDDED**: The link instance runs with CDM. This mode delivers better performance. • **Standalone**: The link instance runs in an independent process. If CDM needs to connect to multiple Hadoop data sources (MRS, Hadoop, or CloudTable) with both Kerberos and Simple authentication modes, select **STANDALONE** or configure different agents. Note: The **STANDALONE** mode is used to solve the version conflict problem. If the connector versions of the source and destination ends of the same data link are different, a JAR file conflict occurs. In this case, you need to place the source or destination end in the STANDALONE process to prevent the migration failure caused by the conflict.	EMBEDDED
Use Cluster Config	You can use the cluster configuration to simplify parameter settings for the Hive link.	No
Cluster Config Name	This parameter is valid only when **Use Cluster Config** is set to **Yes**. Select a cluster configuration that has been created. For details, see **Managing Cluster Configurations**.	hive_01

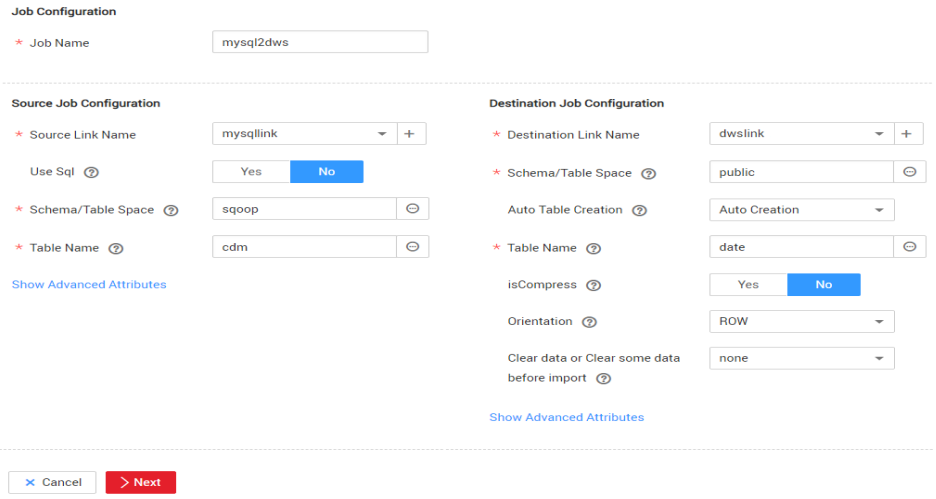
Step 3 Click **Save**. The **Links** page is displayed.

----End

Creating a Migration Job

Step 1 On the **Cluster Management** page, locate the row containing your desired cluster, and click **Job Management** in the **Operation** column. On the page that is displayed, click the **Table/File Migration** tab and then **Create Job** to create a data migration job. See **Figure 2-31**.

Figure 2-31 Creating a job for migrating data from MySQL to Hive



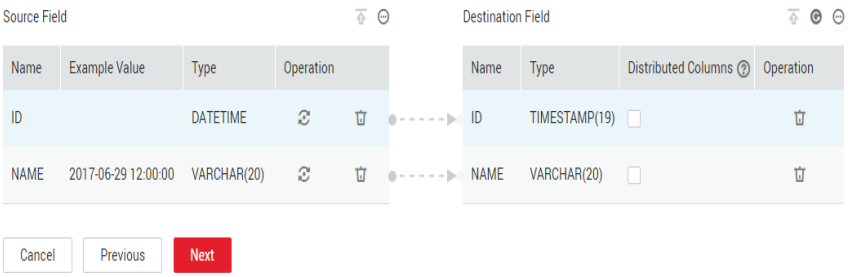
NOTE

Set **Clear Data Before Import** to **Yes** so that the data that has been imported to the Hive table is cleared each time before data is imported.

Step 2 After the parameters are configured, click **Next**. The **Map Field** page is displayed, as shown in **Figure 2-32**.

Map the fields of the MySQL table and Hive table. The Hive table has three more fields **y**, **ym**, and **ymd** than the MySQL table, which are the Hive partition fields. Because the fields of the source table cannot be directly mapped to the destination table, you need to configure an expression to extract data from the **StartDate** field in the source table.

Figure 2-32 Hive field mapping



Step 3 Click  to display the **Converter List** dialog box, and then choose **Create Converter** > **Expression conversion**.

The expressions for the **y**, **ym**, and **ymd** fields are as follows:

DateUtils.format(DateUtils.parseDate(row[2],"yyyy-MM-dd HH:mm:ss.SSS"),"yyyy")

DateUtils.format(DateUtils.parseDate(row[2],"yyyy-MM-dd HH:mm:ss.SSS"),"yyyyMM")

DateUtils.format(DateUtils.parseDate(row[2],"yyyy-MM-dd HH:mm:ss.SSS"),"yyyyMMdd")

NOTE

CDM expressions have built-in ability to convert fields of common strings, dates, and numbers. For details, see **Field Conversion**.

Step 4 Click **Next** to set task parameters. Generally, retain the default values of all parameters.

In this step, you can configure the following optional functions:

- **Retry Upon Failure**: If the job fails to be executed, you can determine whether to automatically retry. Retain the default value **Never**.
- **Group**: Select the group to which the job belongs. The default group is **DEFAULT**. On the **Job Management** page, jobs can be displayed, started, or exported by group.
- **Scheduled Execution**: For details about how to configure scheduled execution, see **Scheduling Job Execution**. Retain the default value **No**.
- **Concurrent Extractors**: Enter the number of extractors to be concurrently executed. Retain the default value **1**.
- **Write Dirty Data**: Specify this parameter if data that fails to be processed or filtered out during job execution needs to be written to OBS for future viewing. Before writing dirty data, create an OBS link. Retain the default value **No** so that dirty data is not recorded.
- **Delete Job After Completion**: Retain the default value **Do not delete**.

Step 5 Click **Save and Run**. The **Job Management** page is displayed, on which you can view the job execution progress and result.

Step 6 After the job is successfully executed, in the **Operation** column of the job, click **Historical Record** to view the job's historical execution records and read/write statistics.

On the **Historical Record** page, click **Log** to view the job logs.

----End

2.9 Migrating Data from MRS HDFS to OBS with CDM

Application Scenarios

With MRS, you can store data in OBS and dedicate MRS clusters solely to computing tasks, isolating storage and compute resources. This approach offers flexible, on-demand scaling at a lower cost, making it well-suited for big data processing.

The Cloud Data Migration (CDM) service supports file-to-file data migration. This topic describes how to migrate data from MRS HDFS to an OBS file system with CDM.

The process of migrating MRS HDFS data to OBS with CDM is as follows:

1. **Creating a CDM Cluster and Binding an EIP to the Cluster**
2. **Creating an MRS Link to HDFS**

3. **Creating an OBS Link**
4. **Creating a Migration Job**

Solution Architecture

The CDM simplifies data migration with easy-to-use tools and integrates multiple data sources into a single data lake, streamlining the process and boosting efficiency.

Figure 2-33 shows the solution for migrating HDFS data to an MRS cluster with CDM.

Figure 2-33 HDFS data migration

Prerequisites

- You have obtained the domain name, port number, AK, and SK for accessing OBS.
- You have created an MRS cluster that contains the Hadoop service.
- You have the EIP quota and have created an EIP.

Creating a CDM Cluster and Binding an EIP to the Cluster

Step 1 If CDM is deployed as an independent service, create a CDM cluster by referring to **Creating a CDM Cluster**. If it is deployed as a component of the DataArts Studio service, create a CDM cluster by referring to **Creating a CDM Cluster**.

The key configurations are as follows:

- The flavor of the CDM cluster is selected based on the amount of data to be migrated. Generally, **cdm.medium** meets the requirements for most migration scenarios.
- The VPC, subnet, and security group of the CDM cluster must be the same as those of the MRS cluster.

Step 2 After the CDM cluster is created, on the **Cluster Management** page, click **Bind Elastic IP** in the **Operation** column to bind an EIP to the cluster. The CDM cluster uses the EIP to access MRS HDFS.

NOTE

If SSL encryption is configured for the access channel of a local data source, CDM cannot connect to the data source using the EIP.

----End

Creating an MRS Link to HDFS

Step 1 On the **Cluster Management** page, click **Job Management** in the **Operation** column of the cluster. On the page that is displayed, click the **Links** tab then

Create Link. On the **Select Connector** page that is displayed, select **MRS HDFS** for **Hadoop**, and click **Next** to set MRS HDFS link parameters.

- **Name**: Enter a custom link name, for example, **mrs_hdfs_link**.
- **Manager IP**: IP address of MRS Manager. Click **Select** next to the **Manager IP** text box to select a created MRS cluster. CDM automatically fills in the authentication information.
- **Username**: If KERBEROS is used for authentication, the username and password for logging in to MRS Manager is required.

  If you need to create a snapshot when exporting a directory from HDFS, the user configured here must have the administrator permission on HDFS.
- **Password**: password for logging in to MRS Manager
- **Authentication Method**: authentication method for accessing MRS
- **Run Mode**: Select the running mode of the HDFS link.

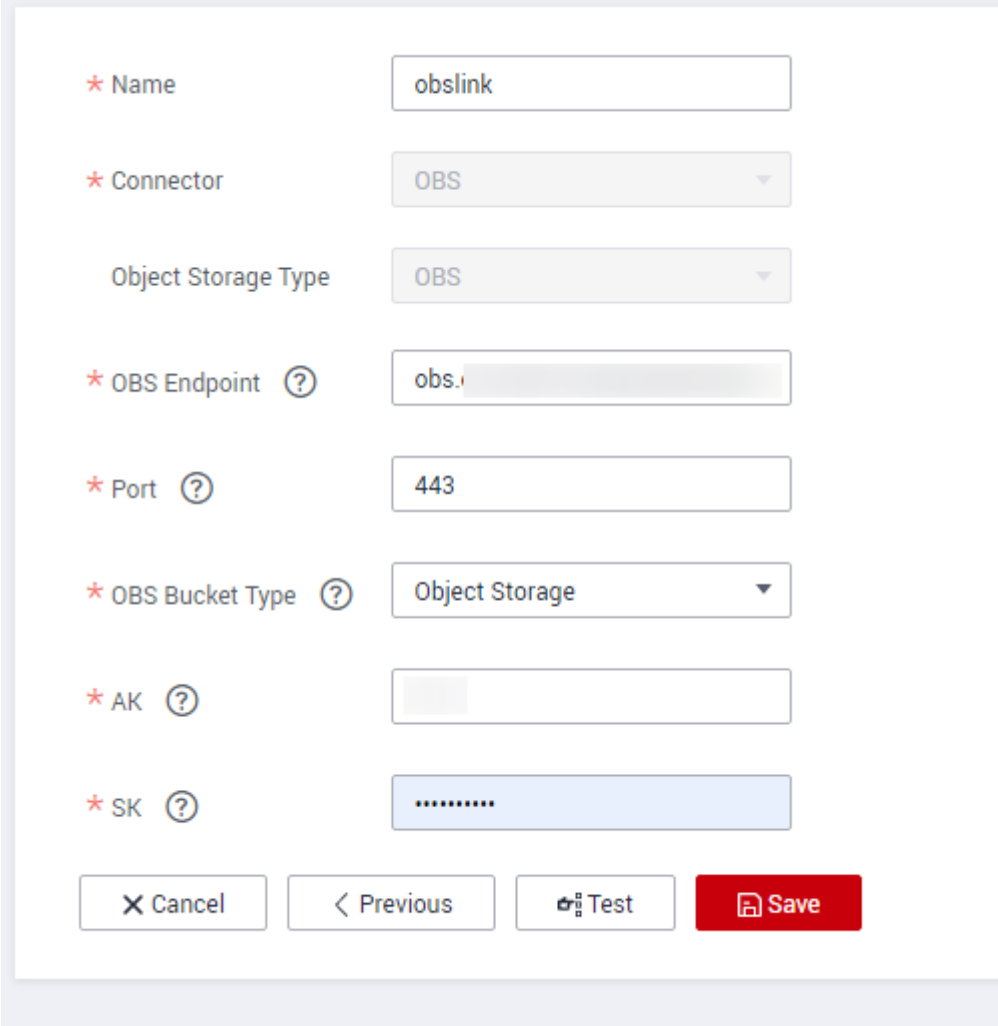
----End

Creating an OBS Link

Step 1 On the **Cluster Management** page, click **Job Management** in the **Operation** column of the cluster. On the page that is displayed, click the **Links** tab then **Create Link**. In the displayed dialog box, select **OBS** for **Connector**, and click **Next** to set OBS link parameters. See **Figure 2-34**.

- **Name**: Enter a custom link name, for example, **obslink**.
- **OBS Endpoint** and **Port**: Enter the actual OBS address information.
- **OBS Bucket Type**: Use the default option.
- **AK** and **SK**: Enter the AK and SK used for logging in to OBS.

Figure 2-34 Creating an OBS link



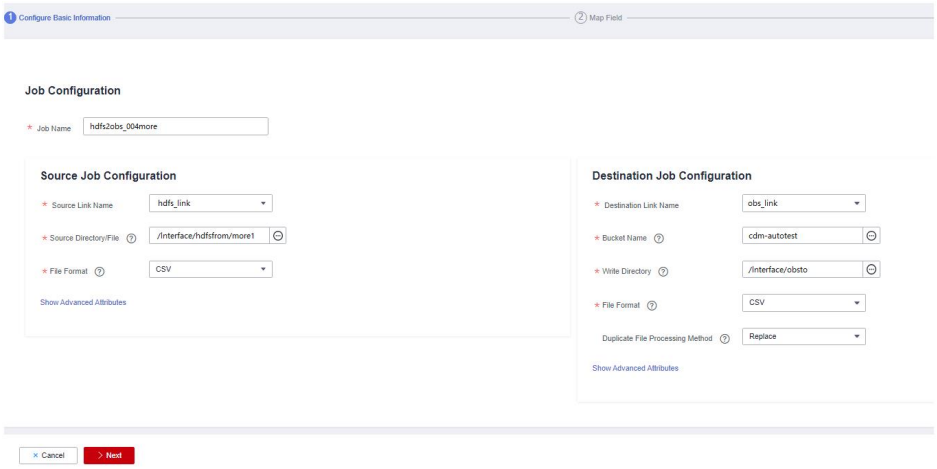
Step 2 Click **Save**. The **Links** page is displayed.

----End

Creating a Migration Job

Step 1 On the **Cluster Management** page, click **Job Management** in the **Operation** column of the cluster. On the page that is displayed, click the **Table/File Migration** tab then **Create Job** to create a job for exporting data from MRS HDFS to OBS.

Figure 2-35 Creating a job for migrating data from MRS HDFS to OBS



- **Job Name**: Enter a unique name.
- **Source Job Configuration**
  - **Source Link Name**: Select the **hdfs_link** created in **Creating an MRS Link to HDFS**.
  - **Source Directory/File**: Enter the directory or file path of the data to be migrated.
  - **File Format**: Select the file format used for data transmission. Select **Binary**. If files are transferred without being parsed, the file format does not have to be **Binary**. This applies to file copy.
  - Retain the default values of other optional parameters. For details, see **From HDFS**.
- **Destination Job Configuration**
  - **Destination Link Name**: Select the **obs_link** created in **Creating an OBS Link**.
  - **Bucket Name**: Select the bucket from which the data will be migrated.
  - **Write Directory**: Enter the directory to which data is to be written on the OBS server.
  - **File Format**: Select **Binary**.
  - Retain the default values of the optional parameters in **Show Advanced Attributes**. For details, see **To OBS**.

Step 2 Click **Next**. The **Map Field** page is displayed. CDM automatically matches the source and destination fields.

- If the field mapping is incorrect, you can drag the fields to adjust the mapping.
- CDM expressions have built-in ability to convert fields of common strings, dates, and numbers. For details, see **Field Conversion**.

Step 3 Click **Next** to set task parameters. Typically, retain the default values for all parameters.

In this step, you can configure the following optional functions:

- **Retry Upon Failure**: If the job fails to be executed, you can determine whether to automatically retry. Retain the default value **Never**.

- **Group**: Select the group to which the job belongs. The default group is **DEFAULT**. On the **Job Management** page, jobs can be displayed, started, or exported by group.
- **Scheduled Execution**: For details about how to configure scheduled execution, see **Scheduling Job Execution**. Retain the default value. The default value is **No**.
- **Concurrent Extractors**: Enter the number of extractors to be concurrently executed. CDM supports concurrent extraction of multiple files. Increasing the value of this parameter can improve migration efficiency.
- **Write Dirty Data**: Select **No**. The file-to-file migration is binary, and no dirty data will be generated.
- **Delete Job After Completion**: Retain the default value **Do not delete**. You can also set this parameter to **Delete** to prevent an accumulation of too many migration jobs.

Step 4 Click **Save and Run**. The **Job Management** page is displayed, on which you can view the job execution progress and result.

Step 5 After the job is successfully executed, in the **Operation** column of the job, click **Historical Record** to view the job's historical execution records and read/write statistics.

On the **Historical Record** page, click **Log** to view the job logs.

----End

3 Interconnection with Other Cloud Services

3.1 Using MRS Spark SQL to Access GaussDB(DWS)

You can use MRS to quickly build and operate a full-stack cloud-native big data platform on Huawei Cloud. Big data components such as HDFS, Hive, HBase, and Spark, are available on the platform for analyzing enterprise data at scale.

You can process structured data with the Spark SQL language that is similar to SQL. With Spark SQL, you can access different databases, extract data from these databases, process the data, and load it to different data stores.

This practice demonstrates how to use MRS Spark SQL to access GaussDB(DWS) data.

NOTE

This section applies only to MRS 3.x or later.

Prerequisites

- You have created an MRS cluster that contains the Spark component. For details, see **Buying an MRS cluster**.
- If Kerberos authentication is enabled for the cluster, log in to FusionInsight Manager, choose **System** > **Permission** > **User**, and add the human-machine user **sparkuser** to the user groups **hadoop** (primary) and **hive**. Add the **ADD JAR** permission by referring to **Adding a Ranger Access Permission Policy for Spark2x**. If Kerberos authentication is disabled for the MRS cluster, you do not need to add the user.
- The MRS cluster client has been installed. For details, see **Installing a Client**.
- You have created a GaussDB (DWS) cluster. For details, see **Creating a GaussDB (DWS) Cluster**. To ensure network connectivity, the AZ, VPC, and security group of the GaussDB (DWS) cluster must be the same as those of the MRS cluster.

- You have obtained the IP address, port number, database name, username, and password for connecting to the GaussDB(DWS) database. The user must have the read and write permissions on GaussDB(DWS) tables.

Procedure

Step 1 Prepare data and create databases and tables in the GaussDB(DWS) cluster.

1. Log in to the GaussDB(DWS) console and click **Log In** in the **Operation** column of the cluster.
2. Log in to the default database **gaussdb** of the cluster and run the following command to create the **dws_test** database:

   CREATE DATABASE dws_test;
3. Connect to the created database and run the following command to create the **dws_order** table:

   CREATE SCHEMA dws_data;

   CREATE TABLE dws_data.dws_order

   (order_id VARCHAR,

   order_channel VARCHAR,

   order_time VARCHAR,

   cust_code VARCHAR,

   pay_amount DOUBLE PRECISION,

   real_pay DOUBLE PRECISION);
4. Run the following command to insert data to the **dws_order** table:

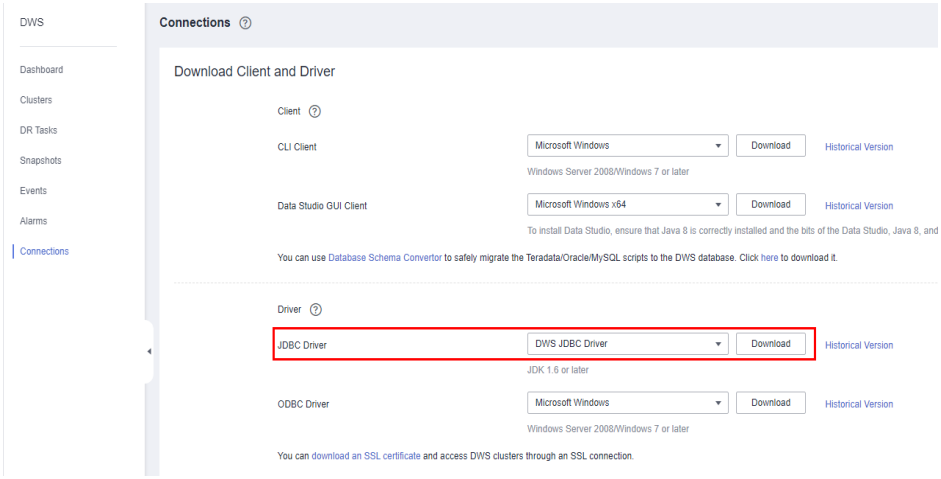
   INSERT INTO dws_data.dws_order VALUES ('202306270001', 'webShop', '2023-06-27 10:00:00', 'CUST1', 1000, 1000);

   INSERT INTO dws_data.dws_order VALUES ('202306270002', 'webShop', '2023-06-27 11:00:00', 'CUST2', 5000, 5000);
5. Run the following command to query the table data to check whether the data is inserted:

   SELECT * FROM dws_data.dws_order;

Step 2 Download the JDBC driver of the GaussDB(DWS) database and upload it to the MRS cluster.

1. Log in to the GaussDB (DWS) console, click **Connections** on the left, and download the JDBC driver.



2. Decompress the package to obtain the **gsjdbc200.jar** file and upload it to the active Master node of the MRS cluster, for example, to the **/tmp** directory.
3. Log in to the active Master node as user **root** and run the following commands:

   cd *Client installation directory*

   source bigdata_env

   kinit sparkuser (Change the password upon the first authentication. If Kerberos authentication is disabled, you do not need to run this command.)

   hdfs dfs -put /tmp/gsjdbc200.jar /tmp

```
[root@node-master2lsnt tmp]# hdfs dfs -put ./gsjdbc200.jar /tmp
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/Bigdata/client/HDFS/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/client/HDFS/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.30.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]
put: `/tmp/gsjdbc200.jar': File exists
[root@node-master2lsnt tmp]#
```

Step 3 Create a data source table in MRS Spark and access the GaussDB(DWS) table.

1. Log in to the Spark client node and run the following commands:

   cd *Client installation directory*

   source ./bigdata_env

   kinit sparkuser

   spark-sql --master yarn
2. Run the following command to add the driver Jar package:

   add jar hdfs://hacluster/tmp/gsjdbc200.jar;

```
spark-sql> add jar hdfs://hacluster/tmp/gsjdbc200.jar;
2023-06-28 01:36:39,554 | WARN  | main | The enable mv value "null" is invalid. Using the default value "false" | org.apache.carbondata.core.util.CarbonProperties.validateEnableMV(CarbonProperties.java:512)
2023-06-28 01:36:39,568 | WARN  | main | The value "LOCALLOCK" configured for key carbon.lock.type is invalid for current file system. Use the default value HDFSLOCK instead. | org.apache.carbondata.core.util.CarbonProperties.validateAndConfigureLockType(CarbonProperties.java:441)
ADD JAR hdfs://hacluster/tmp/gsjdbc200.jar
Added [/opt/Bigdata/client/Spark2x/tmp/b52347ce-d7c4-44d4-8868-cefac46b2d0e_resources/gsjdbc200.jar] to class path
Added resources: [hdfs://hacluster/tmp/gsjdbc200.jar]
ADD JAR hdfs://hacluster/tmp/gsjdbc200.jar
Added [/opt/Bigdata/client/Spark2x/tmp/b52347ce-d7c4-44d4-8868-cefac46b2d0e_resources/gsjdbc200.jar] to class path
Added resources: [hdfs://hacluster/tmp/gsjdbc200.jar]
Time taken: 1.967 seconds
```

3. Run the following commands to create a data source table in Spark and access GaussDB(DWS) data:

   CREATE TABLE IF NOT EXISTS spk_dws_order

   USING JDBC OPTIONS (

'url'='jdbc:gaussdb://192.168.0.228:8000/dws_test',

'driver'='com.huawei.gauss200.jdbc.Driver',

'dbtable'='dws_data.dws_order',

'user'='dbadmin',

'password'='*xxx*');

4. Run the following command to query the Spark table. Check whether the displayed data is the same as the GaussDB(DWS) data.

   SELECT * FROM spk_dws_order;

   ```
   spark-sql> SELECT * FROM spk_dws_order;
   202306270001    webShop 2023-06-27 10:00:00     CUST1   1000.0  1000.0
   202306270002    webShop 2023-06-27 11:00:00     CUST2   5000.0  5000.0
   Time taken: 3.416 seconds, Fetched 2 row(s)
   spark-sql>
   ```

   Verify that the returned data is the same as that shown in **Step 1**.

----End

3.2 Interconnecting Hive with CSS

Scenario

Use the Elasticsearch-Hadoop plug-in to exchange data between Hive and Elasticsearch of Cloud Search Service (CSS) so that Elasticsearch index data can be mapped to Hive tables.

NOTE

This section applies to MRS 3.*x* or later.

Prerequisites

The Hive service of MRS and the Elasticsearch service of CSS have been installed, and the two clusters can communicate with each other.

Procedure

Step 1 On the **Clusters** page of the CSS console, locate the row containing the target cluster and click **Access Kibana** in the **Operation** column. In the navigation pane of Kibana, click **Dev Tools**. On the console page that is displayed, run the following command to create the index **ddj_study_card_ratio_v12**:

```
PUT ddj_study_card_ratio_v12
{
  "mappings" : {
    "properties" : {
      "uniq_id" : {
        "type" : "text",
        "fields" : {
          "keyword" : {
            "type" : "keyword",
            "ignore_above" : 256
          }
        }
      }
    }
  }
```



```
  }
 }
```

If the following information is displayed, the index is created:

```
{
  "acknowledged" : true,
  "shards_acknowledged" : true,
  "index" : "ddj_study_card_ratio_v12"
}
```

Step 2 Run the following command to insert data into the **ddj_study_card_ratio_v12** index:

```
POST /ddj_study_card_ratio_v12/_doc/_bulk
{"index":{}}
{"id":"1", "uniq_id":"23323"}
```

If **errors** is **false** in the command output, the data is imported.

Step 3 Download the corresponding JAR file from **Past Releases** based on the Elasticsearch version in CSS.

For example, the JAR file corresponding to Elasticsearch 7.6.2 is **elasticsearch-hadoop-7.6.2.jar**.

NOTE

- The JAR file and Elasticsearch of CSS must have the same version. This section uses an Elasticsearch 7.6.2 cluster with security mode enabled as an example.
- If there are any additional custom modules, pack them into a separate JAR file.

Step 4 Upload the JAR file in **Step 3** to the **/opt/Bigdata/third_lib/Hive** directory on all HiveServer nodes and run the following command to modify the permission:

chown omm:wheel -R /opt/Bigdata/third_lib/Hive

Step 5 Log in to FusionInsight Manager and choose **Cluster** > **Services** > **Hive** . On the page that is displayed, click the **Instance** tab. On this tab page, select all HiveServer instances, and choose **More** > **Restart Instance**.

Step 6 Download **commons-httpclient-3.1.jar** from **Maven central warehouse** and upload this JAR file and the JAR file in **Step 3** to any node where the HDFS and Hive clients are installed in the cluster.

Step 7 Log in to the node to which the JAR files in **Step 6** are uploaded as the client installation user.

Step 8 Run the following command to authenticate the user:

cd *Client installation directory*

source bigdata_env

kinit *Component service user* (Skip this step for clusters with Kerberos authentication disabled.)

Step 9 Run the following command to create a directory for storing JAR files in HDFS:

hdfs dfs -mkdir *HDFS path for storing JAR files*

Step 10 Run the following command to upload the JAR files in **Step 6** to HDFS:

hdfs dfs -put *JAR file storage path HDFS path for storing JAR files*

Step 11 Run the following command to enable Hive to load a specified JAR file when executing a command line task:

beeline

add jar *HDFS path for storing JAR files*; (Execute this command once for each JAR file.)

Step 12 Run the following command to create an Elasticsearch external table:

```
CREATE EXTERNAL TABLE `ddj_study_card_ratio_v12_test`(
  `uniq_id` string)
ROW FORMAT SERDE
  'org.elasticsearch.hadoop.hive.EsSerDe'
STORED BY
  'org.elasticsearch.hadoop.hive.EsStorageHandler'
WITH SERDEPROPERTIES (
  'field.delim'='',
  'serialization.format'='')
TBLPROPERTIES (
  'bucketing_version'='2',
  'es.index.auto.create'='false',
  'es.mapping.date.rich'='false',
  'es.net.http.auth.pass'='Pzh6537projectx',
  'es.net.http.auth.user'='elastic',
  'es.nodes'='vpcep-e0b33065-75b7-4193-8395-dbd00d10bc39.cn-east-3.huaweicloud.com',
  'es.nodes.wan.only'='true',
  'es.port'='9200',
  'es.read.metadata'='true',
  'es.resource'='ddj_study_card_ratio_v12',
  'es.set.netty.runtime.available.processors'='false',
  'es.write.operation'='index',
  'last_modified_by'='root',
  'last_modified_time'='1655264909',
  'transient_lastDdlTime'='1655264909');
```

NOTE

Key parameters are described as follows:

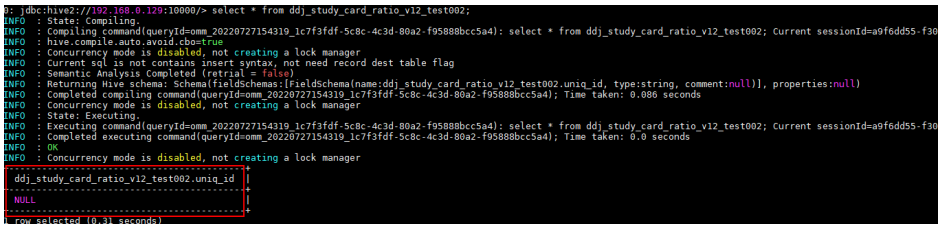
- **es.net.http.auth.pass** and **es.net.http.auth.user**: indicate the password and username of the user created in Kibana who has the permission to perform operations on indexes created in **Step 1**. For details, see **Creating a User and Granting Permissions by Using Kibana**.
- **es.nodes**: IP address to be connected. You can log in to the CSS management console and view the IP address of the cluster in the **Internal Access Addresses** column of the cluster list.
- **es.port**: port for external access to the Elasticsearch cluster. The default value is **9200**.
- **es.resource**: name of the index created in **Step 1**.

For details about parameter configurations, visit **https://www.elastic.co/guide/en/elasticsearch/hadoop/6.1/hive.html**.

Step 13 Run the following command to view the Elasticsearch external table created in **Step 12**:

select * from *ddj_study_card_ratio_v12_test*;

If no error information is displayed and the query is successful, Hive is interconnected with CSS. The command output is as follows:



----End

3.3 Connecting to the OBS File System with an MRS Hive Table

MRS allows you to store data in OBS and use an MRS cluster for data computing only. In this way, storage and compute are decoupled. You can use the IAM service to perform simple configurations to access OBS.

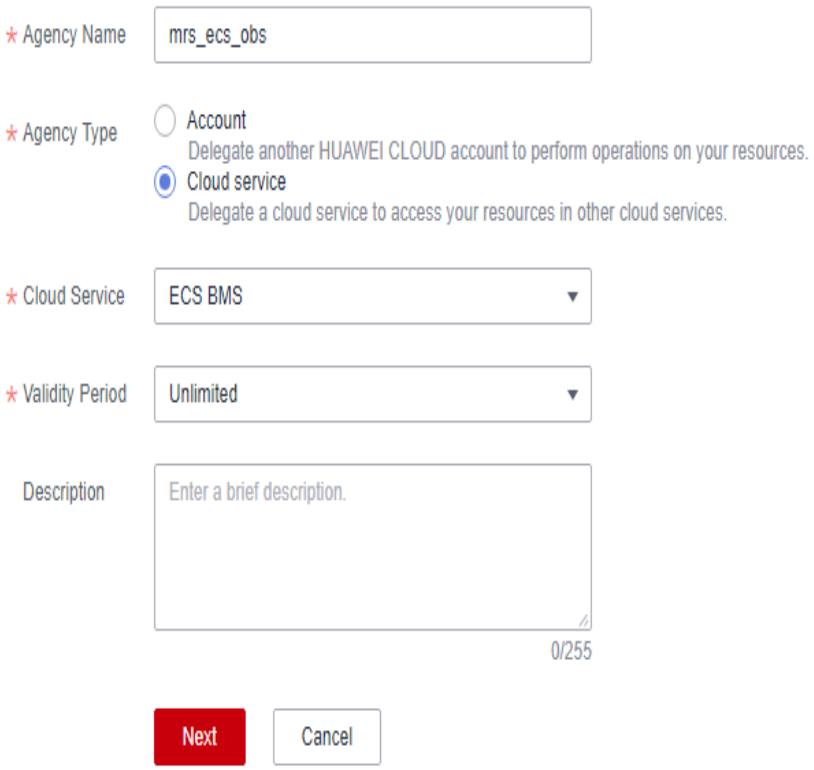
This section describes how to create a Hive table to store data to OBS.

1. **Creating an ECS Agency**
2. **Configuring an Agency for an MRS Cluster**
3. **Creating an OBS File System**
4. **Accessing the OBS File System Through Hive**

Creating an ECS Agency

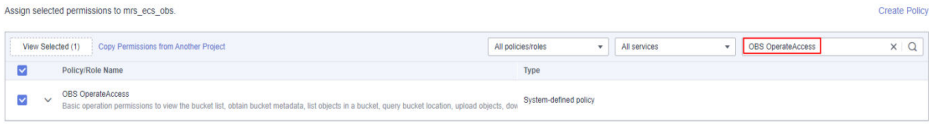
1. Log in to the Huawei Cloud management console.
2. Choose **Service List** > **Management & Governance** > **Identity and Access Management**.
3. Click **Agencies**. On the displayed page, click **Create Agency**.
4. Enter an agency name, for example, **mrs_ecs_obs**.
5. Set **Agency Type** to **Cloud service** and select **ECS BMS** to authorize ECS or BMS to invoke OBS.
6. Set **Validity Period** to **Unlimited** and click **Next**.

Figure 3-1 Creating an agency



7. On the page that is displayed, search for **OBS OperateAccess** in the search box and select it in the result list.

Figure 3-2 Assigning permissions



8. Click **Next**. On the page that is displayed, select the desired scope for the permissions you selected. By default, **All resources** is selected. Click **Show More**, select **Global resources**, and click **OK**.
9. In the dialog box that is displayed, click **OK** to start authorization. After the message "Authorization successful." is displayed, click **Finish**. The agency is created successfully.

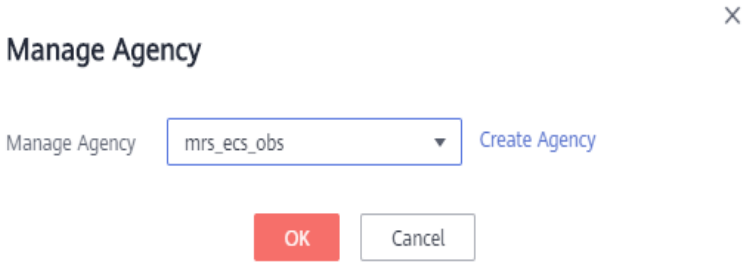
Configuring an Agency for an MRS Cluster

You can configure an agency when creating a cluster or bind an agency to an existing cluster to decouple storage and compute. This section uses an existing cluster as an example to describe how to configure an agency.

1. Log in to the MRS management console. In the left navigation pane, choose **Active Clusters**.
2. Click the name of a cluster to go to the cluster details page.
3. On the **Dashboard** page, click **Synchronize** on the right side of **IAM User Sync** to synchronize IAM users.

4. On the **Dashboard** page, click **Manage Agency** on the right side of **Agency** to select the agency created in **Creating an ECS Agency**, and click **OK** to bind it to the cluster. Alternatively, click **Create Agency** to go to the IAM console to create an agency and bind it to the cluster.

   Figure 3-3 Binding an agency

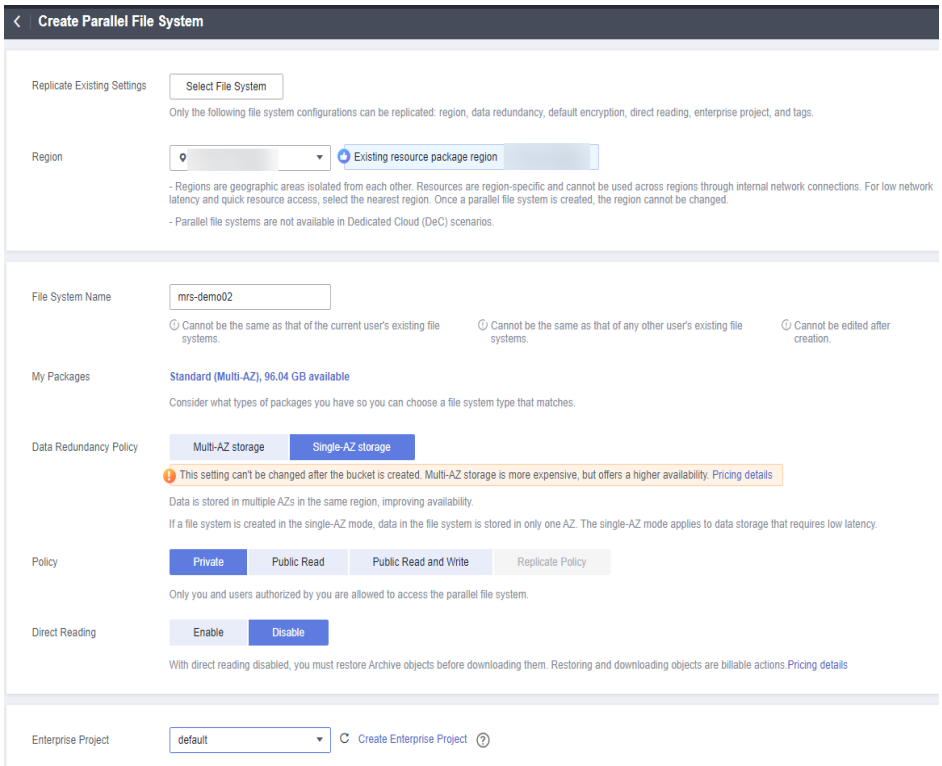


Creating an OBS File System

1. Log in to the OBS console.
2. Choose **Parallel File System** > **Create Parallel File System**.
3. Enter the file system name, for example, **mrs-demo01**.

   Set other parameters as required.

   Figure 3-4 Creating a parallel file system



4. Click **Create Now**.
5. In the parallel file system list on the OBS console, click a file system name to go to the details page.

6. In the navigation pane, choose **Files** and create **program** and **input** folders.
   - **program**: Upload the program package to this folder.
   - **input**: Upload the input data to this folder.

Accessing the OBS File System Through Hive

1. Log in to a master node as user **root**. For details, see **Logging In to an ECS**.
2. Verify that Hive can access OBS.
   a. Log in to the master node of the cluster as user **root** and run the following commands:

      cd /opt/Bigdata/client

      source bigdata_env

      source Hive/component_env

   b. View the list of files in file system **mrs-demo01**.

      hadoop fs -ls obs://mrs-demo01/

   c. Check whether the file list is returned. If it is returned, access to OBS is successful.

      Figure 3-5 Viewing the file list in mrs-demo01

      ```
      Found 2 items
      drwxrwxrwx   - hive hive          0 2021-10-22 10:08 obs://mrs-demo01/input
      drwxrwxrwx   - hive hive          0 2021-10-22 10:08 obs://mrs-demo01/program
      ```

   d. Run the following command to authenticate the user (skip this step for a normal cluster, that is, with Kerberos authentication disabled):

      kinit hive

      Enter the password of user **hive**. The default password is **Hive@123**. Change the password upon the first login.

   e. Run the Hive client command.

      beeline

   f. Access the OBS directory in the Beeline. For example, run the following command to create a Hive table and specify that data is stored in the **test_demo01** table of file system **mrs-demo01**:

      create table test_demo01(name string) location "obs://mrs-demo01/test_demo01";

   g. Run the following command to query all tables. If the **test_demo01** table is displayed in the command output, the access to OBS is successful.

      show tables;

      Figure 3-6 Checking whether the test_demo01 table exists

      ```
      +--------------+
      |   tab_name   |
      +--------------+
      | test_demo01  |
      +--------------+
      1 row selected (0.301 seconds)
      ```


h. Run the following command to check the table location.

show create table test_demo01;

Check whether the location of the table starts with **obs://***OBS bucket name***/**.

Figure 3-7 Checking the location of the test_demo01 table

```
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTe
LOCATION
  'obs://mrs-demo01/test_demo01'
TBLPROPERTIES (
  'bucketing_version'='2',
  'transient_lastDdlTime'='1634872329')
```

i. Run the following command to write data into the table.

insert into test_demo01 values('mm'),('ww'),('ww');

Run the **select * from test_demo01;** command to check whether the data is written successfully.

Figure 3-8 Viewing data in the test_demo01 table

```
+-------------------+
| test_demo01.name  |
+-------------------+
| mm                |
| ww                |
| ww                |
+-------------------+
```

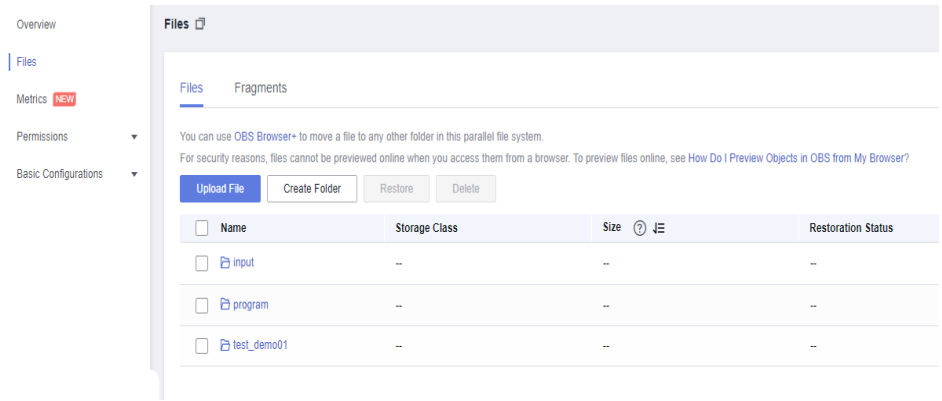
j. Run the **!q** command to exit the Beeline client.

k. Log in to the OBS console again.

l. Click **Parallel File System** and select the created file system.

m. Click **Files** to check whether the data exists in the created table.

Figure 3-9 Viewing data



4 Interconnection with Ecosystem Components

4.1 Using DBeaver to Access Phoenix

This topic uses DBeaver 6.3.5 as an example to describe how to access an MRS 3.1.0 cluster that requires Kerberos authentication. The HBase service in the cluster does not require Ranger authentication.

Prerequisites

- DBeaver 6.3.5 has been installed. You can download the DBeaver installation package by clicking **https://dbeaver.io/files/6.3.5/dbeaver-ce-6.3.5-x86_64-setup.exe**.
- An MRS 3.1.0 cluster, with HBase installed and Kerberos authentication disabled, has been created.
- The HBase client has been installed.
- JDK 1.8.0_*x* has been installed.

Procedure

Step 1 Add the bin directory of JDK 1.8.0_ *x*, for example, **C:\Program Files\Java\jdk1.8.0_121\bin**, to the **dbeaver.ini** file in the DBeaver installation directory.

Figure 4-1 Adding the bin directory of JDK

```
-vm
C:\Program Files\Java\jdk1.8.0_121\bin
```

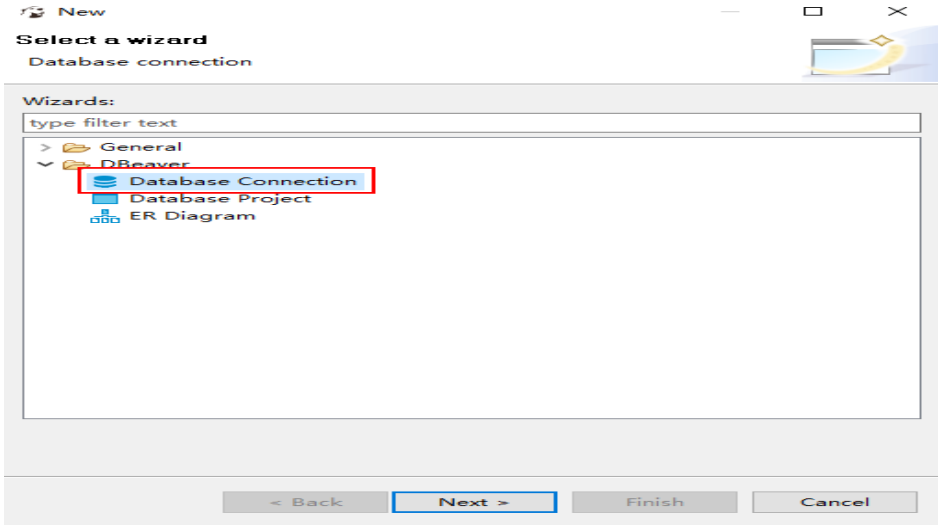
Step 2 Download the Phoenix software package from **https://archive.apache.org/dist/phoenix/apache-phoenix-5.0.0-HBase-2.0/bin/apache-phoenix-5.0.0-HBase-2.0-bin.tar.gz** and decompress it to obtain **phoenix-5.0.0-HBase-2.0-client.jar**.

Step 3 Download the **hbase-site.xml** file from *Client installation directory*/**HBase/hbase/conf** on the node where the client is installed. Use the compression software to

open the **phoenix-5.0.0-HBase-2.0-client.jar** file obtained in **Step 2** and drag **hbase-site.xml** to the JAR file.

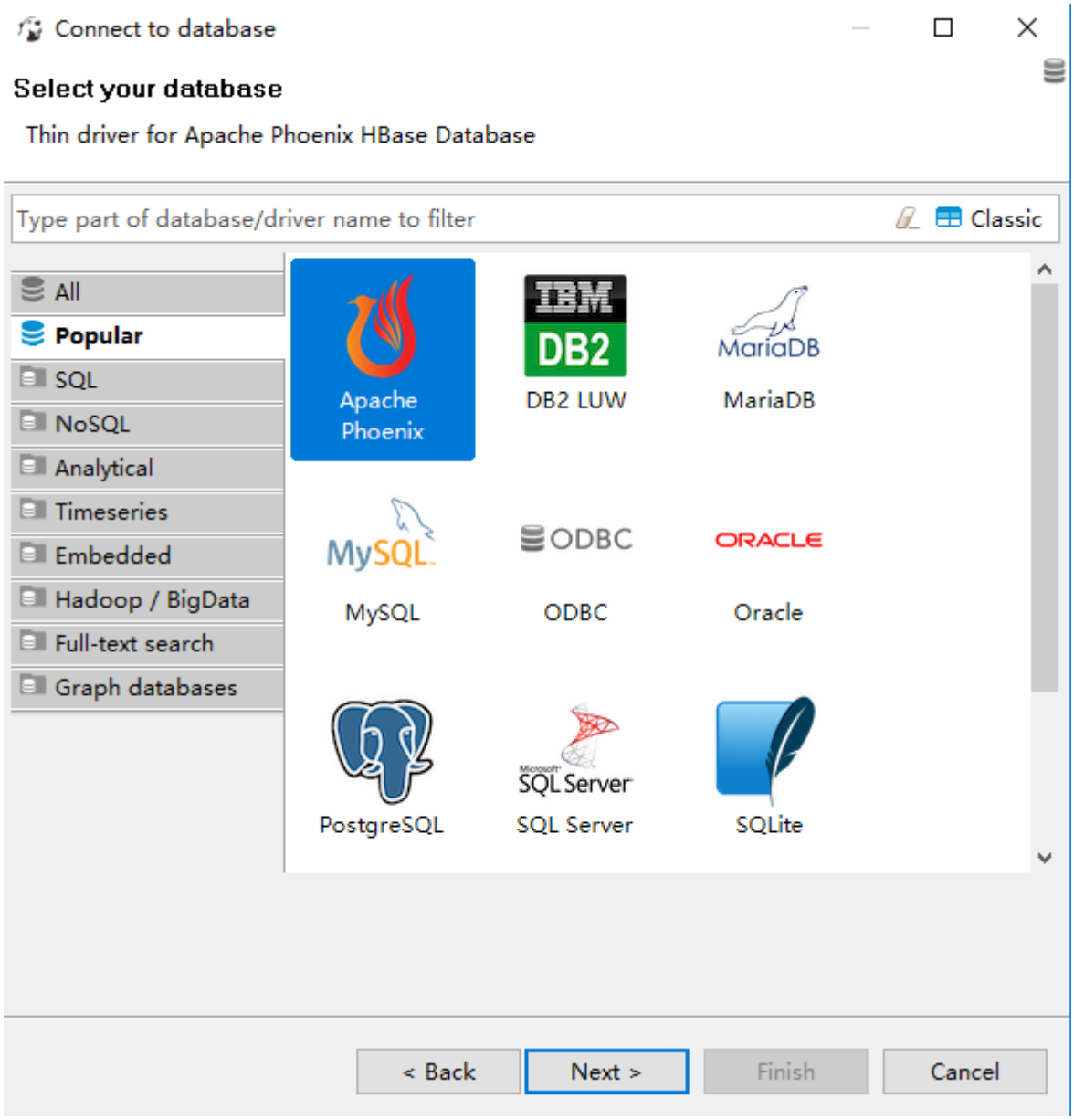
Step 4 Open DBeaver. On the navigation pane, choose **File** > **New** > **DBeaver** > **Database Connection**.

Figure 4-2 Creating a database connection



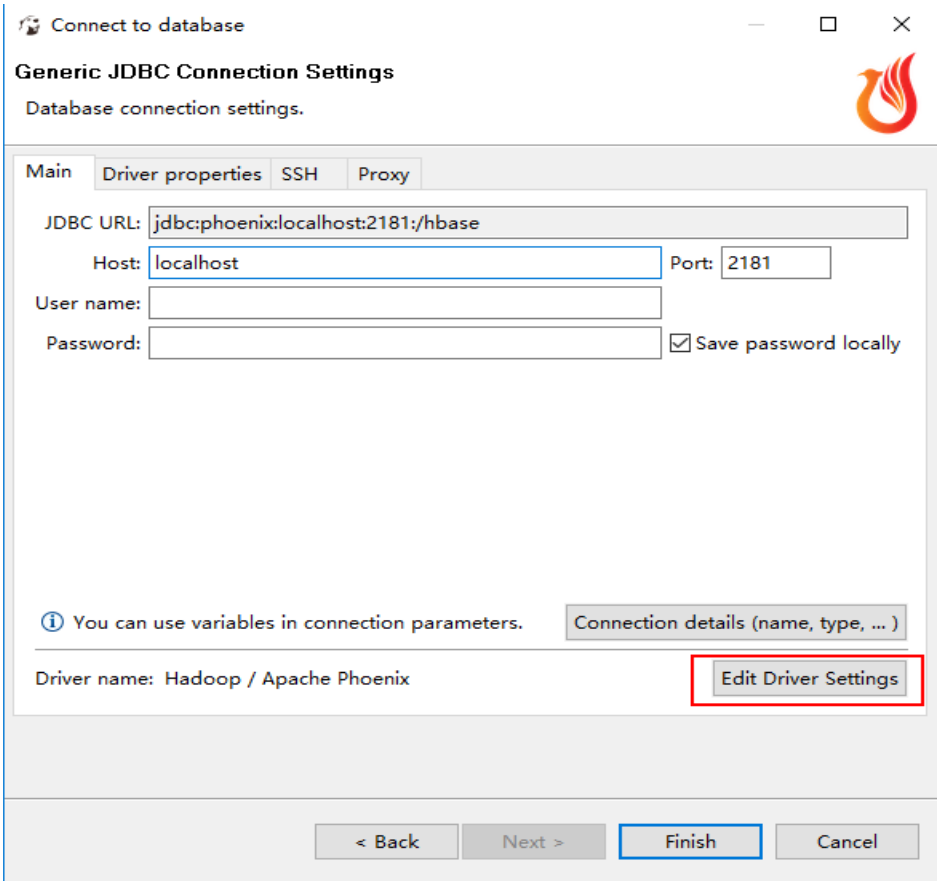
Step 5 Click **Next**. In the **Select your database** dialog box, select **Apache Phoenix** and click **Next**.

Figure 4-3 Selecting a database



Step 6 Click **Edit Driver Settings**.

Figure 4-4 Edit Driver Settings



Step 7 Click **Add File** and select the prepared **phoenix-5.0.0-HBase-2.0-client.jar** file. If there are multiple driver packages, delete them and retain only **added phoenix-5.0.0-HBase-2.0-client.jar**.

Figure 4-5 Deleting original driver packages

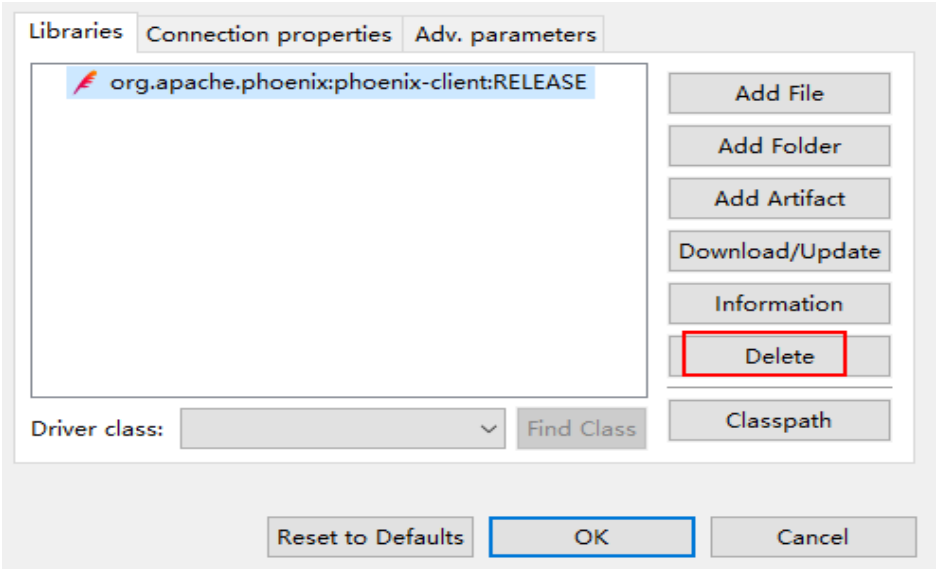
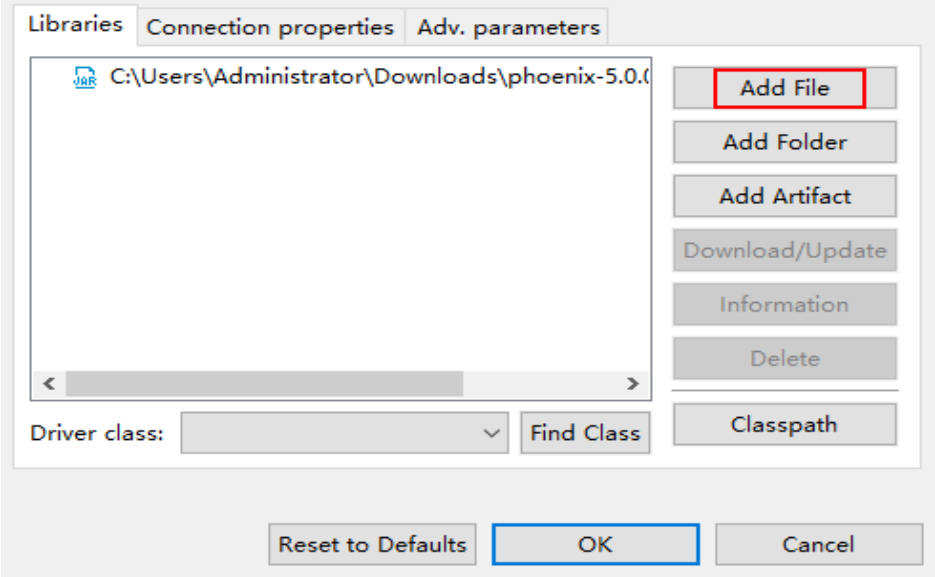
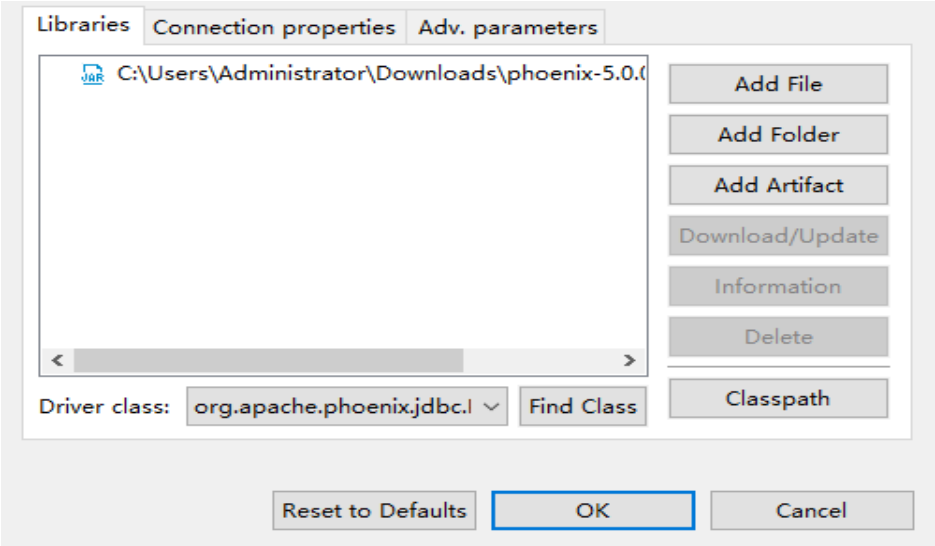


Figure 4-6 Adding the Phoenix JAR file



Step 8 Click **Find Class** and select **org.apache.phoenix.jdbc.PhoenixDriver** for **Driver class**.

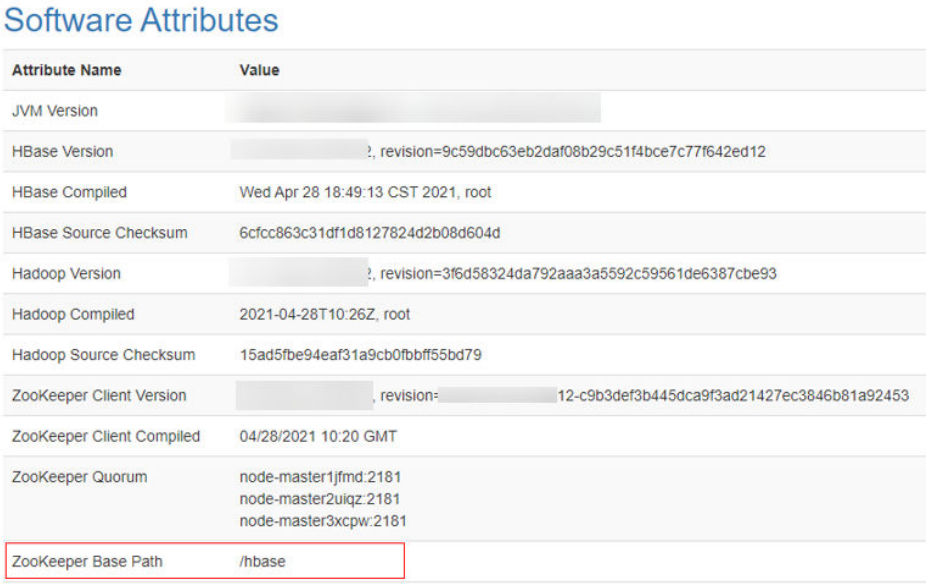
Figure 4-7 Loading a driver class



Step 9 Add the **ZooKeeper Base Path**.

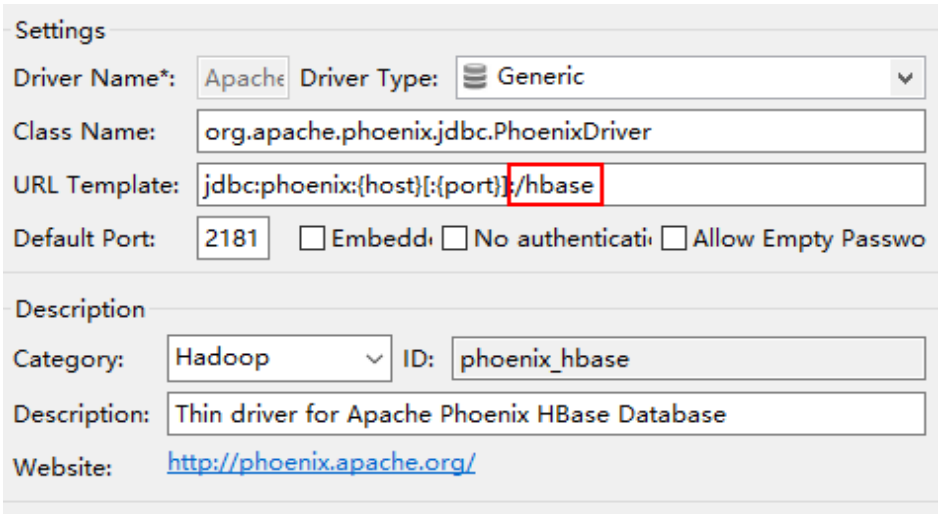
1. Log in to FusionInsight Manager and choose **Cluster** > **Services** > **HBase**. On the **Dashboard** tab page that is displayed, click the link next to **HMaster WebUI** to access the HBase web UI. Search for **ZooKeeper Base Path** and obtain its value. As shown in the following figure, the value of **ZooKeeper Base Path** is **/hbase**.

Figure 4-8 Viewing the value of ZooKeeper Base Path



2. Add a colon (:) and the **ZooKeeper Base Path** value, that is, **:/hbase** to the end of the original URL for **URL Template** and click **OK**.

Figure 4-9 Configuring URL Template



Step 10 Configure EIPs. If the network between local Windows hosts and the cluster is disconnected, configure an EIP for each HBase node and ZooKeeper node, and add the mapping between the EIPs of all nodes and the host domain names to the **hosts** file on the local Windows hosts. An example is as follows:

```
100.          node-master3xCPw node-master3xCPw.
100.         node-group-1ZqBd0001 node-group-1ZqBd0001.
100.          node-master2uIQz node-master2uIQz.
100.         0 node-group-1ZqBd0002 node-group-1ZqBd0002.
```

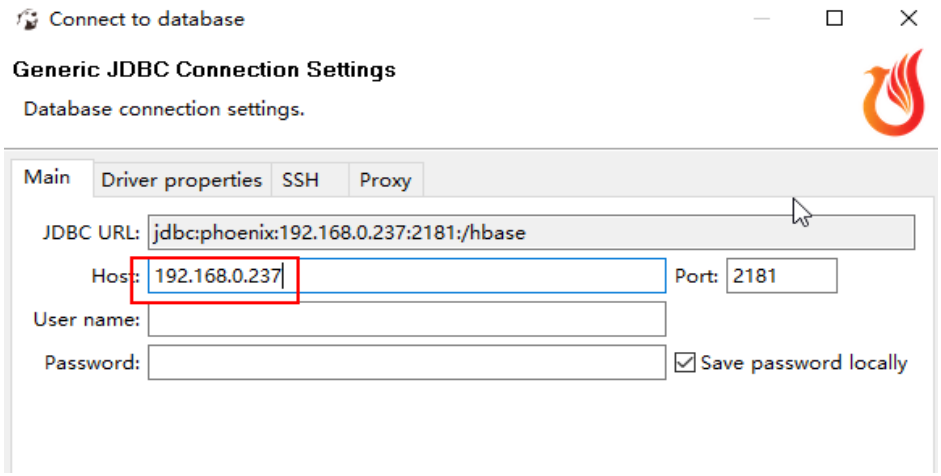
NOTE

If Windows ECSs are used and they can communicate with the cluster, you do not need to configure EIPs.

Step 11 Log in to FusionInsight Manager, choose **Cluster** > **Services** > **ZooKeeper**, and click the **Instance** tab.

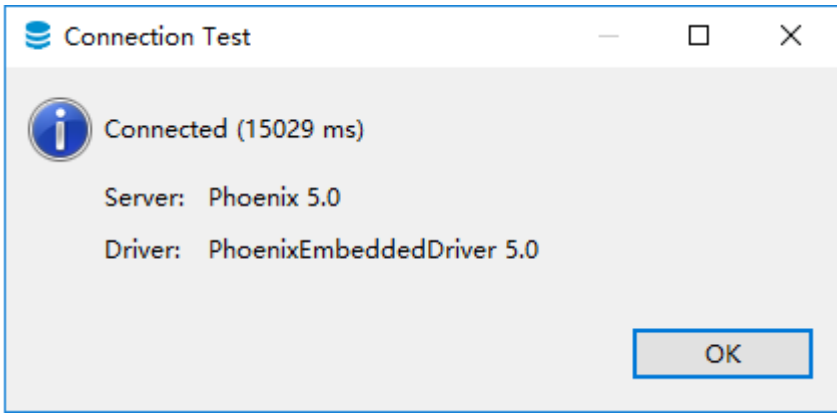
Select a node and enter the EIP of the node in **Host**. (If a Windows ECS is selected and it can communicate with the cluster properly, enter the service IP address of the ECS.)

Figure 4-10 Configuring Host



Step 12 Click **Test Connection**. If the information shown in **Figure 4-11** is displayed, the interconnection is successful. Click **OK**.

Figure 4-11 Connection Test dialog box



Step 13 Log in to the node where the HBase client is installed as the client installation user and run the following commands to create the *MY_NS* namespace:

cd *Client installation directory*

source bigdata_env

hbase shell

create_namespace "*MY_NS*"

Step 14 Open DBeaver and choose **SQL Editor** > **New SQL Editor** to run related SQL statements.

1. Enter the following commands in the editor and choose **SQL Editor** > **Execute SQL Statement** to create the *TEST* table in the **DEFAULT** namespace:

CREATE TABLE IF NOT EXISTS *TEST* **(id VARCHAR PRIMARY KEY, name VARCHAR);**

UPSERT INTO TEST(id,name) VALUES ('1','jamee');

2. Enter the following commands in the editor and choose **SQL Editor** > **Execute** to create the *TEST* table in the *MY_NS* namespace and inset data to the namespace:

   CREATE TABLE IF NOT EXISTS *MY_NS.TEST* **(id integer not null primary key, name varchar);**

   UPSERT INTO *MY_NS.TEST* **VALUES(1,'John');**

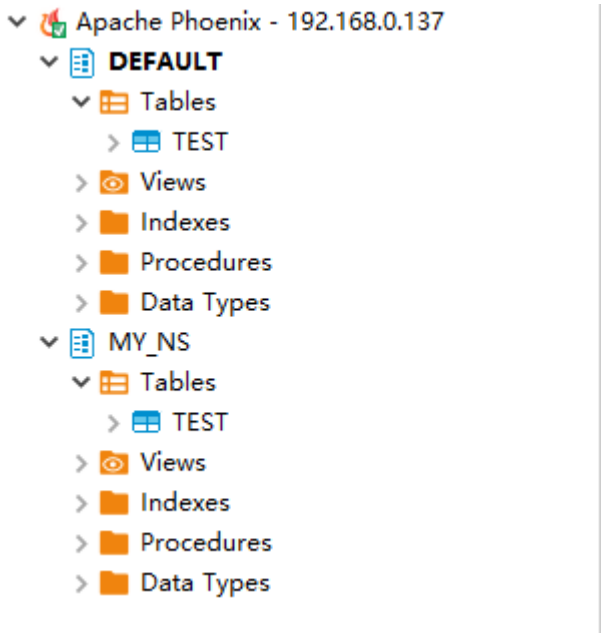
   UPSERT INTO *MY_NS.TEST* **VALUES(2,'Tom');**

   UPSERT INTO *MY_NS.TEST* **VALUES(3,'Manson');**

   UPSERT INTO *MY_NS.TEST* **VALUES(4,'Aurora');**

Step 15 Right-click the connection name, click **Refresh**, and click ˅ on the left of the connection name to view the tables created in **DEFAULT** and *MY_NS*.

Figure 4-12 Viewing tables



----End

4.2 Using DBeaver to Access HetuEngine

Use DBeaver 7.2.0 as an example to describe how to access HetuEngine.

Prerequisites

- The DBeaver has been installed properly. Download the DBeaver software from **https://dbeaver.io/files/7.2.0/**.
- You have created a human-machine user, for example, **hetu_user**, in the cluster. For details, see **Creating a HetuEngine User**. For clusters with Ranger authentication enabled, you need to grant the Ranger permission to

hetu_user based on service requirements. For details, see **Adding a Ranger Access Permission Policy for HetuEngine**.

- A compute instance has been created and is running properly. For details, see **Creating a HetuEngine Compute Instance**.

Procedure

Step 1 Download the HetuEngine client to obtain the JDBC JAR package.

1. Log in to FusionInsight Manager.
2. Choose **Cluster** > **Services** > **HetuEngine** > **Dashboard**.
3. In the upper right corner of the page, choose **More** > **Download Client** and download the **Complete Client** to the local PC as prompted.
4. Decompress the HetuEngine client package **FusionInsight_Cluster**_*Cluster ID*_**HetuEngine_Client.tar** to obtain the JDBC file and save it to a local directory, for example, **D:\test**.

   NOTE

   How to obtain the JDBC file:

   Decompress the package in the **FusionInsight_Cluster**_*Cluster ID*_**HetuEngine_ClientConfig\HetuEngine\xxx** directory to obtain the **hetu-jdbc-*.jar** file.

   Note: *xxx* can be **arm** or **x86**.

Step 2 Add the host mapping to the local **hosts** file.

Add the mapping of the host where the instance is located in the HSFabric or HSBroker mode. The format is *Host IP address Host name*.

Example: **192.168.42.90 server-2110081635-0001**

NOTE

The local **hosts** file in a Windows environment is stored in, for example, **C:\Windows\System32\drivers\etc**.

Step 3 Open DBeaver, choose **Database** > **New Database Connection**, search for **PrestoSQL** in **ALL**, and open PrestoSQL.

Step 4 Click **Edit Driver Settings** and set parameters by referring to the following table.

Table 4-1 Driver settings

Parameter	Value Description
Class Name	io.prestosql.jdbc.PrestoDriver

Parameter	Value Description
URL Template	• Access HetuEngine with HSFabric jdbc:presto://<*HSFabricIP1:port1*>,<*HSFabricIP2:port2*>,<*HSFabricIP3:port3*>/hive/default?serviceDiscoveryMode=hsfabric The following is an example. jdbc:presto://192.168.42.90:29902,192.168.42.91:29902,192.168.42.92:29902/hive/default?serviceDiscoveryMode=hsfabric • Access HetuEngine with HSBroker jdbc:presto://<*HSBrokerIP1:port1*>,<*HSBrokerIP2:port2*>,<*HSBrokerIP3:port3*>/hive/default?serviceDiscoveryMode=hsbroker The following is an example. jdbc:presto://192.168.42.90:29860,192.168.42.91:29860,192.168.42.92:29860/hive/default?serviceDiscoveryMode=hsbroker

NOTE

- To obtain the IP addresses and port numbers of the HSFabric and HSBroker nodes, perform the following operations:
  1. Log in to FusionInsight Manager.
  2. Choose **Cluster** > **Services** > **HetuEngine**. Click the **Instance** tab to obtain the service IP addresses of all HSFabric or HSBroker instances. You can select one or more normal instances for connection.
  3. To obtain the port numbers, choose **Cluster** > **Services** > **HetuEngine**. Click **Configurations** then **All Configurations**.

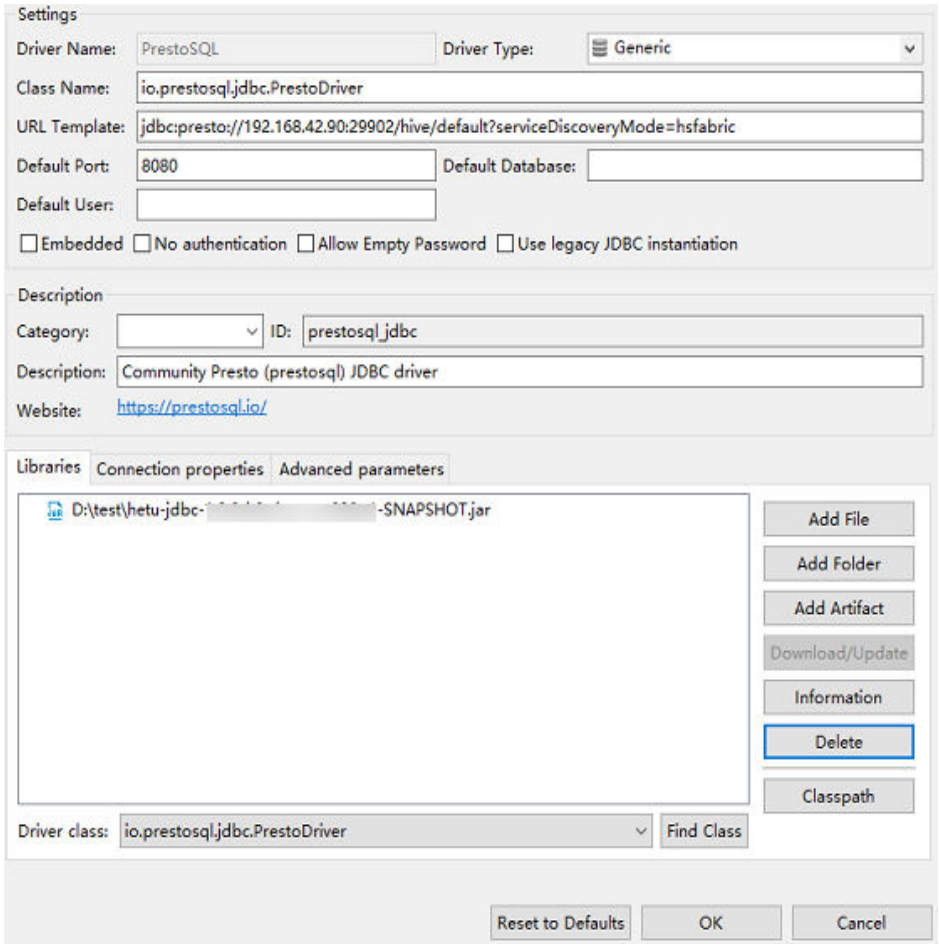
     Search for **gateway.port** to obtain the HSFabric port number. The default port number is **29902** in security mode and **29903** in normal mode.

     Search for **server.port** to obtain the HSBroker port number. The default port number is **29860** in security mode and **29861** in normal mode.
- If the connection fails, disable the proxy and try again.

Step 5 Click **Add File** and upload the JDBC driver package obtained in **Step 1**.

Step 6 Click **Find Class**. The driver class is automatically obtained. Click **OK** to complete the driver setting. If **io.prestosql:presto-jdbc:RELEASE** exists in **Libraries**, delete it before clicking **Find Class**.

Figure 4-13 Configuring the driver in security mode

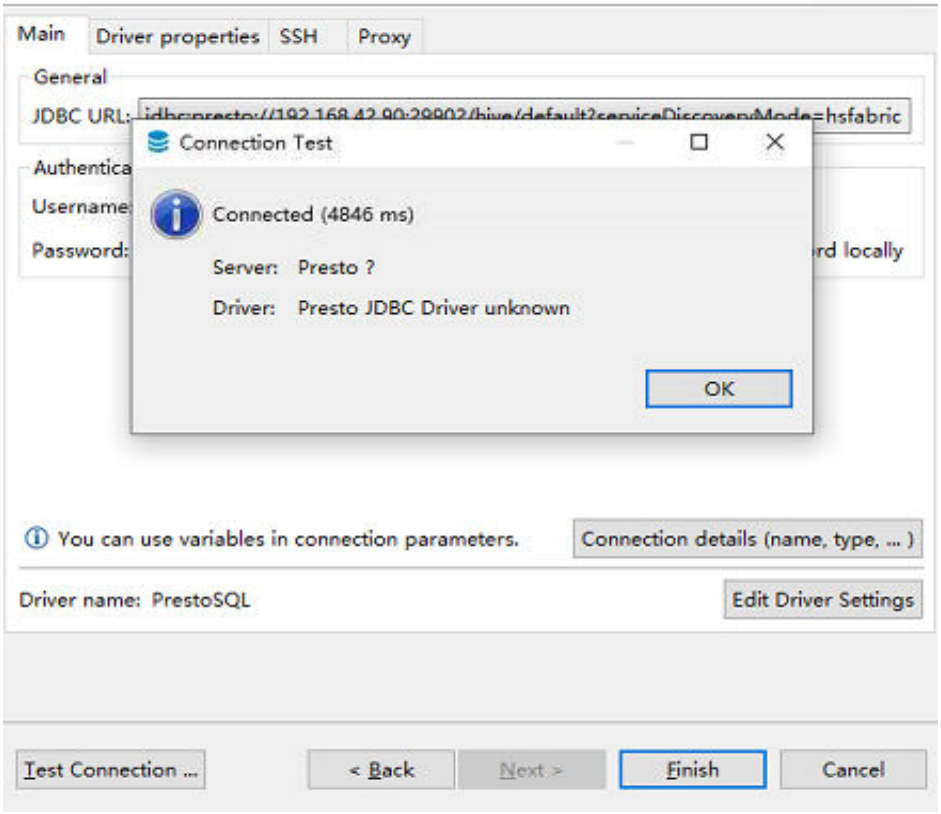


Step 7 Configure the connection.

- Security mode (clusters with Kerberos authentication enabled):

  On the **Main** tab page for creating a connection, enter the username and password created in **Prerequisites**, and click **Test Connection**. After the connection is successful, click **OK** then **Finish**. You can click **Connection details (name, type, ...)** to change the connection name.

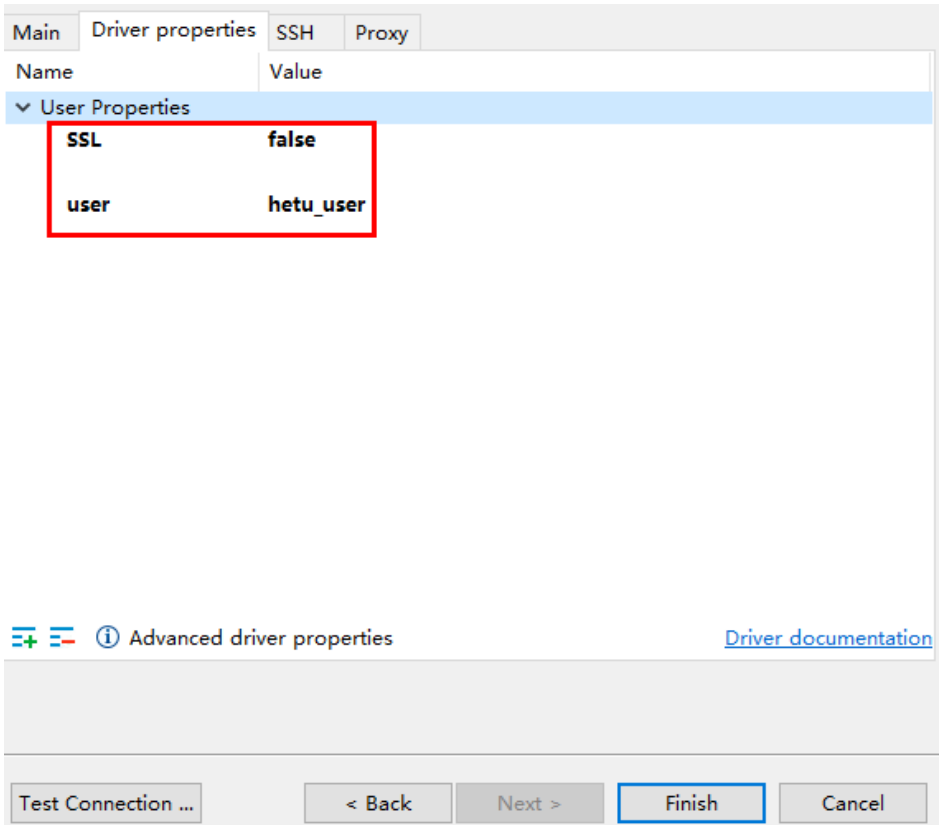
Figure 4-14 Configuring parameters on the Main tab in security mode



- Normal mode (clusters with Kerberos authentication disabled):

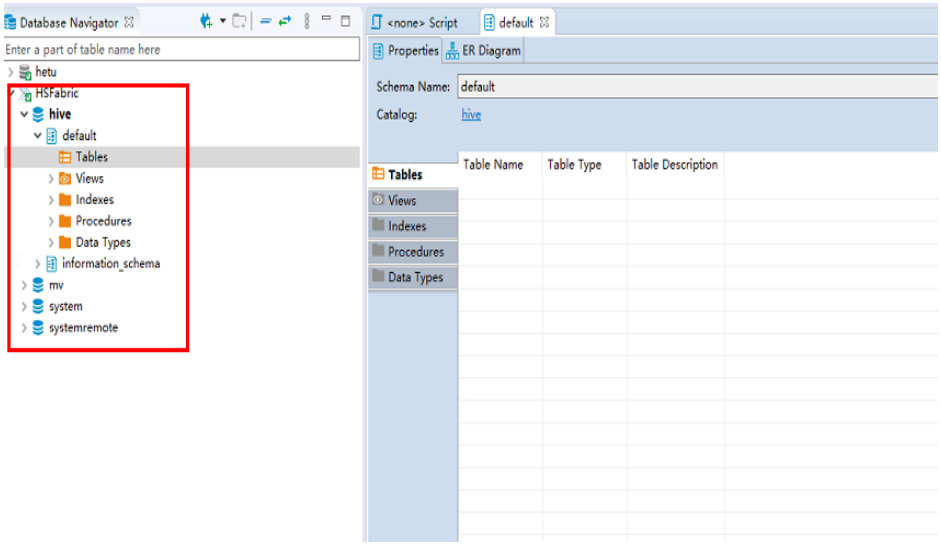
  On the page for creating a connection, configure the parameters on the **Driver properties** tab. Set **user** to the user created in **Prerequisites**. Click **Test Connection**. After the connection is successful, click **OK** then **Finish**. You can click **Connection details (name, type, ...)** to change the connection name.

Figure 4-15 Configuring parameters on the Driver properties tab in normal mode



Step 8 After the connection is successful, the page shown in **Figure 4-16** is displayed.

Figure 4-16 Successful connection



----End

4.3 Using Tableau to Access HetuEngine

Use Tableau Desktop 2022.2 as an example to describe how to access HetuEngine in a security cluster.

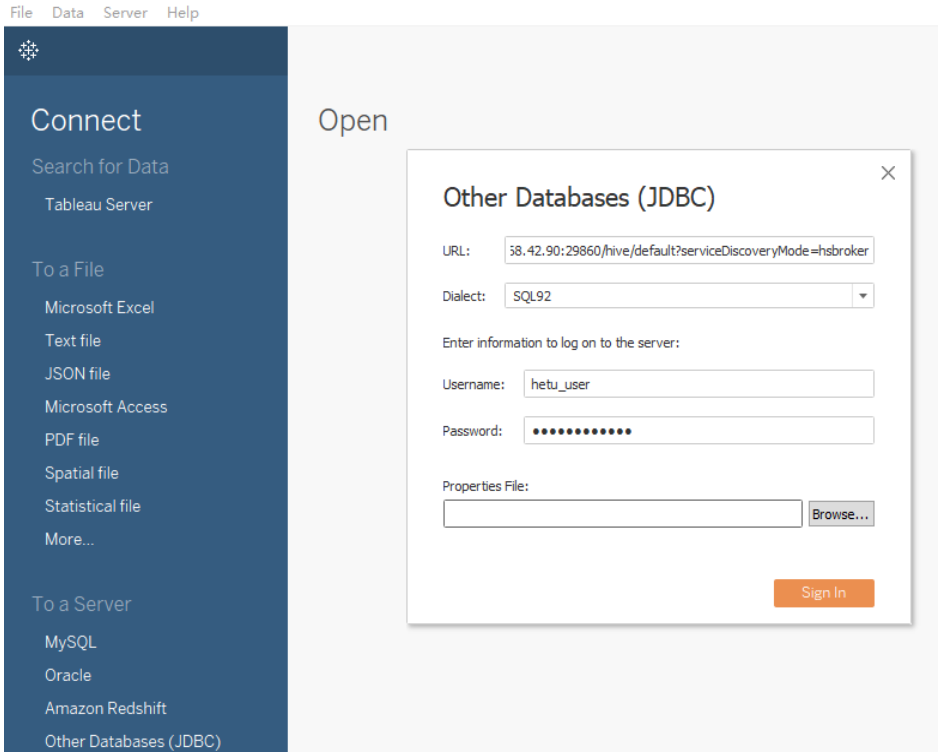
Prerequisites

- Tableau Desktop has been installed.
- The JDBC JAR file has been obtained. For details, see **Step 1**.
- You have created a human-machine user, for example, **hetu_user**, in the cluster. For details, see **Creating a HetuEngine User**. For clusters with Ranger authentication enabled, you need to grant the Ranger permission to **hetu_user** based on service requirements. For details, see **Adding a Ranger Access Permission Policy for HetuEngine**.
- A compute instance has been created and is running properly. For details, see **Creating a HetuEngine Compute Instance**.

Procedure

Step 1 Place the obtained JAR file to the Tableau installation directory, for example, **C:\Program Files\Tableau\Drivers**.

Step 2 Open Tableau, choose **To a Server** > **Other Databases (JDBC)**, enter the URL and the username and password of the created human-machine user, and click **Sign In**. HetuEngine is accessible either with HSFabric or HSBroker. For details about the URL format, see **Table 4-1**.



Step 3 After the login is successful, drag the desired data table to the operation window on the right and refresh data.

----End

4.4 Using Yonghong BI to Access HetuEngine

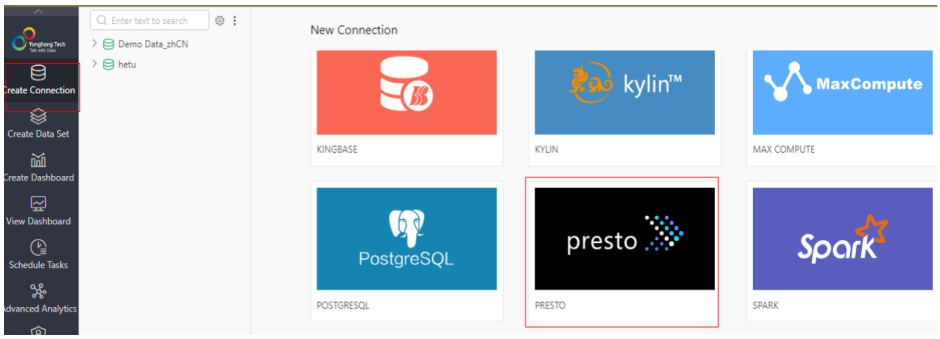
Use Yonghong Desktop 9.1 as an example to describe how to access HetuEngine in a security cluster.

Prerequisites

- Yonghong Desktop has been installed.
- The JDBC JAR file has been obtained. For details, see **Step 1**.
- You have created a human-machine user, for example, **hetu_user**, in the cluster. For details, see **Creating a HetuEngine User**. For clusters with Ranger authentication enabled, you need to grant the Ranger permission to **hetu_user** based on service requirements. For details, see **Adding a Ranger Access Permission Policy for HetuEngine**.
- A compute instance has been created and is running properly. For details, see **Creating a HetuEngine Compute Instance**.

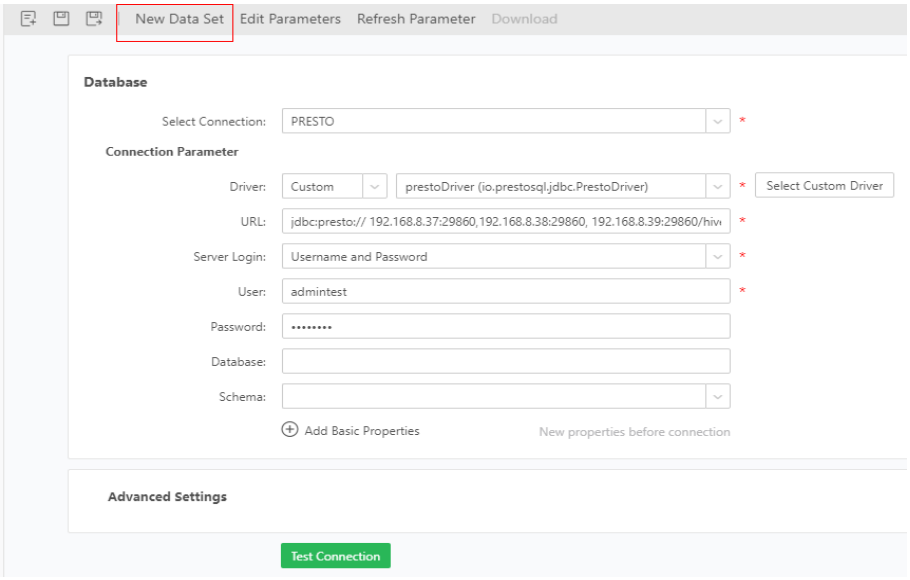
Procedure

Step 1 Open Yonghong Desktop and choose **Create Connection** > **presto**.

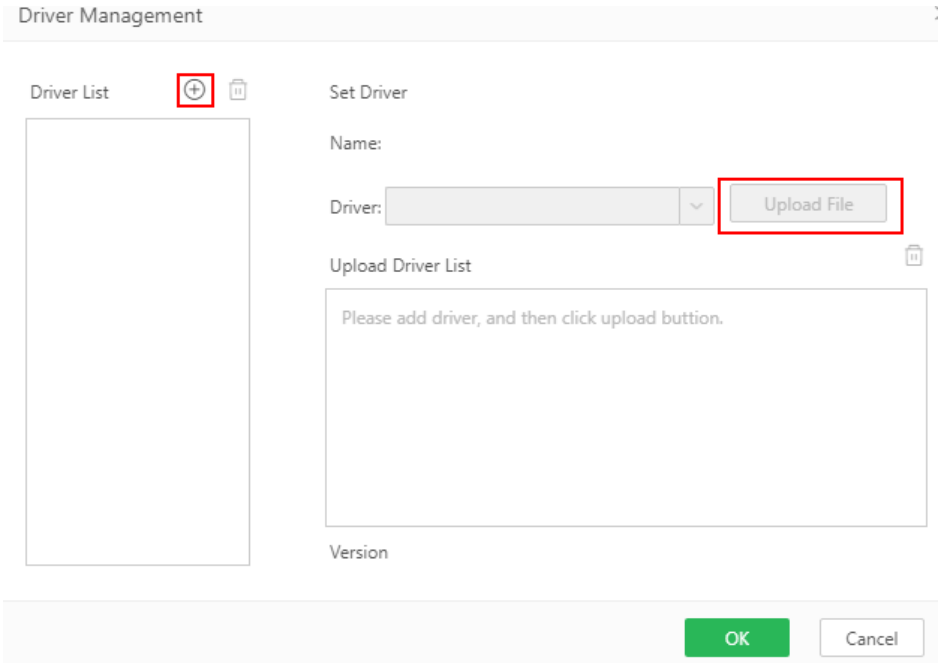


Step 2 On the data source configuration page, set parameters by referring to **Figure 4-17**. **User** and **Password** are the username and password of the created human-machine user. After the configuration is complete, click **Test Connection**.

Figure 4-17 Configuring the data source



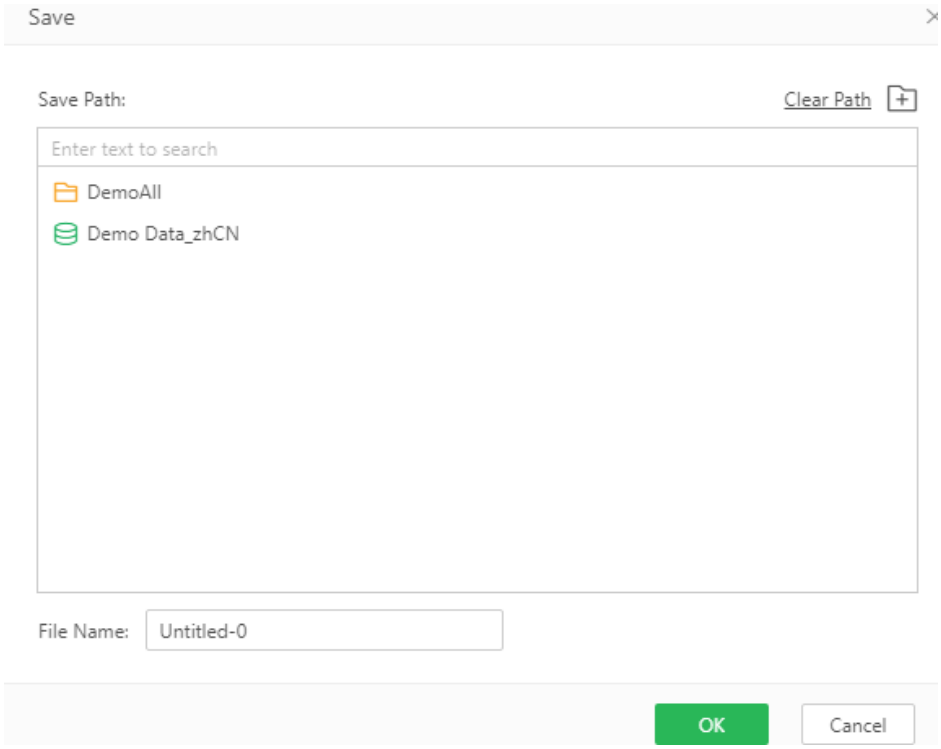
- **Driver**: Choose **Custom** > **Select Custom Driver**. Click , edit the driver name, click **Upload File** to upload the obtained JDBC JAR file, and click **OK**.



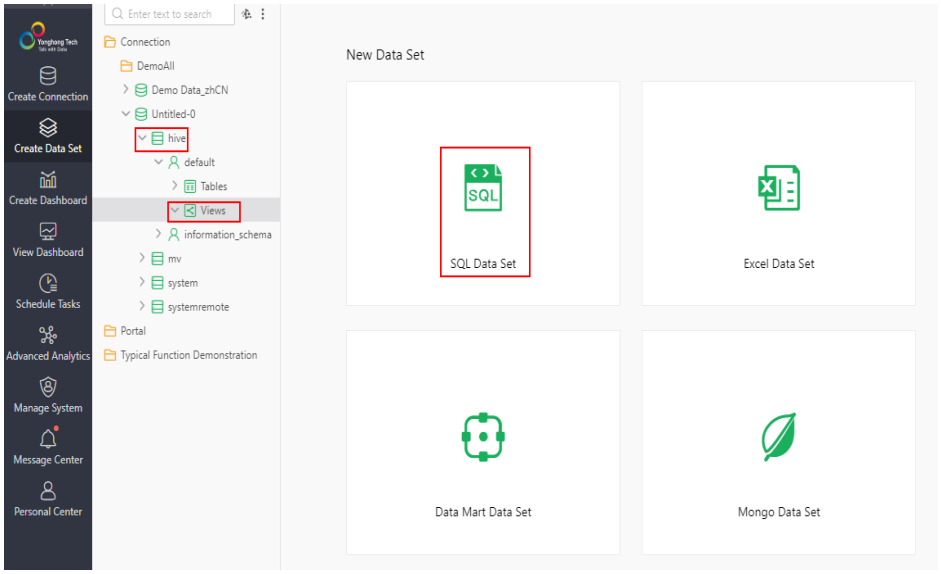
- **URL**: Enter the URL required for using either HSFabric or HSBroker. For details, see **Table 4-1**.
- **Server Login**: Select **Username and Password** and enter the username and password.

Step 3 Click **New Data Set**. On the page that is displayed, modify the save path and change the file name by referring to **Figure 4-18**, and click **OK**.

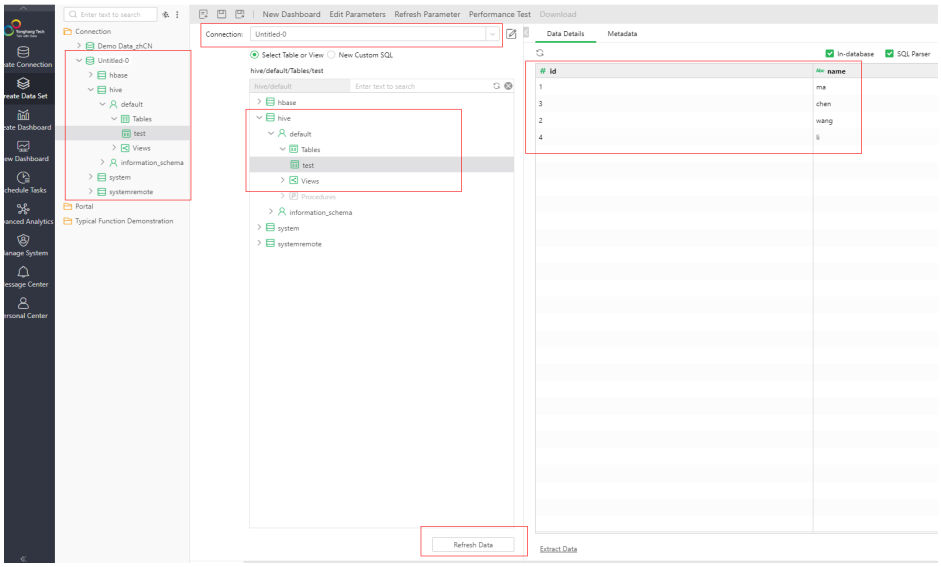
Figure 4-18 Modifying the save path and changing the file name



Step 4 Select the file name of the data set created in **Step 3** under **DemoAll**. The default file name **Untitled-0** is used as an example. Choose **Untitled-0** > **hive** > **default** > **Views** and select **SQL Data Set** under **New Data Set** in the right pane.



Step 5 In the **Connection** area, select the new data set created in **Step 3**. All table information is displayed. Select a table, for example, **test**, and click **Refresh Data**. All table information is displayed in the **Data Details** area on the right.



----End

4.5 Interconnecting Hive with External Self-Built Relational Databases

NOTE

- This section applies to MRS 3.*x* or later.
- This section describes how to connect Hive with built-in relational databases open-source MySQL and Postgres.
- After an external metadata database is deployed in a cluster with Hive data, the original metadata tables will not be automatically synchronized. Before installing Hive, determine whether to store metadata in an external database or DBService. For the former, deploy an external database when installing Hive or when there is no Hive data. After Hive installation, the metadata storage location cannot be changed. Otherwise, the original metadata will be lost.
- After external metadata is imported to the MySQL database, Hive supports only table names, field names, and table description in Chinese.

Hive supports access to open source MySQL and Postgres metabases.

Step 1 Install the open source MySQL or Postgres database.

NOTE

The node where the database is installed must be in the same network segment as the cluster, so that they can access each other.

Step 2 Upload the driver package.

- PostgreSQL:

  Use the open source driver package to replace the cluster's existing one. Download the open source PostgreSQL driver package **postgresql-42.2.5.jar** at **https://repo1.maven.org/maven2/org/postgresql/postgresql/42.2.5/** and upload it to the **${BIGDATA_HOME}/third_lib/Hive** directory on all MetaStore nodes.

Run the following commands on all MetaStore nodes to modify the permission on the driver package:

cd ${BIGDATA_HOME}/third_lib/Hive

chown omm:wheel postgresql-42.2.5.jar

chmod 600 postgresql-42.2.5.jar

- MySQL:

  Visit the MySQL official website at **https://www.mysql.com/**, choose **DOWNLOADS** > **MySQL Community(GPL) DownLoads** > **Connector/J**, and download the driver package of the required version.

  - For versions earlier than MRS 8.2.0, upload the MySQL driver package of the required version to the **/opt/Bigdata/FusionInsight_HD_*/install/FusionInsight-Hive-*/hive-*/lib/** directory on all Metastore nodes.
  - For MRS 8.2.0 and later versions, upload the MySQL driver package of the required version to the **${BIGDATA_HOME}/third_lib/Hive** directory on all Metastore nodes.

  Run the following commands on all MetaStore nodes to modify the permission on the driver package:

  cd /opt/Bigdata/FusionInsight_HD_*/install/FusionInsight-Hive-*/hive-*/lib/

  chown omm:wheel mysql-connector-java-*.jar

  chmod 600 mysql-connector-java-*.jar

Step 3 Create a user and metadata database in the self-built database and assign all permissions on the database to the user. For example:

- Run the following commands as the database administrator in PostgreSQL to create database **test** and user **testuser**, and assign all permissions on **test** to **testuser**:

  create user testuser with password '*password***';**

  create database test owner testuser;

  grant all privileges on database test to testuser;

- Run the following commands as the database administrator in MySQL to create database **test** and user **testuser**, and assign all permissions on **test** to **testuser**:

  create database test;

  create user 'testuser'@'%' identified by '*password***';**

  grant all privileges on test.* to 'testuser';

  flush privileges;

Step 4 Import the SQL statements for creating metadata tables.

- SQL script path in the PostgreSQL database: **${BIGDATA_HOME}/FusionInsight_HD_*/install/FusionInsight-Hive-*/hive-*/scripts/metastore/upgrade/postgres/hive-schema-3.1.0.postgres.sql**

  Run the following command to import the SQL file to Postgres:

  ./bin/psql -U *username* **-d** *databasename* **-f hive-schema-3.1.0.postgres.sql**

  ./bin/psql is in the Postgres installation directory.

  username indicates the username for logging in to Postgres.

databasename indicates the database name.

- SQL script path in the MySQL database: **${BIGDATA_HOME}/FusionInsight_HD_*/install/FusionInsight-Hive-*/hive-*/scripts/metastore/upgrade/mysql/hive-schema-3.1.0.mysql.sql**

  Run the following command to import the SQL file to the MySQL database:

  ./bin/mysql -u *username* **-p -D** *databasename***<hive-schema-3.1.0.mysql.sql**

  ./bin/mysql is in the MySQL installation directory.

  username indicates the user name for logging in to MySQL.

  databasename indicates the database name.

Step 5 Log in to FusionInsight Manager, choose **Cluster** > **Services**, and click **Hive**. On the displayed page, click **Configuration** > **All Configurations**, and choose **Hive (Service)** > **MetaDB**. Modify the parameters described in **Table1 Parameters**, and save the modification so that the Hive configuration can be connected to the open-source database.

Table 4-2 Parameters

Parameter	Default value	Description
javax.jdo.option.ConnectionDriverName	org.postgresql.Driver	Driver class for connecting metadata on MetaStore • If an external MySQL database is used, the value is: com.mysql.jdbc.Driver • If an external Postgres database is used, the value is: org.postgresql.Driver

Parameter	Default value	Description
javax.jdo.option.ConnectionURL	jdbc:postgresql://%{DBSERVICE_FLOAT_IP}%{DBServer}:%{DBSERVICE_CPORT}/hivemeta?socketTimeout=60	URL of the JDBC link of the MetaStore metadata • If an external MySQL database is used, the value is: **jdbc:mysql://***IP address of the MySQL database*:*Port number of the MySQL database*/**test?characterEncoding=utf-8** • If an external Postgres database is used, the value is: **jdbc:postgresql://***IP address of the PostgreSQL database*:*Port number of the PostgreSQL database*/**test** **NOTE** **test** is the name of the database created in MySQL or PostgreSQL in **Step 3**.
javax.jdo.option.ConnectionUserName	hive${SERVICE_INDEX}${SERVICE_INDEX}	Username for connecting to the metadata database on Metastore

Step 6 Change the Postgres database password in MetaStore. Choose **Cluster** > ***Name of the desired cluster*** > **Services** > **Hive**. On the displayed page, click **Configurations** > **All Configurations** and choose **MetaStore(Role)** > **MetaDB**, modify the following parameters, and click **Save**.

Table 4-3 Parameters

Parameter	Default value	Description
javax.jdo.option.extend.ConnectionPassword	******	User password for connecting to the external metadata database on Metastore. The password is encrypted in the background.

Step 7 Log in to each MetaStore background node and check whether the local directory **/opt/Bigdata/tmp** exists.

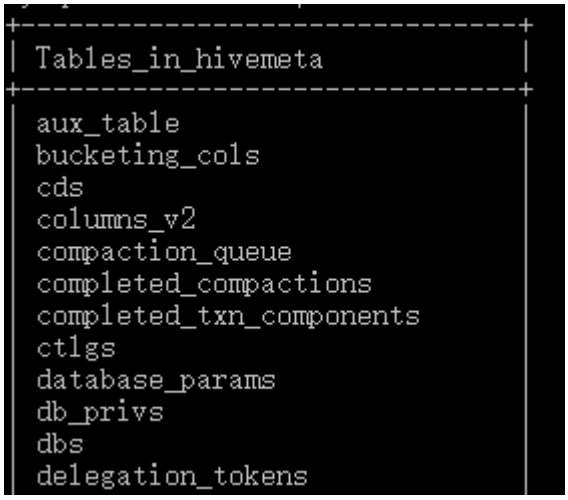
- If it exists, go to **Step 8**.
- If it is not, run the following commands to create one:

  mkdir -p /opt/Bigdata/tmp

  chmod 755 /opt/Bigdata/tmp

Step 8 Save the configuration. Choose **Dashboard** > **More** > **Restart Service**, and enter the password to restart the Hive service.

Step 9 Log in to the MySQL or PostgreSQL database and view metadata tables generated in the metadata database created in **Step 3**.



Step 10 Check whether the metadata database is successfully deployed.

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

   cd *Client installation directory*

   source bigdata_env

   kinit *Component service user* (Skip this step for clusters with Kerberos authentication disabled.)

2. Run the following command to log in to the Hive client CLI:

   beeline

3. Run the following command to create the **test** table:

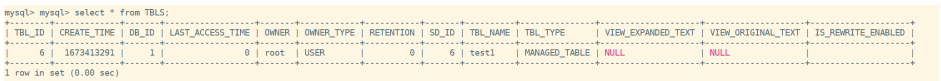
   create table *test*(**id int,str1 string,str2 string);**

4. Run the following command in the **test** database of the MySQL or PostgreSQL database to check whether there is any information about the **test** table:

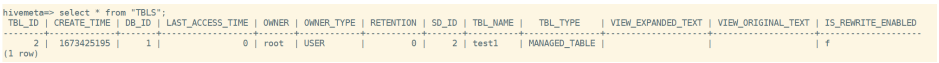
   select * from TBLS;

   If information about the **test** table is displayed, the external database is successfully deployed. For example:

   – The result in the MySQL database is as follows:



– The result in the PostgreSQL database is as follows:



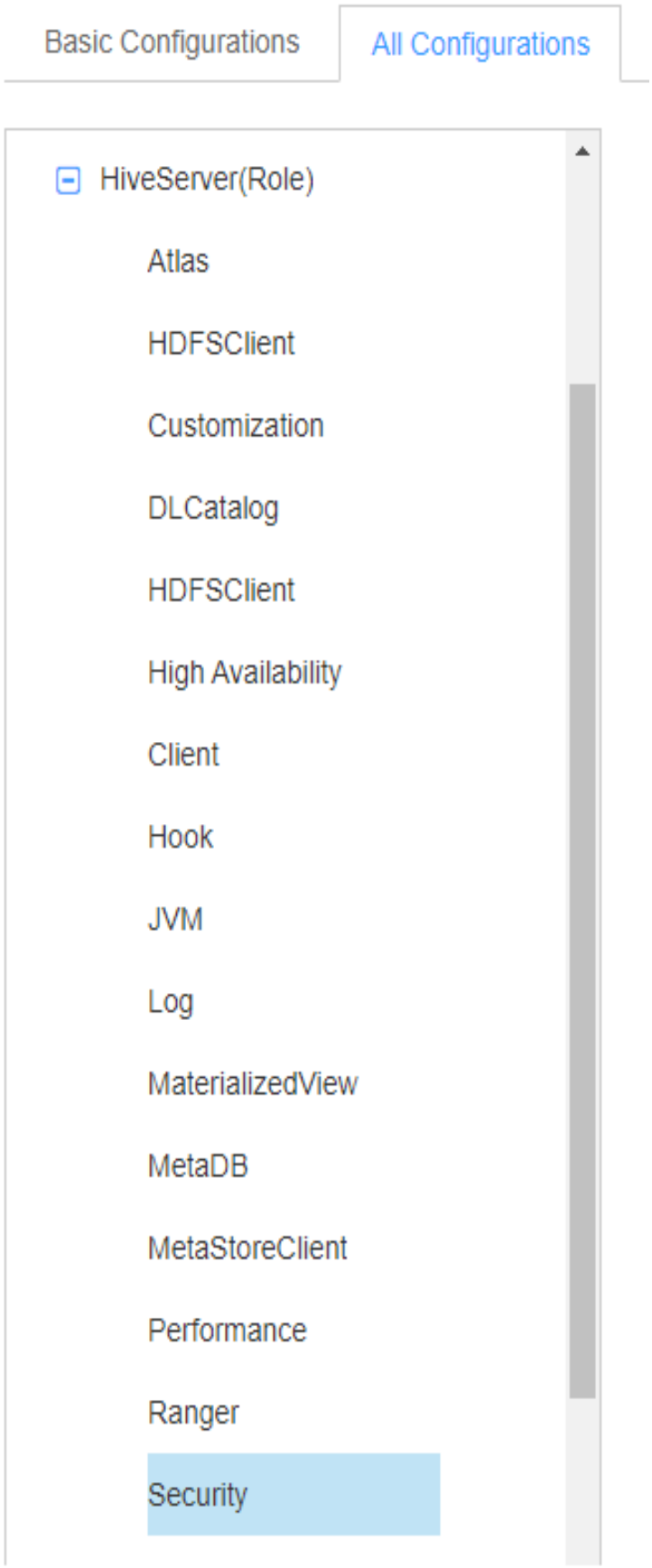
----End

4.6 Interconnecting Hive with External LDAP

This section applies to MRS 3.1.0 or later.

Step 1 Log in to FusionInsight Manager.

Step 2 On FusionInsight Manager, choose **Cluster** > *Name of the desired cluster* > **Services** > **Hive**. On the page that is displayed, click the **Configurations** tab then the **All Configurations** sub-tab. On this sub-tab page, click **HiveServer(Role)** and select **Security**.



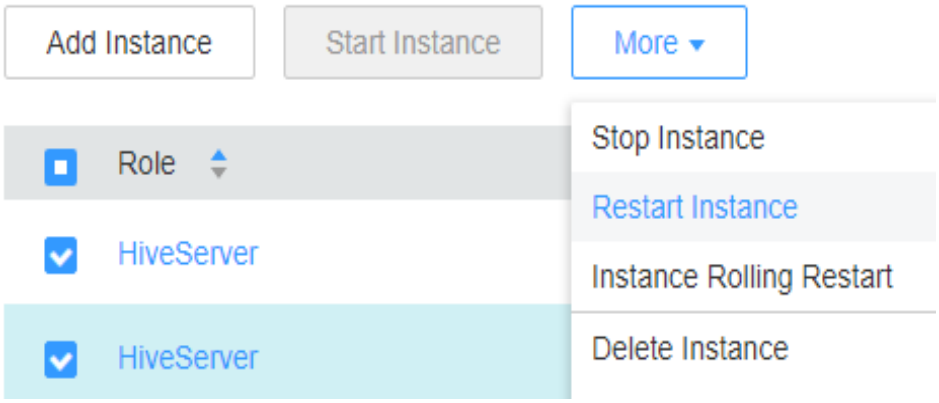
Step 3 Set the following parameters.

Table 4-4 Parameter configuration

Parameter	Description	Remarks
hive.server2.authentication	HiveServer authentication mode	Value: **KERBEROS** or **LDAP** Default value: **KERBEROS**
hive.server2.authentication.ldap.baseDN	LDAP base DN	-
hive.server2.authentication.ldap.password	LDAP password	LDAP password used for health check
hive.server2.authentication.ldap.url.ip	LDAP IP address	-
hive.server2.authentication.ldap.url.port	LDAP port number	Default value: **389**
hive.server2.authentication.ldap.userDNPattern	LDAP user DN pattern	Separate multiple values with colons (:), for example, **cn=%s,ou=People1,dc=huawei,dc=com: cn=%s,ou=People2,dc=huawei,dc=com**.
hive.server2.authentication.ldap.username	LDAP username	LDAP username used for health check

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** > **Hive** > **Instance**. On the displayed page, select the instances whose **Configuration Status** is **Expired**, choose **More** > **Restart Instance**, and restart the instance.



----End

4.7 Interconnecting MRS Kafka with Kafka Eagle

Introduction to Kafka Eagle

Kafka Eagle is a distributed and highly available Kafka monitoring software. It provides a range of Kafka monitoring metrics, such as the number of brokers, topics, consumers, Topic LogSize Top10, Topic Capacity Top10, Lag squeeze, and CPU/memory of Kafka clusters.

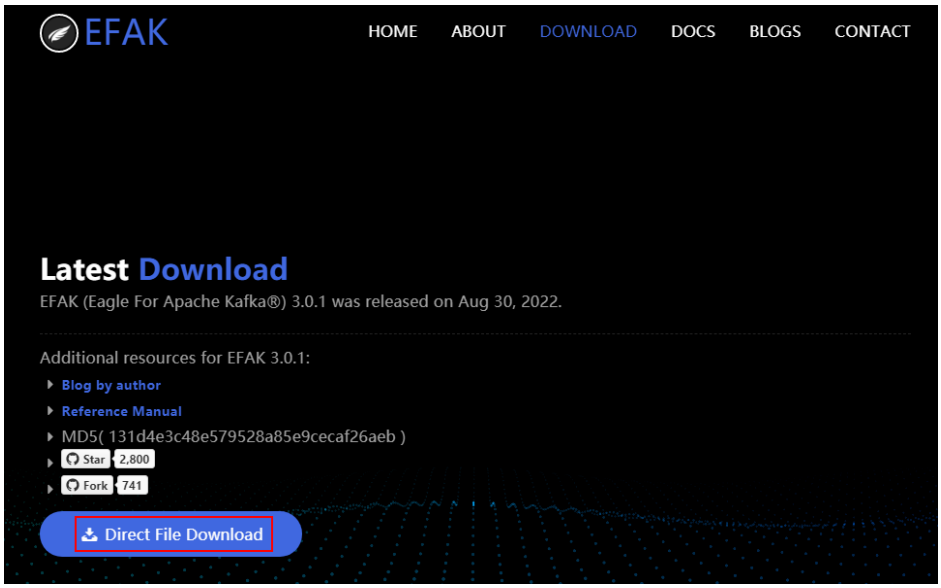
Eagle is renamed EFAK in the new version.

Prerequisites

- You have bought an MRS 3.1.0 cluster that contains the Kafka component and enabled Kerberos authentication for the cluster. For details, see **Buying a Custom Cluster**.
- The MRS cluster client has been installed. For details, see **Installing a Client**.

Kafka Eagle Installation Procedure

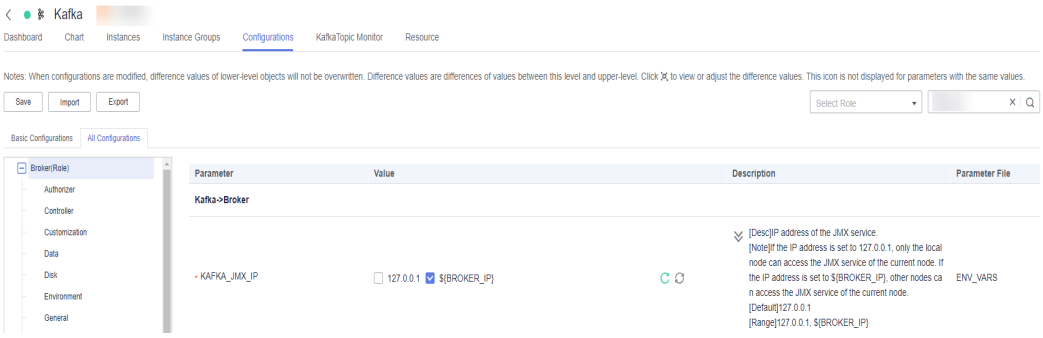
Step 1 Download **Kafka Eagle**. The following uses EFAK3.0.1 as an example.



For example, download the **kafka-eagle-bin-3.0.1.tar.gz** software package.

Step 2 Log in to FusionInsight Manager, choose **Cluster** > **Services** > **Kafka**, click **Configurations**, and then **All Configurations**. Search for **KAFKA_JMX_IP** and change the value to **${BROKER_IP}**.

Figure 4-19 Modifying Kafka parameters



Step 3 Click **Save** in the upper left corner. In the displayed dialog box, click **OK**.

Step 4 Click the **Dashboard** tab and choose **More** > **Restart Service** in the upper right corner to restart the Kafka service.

Step 5 Log in to the active node of the cluster as the **root** user, save the obtained EFAK installation package **kafka-eagle-bin-3.0.1.tar.gz** to the cluster directory, for example, **/opt**, and run the following command to decompress the package:

cd /opt

tar -xvf kafka-eagle-bin-3.0.1.tar.gz

cd kafka-eagle-bin-3.0.1

tar -xvf efak-web-3.0.1-bin.tar.gz

Step 6 Create a directory in the **opt** directory, for example, **efak**, and copy **efak-web-3.0.1** to the **/opt/efak** directory.

mkdir /opt/efak

cp -r /opt/kafka-eagle-bin-3.0.1/efak-web-3.0.1 /opt/efak/

Step 7 Add environment variables.

vi /etc/profile

Add the **export KE_HOME** parameter. The parameter value is the path of the **efak-web-3.0.1** file (example value: **/opt/efak/efak-web-3.0.1**). Add **$KE_HOME/bin** to the end of the **export PATH** value. The following is an example:

```
export KE_HOME=/opt/efak/efak-web-3.0.1
export PATH=$PATH:$KE_HOME/bin
```

Step 8 Modify the **system-config.properties** configuration file.

cd /opt/efak/efak-web-3.0.1/conf/

vi system-config.properties

```
# Configure a cluster.
kafka.eagle.zk.cluster.alias=cluster1
cluster1.zk.list=10.20.90.24:2181
#cluster2.zk.list=xdn10:2181,xdn11:2181,xdn12:2181
# Modify kafka jmx uri.
cluster1.efak.jmx.uri=service:jmx:rmi:///jndi/rmi://%s/kafka
# Modify the database configuration.
efak.driver=com.mysql.cj.jdbc.Driver
```



```
efak.url=jdbc:mysql://IP:Port/ke?
useUnicode=true&characterEncoding=UTF-8&zeroDateTimeBehavior=convertToNull
efak.username=root
efak.password=XXX
```

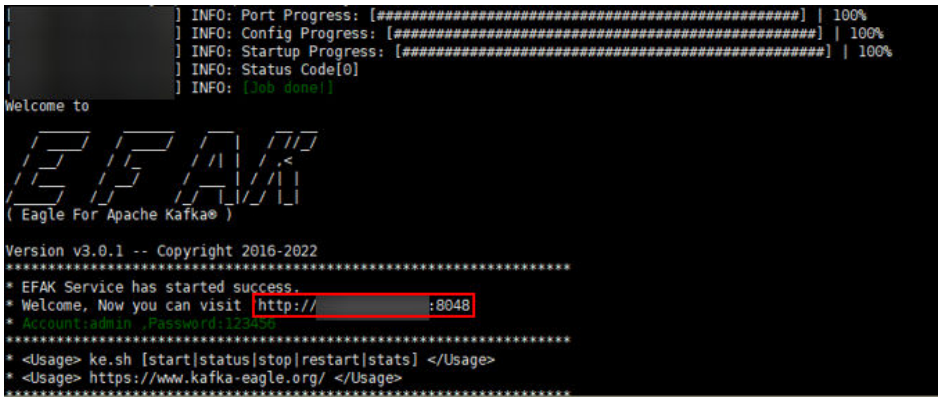
NOTE

- The value of **cluster1.zk.list** is the value of the Kafka component parameter **metrics.reporter.zookeeper.url**. To obtain the value, you can log in to FusionInsight Manager, choose **Cluster** > **Services** > **Kafka**, click **Configurations**, and then **All Configurations**. Search for **metrics.reporter.zookeeper.url**.
- The value of **efak.url** is the string representation of the MySQL JDBC connection.
- The value of **efak.username** is the username for connecting to the database.
- The value of **efak.password** is the password of the username for connecting to the database.

Step 9 Start the EFAK service.

sh /opt/efak/efak-web-3.0.1/bin/ke.sh start

Obtain the EFAK web UI login address from the command output.



Step 10 Log in to the EFAK web UI.

NOTE

The default initial username for logging in to the EFKA web UI is **admin** and the password is **123456**.

You can access the Kafka cluster monitoring page, topic monitoring page, and consumer monitoring page.

Figure 4-20 Cluster monitoring

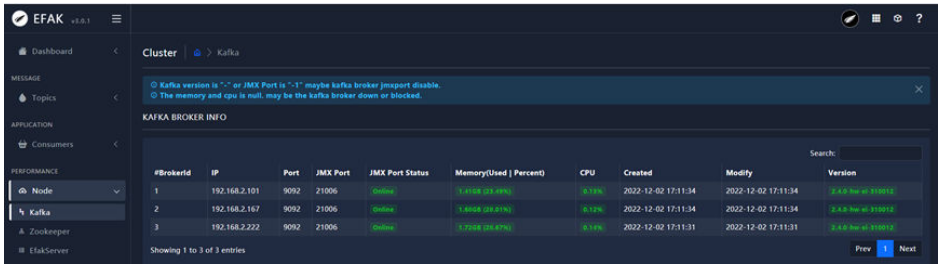
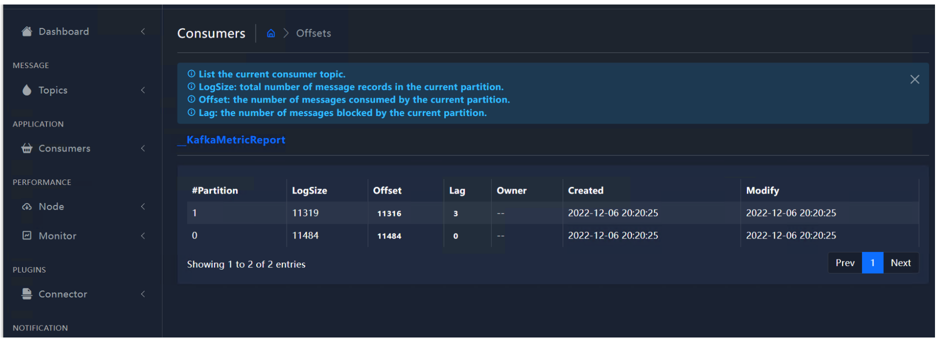


Figure 4-21 Topic monitoring

Figure 4-22 Consumer monitoring



----End

FAQs

Symptom:

The Kafka CPU and memory monitoring information cannot be obtained.

```
java.io.IOException cannot be cast to javax.management.remote.JMXConnector
```

Possible Cause

The JMX URI is incorrectly configured. The default JMX URI is as follows:

```
cluster1.efak.jmx.uri=service:jmx:rmi:///jndi/rmi://%s/jmxrmi
```

Solution

If the Kafka JMX name in MRS is **kafka**, set the URI parameter as follows:

```
cluster1.efak.jmx.uri=service:jmx:rmi:///jndi/rmi://%s/kafka
```

4.8 Using Jupyter Notebook to Connect to MRS Spark

Overview

MRS allows you to boost your machine learning, data exploration, and ETL application development efficiency by using PySpark with Jupyter Notebook.

This practice describes how to configure Jupyter Notebook in an MRS cluster to use PySpark.

The detailed steps are as follows:

1. **Step 1: Install the Client on a Node Outside the MRS Cluster**
2. **Step 2: Install Python3**
3. **Step 3: Install Jupyter Notebook.**
4. **Step 4: Verify Jupyter Notebook access to MRS**

NOTE

This practice is available for MRS 3.x and later versions where Python3 is installed on the client node outside the cluster.

Step 1: Install the Client on a Node Outside the MRS Cluster

Step 1 Prepare a Linux ECS that does not belong to the MRS cluster, associate an EIP to the ECS, and install the cluster client by referring to **Installing a Client on a Node Outside the Cluster**. For example, the installation directory is **/opt/client**.

Step 2 Check whether Kerberos authentication is enabled for the desired MRS cluster.

- If yes, go to **Step 3**.
- If no, go to **Step 2: Install Python3**.

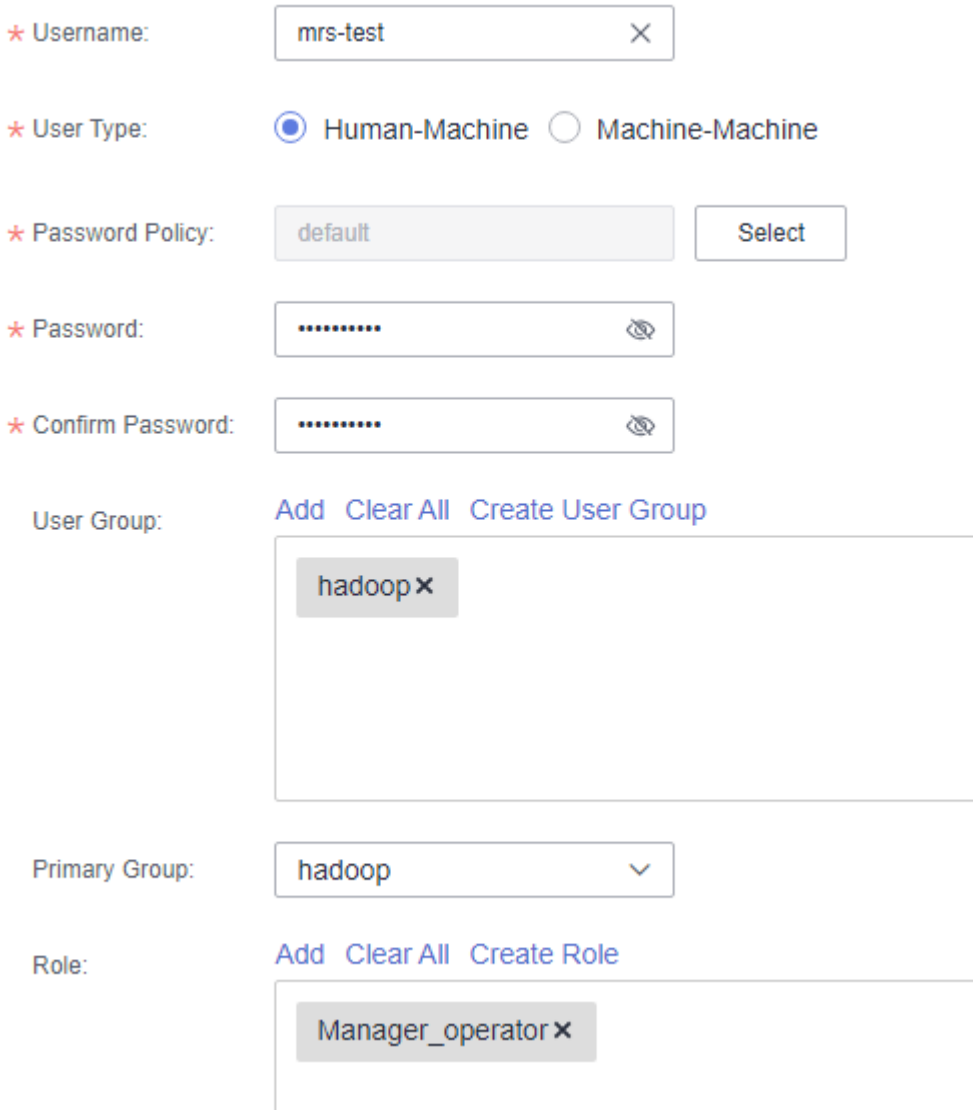
Step 3 Log in to FusionInsight Manager of the cluster.

Step 4 Choose **System** > **Permission** > **User** to create a service user.

Set **User Type** to **Human-Machine**, add **hadoop** to **User Group**, select **hadoop Primary Group**, and add **Manager_operator** to **Role**.

For example, the created user is **mrs-test**.

Figure 4-23 Creating an MRS service user



Step 5 Log in to the cluster client node as user **root** and run the following commands to configure environment variables and authenticate the user. Change the user password upon the first user authentication.

source /opt/client/bigdata_env

kinit *mrs-test*

----End

Step 2: Install Python3

Step 1 Log in to the client node outside the cluster as user **root** and run the following command to check whether Python3 is installed:

python3 --version

```
[root@ecs-notebook FusionInsight_Cluster_1_Services_ClientConfig]# python3 --version
-bash: python3: command not found
```


- If yes, go to **Step 8**.
- If no, go to **Step 2**.

NOTE

This case is available only when **Python3 is installed on the client node outside the cluster**.

Step 2 Install Python. Python 3.6.6 is used as an example.

1. Install the following dependencies:

   yum install zlib zlib-devel zip -y

   yum install gcc-c++

   yum install openssl-devel

   yum install sqlite-devel -y

   If the pandas library requires the following dependencies:

   yum install -y xz-devel

   yum install bzip2-devel

2. Download the source code of the corresponding Python version.

   wget https://www.python.org/ftp/python/3.6.6/Python-3.6.6.tgz

3. Decompress the Python source code package, for example, to the **opt** directory.

   cd /opt

   tar -xvf Python-3.6.6.tgz

4. Create a Python installation directory, for example, **/opt/python36**.

   mkdir /opt/python36

5. Compile Python.

   cd /opt/python-3.6.6

   ./configure --prefix=/opt/python36

   The following information is displayed if the commands are executed successfully.

```
configure: creating ./config.status
config.status: creating Makefile.pre
config.status: creating Modules/Setup.config
config.status: creating Misc/python.pc
config.status: creating Misc/python-config.sh
config.status: creating Modules/ld_so_aix
config.status: creating pyconfig.h
creating Modules/Setup
creating Modules/Setup.local
creating Makefile


If you want a release build with all stable optimizations active (PGO, etc),
please run ./configure --enable-optimizations
```

   Run the **make -j8** command. If the command is successfully executed, the following information is displayed.


```
creating build/scripts-3.6
copying and adjusting /tmp/python366/Python-3.6.6/Tools/scripts/pydoc3 -> build/scripts-3.6
copying and adjusting /tmp/python366/Python-3.6.6/Tools/scripts/idle3 -> build/scripts-3.6
copying and adjusting /tmp/python366/Python-3.6.6/Tools/scripts/2to3 -> build/scripts-3.6
copying and adjusting /tmp/python366/Python-3.6.6/Tools/scripts/pyvenv -> build/scripts-3.6
changing mode of build/scripts-3.6/pydoc3 from 644 to 755
changing mode of build/scripts-3.6/idle3 from 644 to 755
changing mode of build/scripts-3.6/2to3 from 644 to 755
changing mode of build/scripts-3.6/pyvenv from 644 to 755
renaming build/scripts-3.6/pydoc3 to build/scripts-3.6/pydoc3.6
renaming build/scripts-3.6/idle3 to build/scripts-3.6/idle3.6
renaming build/scripts-3.6/2to3 to build/scripts-3.6/2to3-3.6
renaming build/scripts-3.6/pyvenv to build/scripts-3.6/pyvenv-3.6
```

Run the **make install** command. If the command is successfully executed, the following information is displayed.

```
rm -f /opt/python36/share/man/man1/python3.1
(cd /opt/python36/share/man/man1; ln -s python3.6.1 python3.1)
if test "xupgrade" != "xno"  ; then \
        case upgrade in \
                upgrade) ensurepip="--upgrade" ;; \
                install|*) ensurepip="" ;; \
        esac; \
         ./python -E -m ensurepip \
                $ensurepip --root=/ ; \
fi
Looking in links: /tmp/tmp6ldv525m
Collecting setuptools
Collecting pip
Installing collected packages: setuptools, pip
Successfully installed pip-10.0.1 setuptools-39.0.1
```

6. Configure Python environment variables.

   export PYTHON_HOME=/opt/python36

   export PATH=$PYTHON_HOME/bin:$PATH

7. Run the **python3 --version** command. If the following information is displayed, Python has been installed.

   ```
   Python 3.6.6
   ```

Step 3 Verify Python 3.

pip3 install helloword

python3

import helloworld

helloworld.say_hello("test")

```
[root@ecs-notebook Python-3.6.6]# pip3 install helloword
Collecting helloword
  Downloading https://files.pythonhosted.org/packages/1b/bf/f0f69f122158e0e98b5d95987a7ef5add3f8a340c6eb78d5871f855ca04e/helloword-0.0.1-py3-none-any.whl
Installing collected packages: helloword
Successfully installed helloword-0.0.1
You are using pip version 10.0.1, however version 21.3.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
[root@ecs-notebook Python-3.6.6]# python3
Python 3.6.6 (default, Dec 15 2021, 06:12:40)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-44)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import helloworld
helloworld.say_hello("test")Hello, Sara!
>>>
'Hello, test!'
>>>
```

Step 4 Install third-party Python libraries (such as pandas and sklearn).

pip3 install pandas


```
[root@ecs-mrs-test Python-3.6.6]# pip3 install pandas
Collecting pandas
  Downloading https://files.pythonhosted.org/packages/c3/e2/00cacecafbab071c787019f00ad84ca3185952f6bb9bca9550ed83870d4d/pandas-1.1.5-cp36-cp36m-man
_x86_64.whl (9.5MB)
    100% |                                | 9.5MB 6.5MB/s
Collecting pytz>=2017.2 (from pandas)
  Downloading https://files.pythonhosted.org/packages/60/2e/dec1cc18c51b8df33c7c4d0a321b084cf38e1733b98f9d15018880fb4970/pytz-2022.1-py2.py3-none-an
503kB)
    100% |                                | 512kB 47.2MB/s
Collecting python-dateutil>=2.7.3 (from pandas)
  Downloading https://files.pythonhosted.org/packages/36/7a/87837f39d0296e723bb9b62bbb257d0355c7f6128853c78955f57342a56d/python_dateutil-2.8.2-py2.p
-any.whl (247kB)
    100% |                                | 256kB 54.5MB/s
Collecting numpy>=1.15.4 (from pandas)
  Downloading https://files.pythonhosted.org/packages/45/b2/6c7545bb7a38754d63048c7696804a0d947328125d81bf12beaa692c3ae3/numpy-1.19.5-cp36-cp36m-man
_x86_64.whl (13.4MB)
    100% |                                | 13.4MB 4.2MB/s
Collecting six>=1.5 (from python-dateutil>=2.7.3->pandas)
  Downloading https://files.pythonhosted.org/packages/d9/5a/e7c31adbe875f2abbb91bd84cf2dc52d792b5a01506781dbcf25c91daf11/six-1.16.0-py2.py3-none-any
Installing collected packages: pytz, six, python-dateutil, numpy, pandas
Successfully installed numpy-1.19.5 pandas-1.1.5 python-dateutil-2.8.2 pytz-2022.1 six-1.16.0
You are using pip version 10.0.1, however version 21.3.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
```

pip3 install backports.lzma

```
[root@ecs-mrs-test Python-3.6.6]# pip3 install backports.lzma
Collecting backports.lzma
  Using cached https://files.pythonhosted.org/packages/21/0f/1a9990233076d48aa2084100ba289ca162975e73a688f3a56c0ee2bb441a/backports.lzma-0.0.14.tar.gz
Installing collected packages: backports.lzma
  Running setup.py install for backports.lzma ... done
Successfully installed backports.lzma-0.0.14
You are using pip version 10.0.1, however version 21.3.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
```

pip3 install sklearn

```
[root@ecs-mrs-test Python-3.6.6]# pip3 install sklearn
Collecting sklearn
  Downloading https://files.pythonhosted.org/packages/1e/7a/dbb3be0ce9bd5c8b7e3d87328e79063f8b263b2b1bfa4774cb1147bfcd3f/sklearn-0.0.tar.
Collecting scikit-learn (from sklearn)
  Downloading https://files.pythonhosted.org/packages/f5/ef/bcd79e8d59250d6e8478eb1290dc6e05be42b3be8a86e3954146adbc171a/scikit_learn-0.2
ylinux1_x86_64.whl (20.0MB)
    100% |                                | 20.0MB 3.4MB/s
Collecting joblib>=0.11 (from scikit-learn->sklearn)
  Downloading https://files.pythonhosted.org/packages/3e/d5/0163eb0cfa0b673aa4fe1cd3ea9d8a81ea0f32e50807b0c295871e4aab2e/joblib-1.1.0-py2
(306kB)
    100% |                                | 307kB 46.5MB/s
Requirement already satisfied: scipy>=0.19.1 in /root/.local/lib/python3.6/site-packages (from scikit-learn->sklearn) (1.5.4)
Collecting threadpoolctl>=2.0.0 (from scikit-learn->sklearn)
  Downloading https://files.pythonhosted.org/packages/61/cf/6e354304bcb9c6413c4e02a747b600061c21d38ba51e7e544ac7bc66aecc/threadpoolctl-3.
nl
Requirement already satisfied: numpy>=1.13.3 in /opt/python36/lib/python3.6/site-packages (from scikit-learn->sklearn) (1.19.5)
Installing collected packages: joblib, threadpoolctl, scikit-learn, sklearn
  Running setup.py install for sklearn ... done
Successfully installed joblib-1.1.0 scikit-learn-0.24.2 sklearn-0.0 threadpoolctl-3.1.0
You are using pip version 10.0.1, however version 21.3.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
```

Step 5 Run the **python3 -m pip list** command to check the installation result.

```
[root@ecs-mrs-test Python-3.6.6]# python3 -m pip list
Package         Version
--------------- -------
cycler          0.11.0
joblib          1.1.0
kiwisolver      1.3.1
numpy           1.19.5
pandas          1.1.5
pip             10.0.1
pyparsing       3.0.7
python-dateutil 2.8.2
pytz            2022.1
scikit-learn    0.24.2
scipy           1.5.4
setuptools      39.0.1
six             1.16.0
sklearn         0.0
threadpoolctl   3.1.0
```

Step 6 Pack them into **Python.zip**.

cd /opt/python36/

zip -r python36.zip ./*

Step 7 Upload the file to the specified HDFS directory.

hdfs dfs -mkdir /user/python

hdfs dfs -put python36.zip /user/python

Step 8 Configure the MRS client.

Go to the Spark client installation directory **/opt/client/Spark2x/spark/conf** and configure the following parameters in the **spark-defaults.conf** file:


```
spark.pyspark.driver.python=/usr/bin/python3
spark.yarn.dist.archives=hdfs://hacluster/user/python/python36.zip#Python
```

----End

Step 3: Install Jupyter Notebook.

Step 1 Log in to the client node as user **root** and run the following command to install Jupyter Notebook:

pip3 install jupyter notebook

The installation is successful if the following command output is displayed.

Step 2 For security purpose, generate a ciphertext password for logging in to Jupyter and save the password in the Jupyter Notebook configuration file.

Run the following command and enter the password twice (exit at Out[3]):

ipython

```
[root@ecs-notebook python36]# ipython
Python 3.6.6 (default, Dec 20 2021, 09:32:25)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.16.2 -- An enhanced Interactive Python. Type '?' for help.
In [1]: from notebook.auth import passwd
In [2]: passwd()
Enter password:
Verify password:
Out[2]: 'argon2:$argon2id$v=19$m=10240,t=10,p=8$g14BqLddl927n/unsyPlLQ
$YmoKJzbUfNG7LcxylJzm90bgbKWUIiHy6ZV+ObTzdcA
```

Step 3 Generate the Jupyter configuration file.

jupyter notebook --generate-config

Step 4 Modify the configuration file.

vi ~/.jupyter/jupyter_notebook_config.py

Add the following configurations:

```
# -*- coding: utf-8 -*-
c.NotebookApp.ip='*' #Enter the internal IP address of the ECS.
c.NotebookApp.password = u'argon2:$argon2id$v=19$m=10240,t=10,p=8$NmoAVwd8F6vFP2rX5ZbV7w
$SyueJoC0a5TbCuHYzqfSx1vQcFvOTTryR+0uk2MNNZA' # Enter the ciphertext generated at Out[2] in step 2.
c.NotebookApp.open_browser = False # Disable automatic browser opening.
c.NotebookApp.port = 9999 # Specified port number
c.NotebookApp.allow_remote_access = True
```

----End

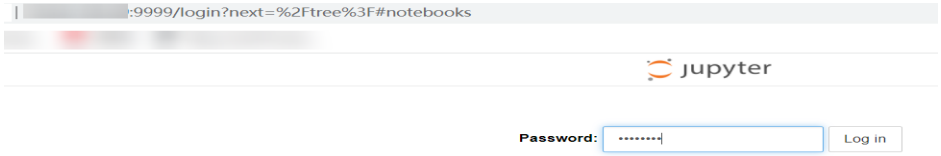
Step 4: Verify Jupyter Notebook access to MRS

Step 1 Run the following command on the client node to start Jupyter Notebook:

PYSPARK_PYTHON=./Python/bin/python3 PYSPARK_DRIVER_PYTHON=jupyter-notebook PYSPARK_DRIVER_PYTHON_OPTS="--allow-root" pyspark --master yarn --executor-memory 2G --driver-memory 1G

Step 2 Enter *EIP*:**9999** in the address box of the browser to log in to the Jupyter web UI (ensure that the security group of the ECS allows access from the local public IP address and port 9999). The login password is the one set in **Step 2**.

Figure 4-24 Logging In to the Jupyter web UI



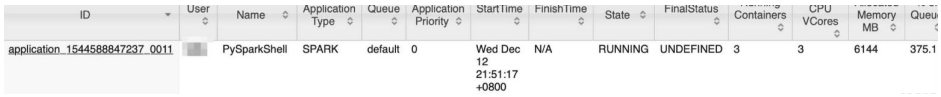
Step 3 Create code.

Create a python3 task and use Spark to read files.

Figure 4-25 Creating a Python task

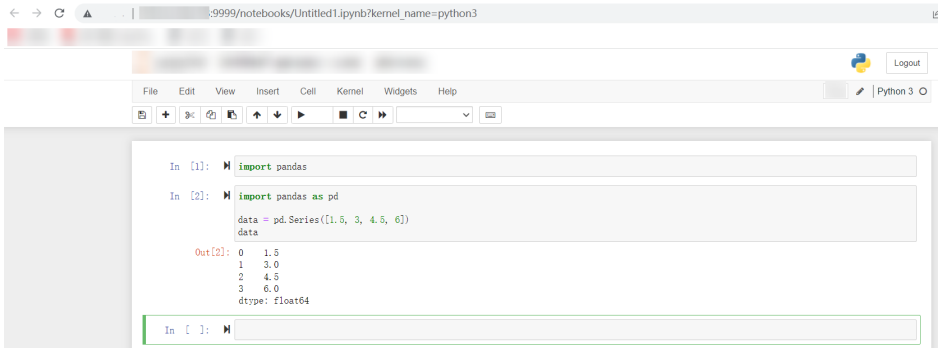
Log in to FusionInsight Manager and view the submitted PySpark application on the Yarn web UI.

Figure 4-26 Viewing the task status



Step 4 Verify that the pandas library can be called.

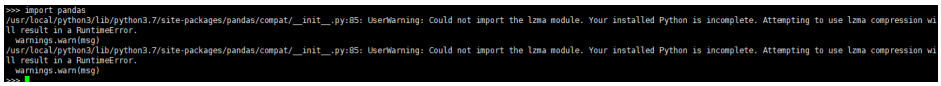
Figure 4-27 Verifying pandas



----End

FAQs About Interconnection with Jupyter

When the pandas is used for local import, the following error message is displayed.



Perform the following steps to rectify the fault:

Step 1 Run the **python -m pip install backports.lzma** command to install the LZMA module.

```
[root@master ~]# python -m pip install backports.lzma
Looking in indexes: http://mirrors.aliyun.com/pypi/simple/
Requirement already satisfied: backports.lzma in /usr/local/python3/lib/python3.7/site-packages (0.0.14)
You are using pip version 10.0.1, however version 19.3.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
```

Step 2 Go to the **/usr/local/python3/lib/python3.6** directory and edit the **lzma.py** file. The directory varies depending on hosts. You can run the **which** command to query the directory used by Python.

Change

```
from _lzma import *
from _lzma import _encode_filter_properties, _decode_filter_properties
```

To

```
try:
    from _lzma import *
    from _lzma import _encode_filter_properties, _decode_filter_properties
except ImportError:
    from backports.lzma import *
    from backports.lzma import _encode_filter_properties, _decode_filter_properties
```

Before modification

```
"""Interface to the liblzma compression library.

This module provides a class for reading and writing compressed files,
classes for incremental (de)compression, and convenience functions for
one-shot (de)compression.

These classes and functions support both the XZ and legacy LZMA
container formats, as well as raw compressed data streams.
"""

__all__ = [
    "CHECK_NONE", "CHECK_CRC32", "CHECK_CRC64", "CHECK_SHA256",
    "CHECK_ID_MAX", "CHECK_UNKNOWN",
    "FILTER_LZMA1", "FILTER_LZMA2", "FILTER_DELTA", "FILTER_X86", "FILTER_IA64",
    "FILTER_ARM", "FILTER_ARMTHUMB", "FILTER_POWERPC", "FILTER_SPARC",
    "FORMAT_AUTO", "FORMAT_XZ", "FORMAT_ALONE", "FORMAT_RAW",
    "MF_HC3", "MF_HC4", "MF_BT2", "MF_BT3", "MF_BT4",
    "MODE_FAST", "MODE_NORMAL", "PRESET_DEFAULT", "PRESET_EXTREME",

    "LZMACompressor", "LZMADecompressor", "LZMAFile", "LZMAError",
    "open", "compress", "decompress", "is_check_supported",
]

import builtins
import io
import os
from _lzma import *
from _lzma import _encode_filter_properties, _decode_filter_properties
import _compression
```

After modification


```
These classes and functions support both the XZ and legacy LZMA
container formats, as well as raw compressed data streams.
"""

__all__ = [
    "CHECK_NONE", "CHECK_CRC32", "CHECK_CRC64", "CHECK_SHA256",
    "CHECK_ID_MAX", "CHECK_UNKNOWN",
    "FILTER_LZMA1", "FILTER_LZMA2", "FILTER_DELTA", "FILTER_X86", "FILTER_IA64",
    "FILTER_ARM", "FILTER_ARMTHUMB", "FILTER_POWERPC", "FILTER_SPARC",
    "FORMAT_AUTO", "FORMAT_XZ", "FORMAT_ALONE", "FORMAT_RAW",
    "MF_HC3", "MF_HC4", "MF_BT2", "MF_BT3", "MF_BT4",
    "MODE_FAST", "MODE_NORMAL", "PRESET_DEFAULT", "PRESET_EXTREME",

    "LZMACompressor", "LZMADecompressor", "LZMAFile", "LZMAError",
    "open", "compress", "decompress", "is_check_supported",
]

import builtins
import io
import os
#from _lzma import *
#from _lzma import _encode_filter_properties, _decode_filter_properties
try:
    from _lzma import *
    from _lzma import _encode_filter_properties, _decode_filter_properties
except ImportError:
    from backports.lzma import *
    from backports.lzma import _encode_filter_properties, _decode_filter_properties
import _compression
```

Step 3 Save the settings and exit, and then run the **import** command again.

```
[root@master python3.7]# python
Python 3.7.0 (default, Oct 26 2019, 01:19:22)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-36)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pandas
>>>
```

----End

5 MRS Cluster Management

5.1 Configuring Thresholds for Alarms

MRS clusters provide easy-to-use alarming functions with intuitive monitoring metric views. You can quickly view statistics on key performance metrics (KPIs) of a cluster and evaluate the cluster health status. MRS allows you to configure metric thresholds to stay informed of cluster health status. If a threshold value is met, the system generates and displays an alarm on the metric dashboard.

If it is **verified** that the impact of some alarms on services can be ignored or the alarm thresholds need to be adjusted, you can customize cluster metrics or mask some alarms as required.

You can set thresholds for alarms of node information metrics and cluster service metrics. For details about these metrics, their impacts on the system, and default thresholds, see **Monitoring Metric Reference**.

NOTICE

These alarms may affect cluster functions or job running. If you want to mask or modify alarm rules, evaluate operation risks in advance.

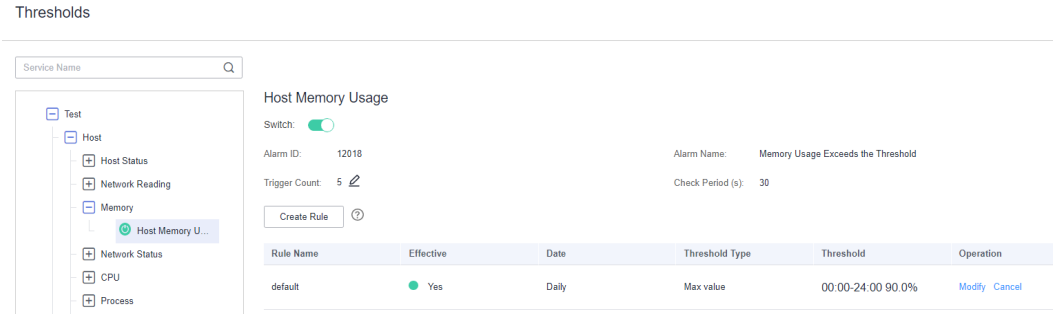
Modifying Rules for Alarms with Custom Thresholds

Step 1 Log in to FusionInsight Manager of the target MRS cluster by referring to **Accessing Log in the FusionInsight Manager (MRS 3.x or Later)**.

Step 2 Choose **O&M** > **Alarm** > **Thresholds**.

Step 3 Select a metric for a host or service in the cluster. For example, select **Host Memory Usage**.

Figure 5-1 Viewing an alarm threshold



- **Switch**: If this switch is turned on, an alarm will be triggered when the metric breaches this threshold.
- **Trigger Count**: Manager checks whether the metric meets the threshold value. If the number of consecutive checks where the metric fails equals the value of **Trigger Count**, an alarm is generated. The value can be customized. **If an alarm is frequently reported, you can set Trigger Count to a larger value to reduce the alarming frequency.**
- **Check Period (s)**: Interval between each two checks
- The rules to trigger alarms are listed on the page.

Step 4 Modify an alarm rule.

- Add a new rule.
  a. Click **Create Rule** to add a rule that defines how an alarm will be triggered. For details, see **Table 5-1**.
  b. Click **OK** to save the rule.
  c. Locate the row that contains a rule that is in use, and click **Cancel** in the **Operation** column. If no rule is in use, skip this step.
  d. Locate the row that contains the new rule, and click **Apply** in the **Operation** column. The value of **Effective** for this rule changes to **Yes**.
- Modify an existing rule.
  a. Click **Modify** in the **Operation** column of the row that contains the target rule.
  b. Modify rule parameters by referring to **Table 5-1**.
  c. Click **OK**.

The following table lists the rule parameters you need to set for triggering an alarm of **Host Memory Usage**.

Table 5-1 Alarm rule parameters

Parameter	Description	Example Value
Rule Name	Rule name	mrs_test

Parameter	Description	Example Value
Severity	Alarm severity. The options are as follows: • **Critical** • **Major** • **Minor** • **Warning**	Major
Threshold Type	Maximum or minimum value of a metric • **Max value**: An alarm will be generated when the metric value is greater than this value. • **Min value**: An alarm will be generated when the metric value is less than this value.	Max. Value
Date	How often the rule takes effect • Daily • Weekly • Others	Daily
Add Date	Date when the rule takes effect. This parameter is available only when **Date** is set to **Others**. You can set multiple dates.	-
Thresholds	**Start and End Time**: Period when the rule takes effect.	00:00 - 23:59
	Threshold: Alarm threshold value	85

----End

Masking Specified Alarms

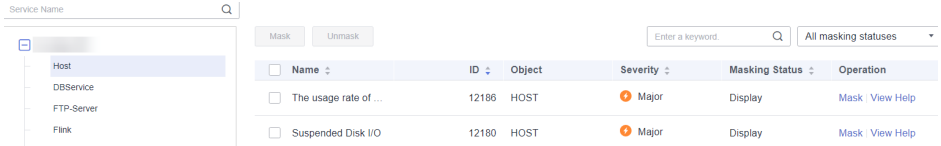
Step 1 Log in to FusionInsight Manager of the target MRS cluster by referring to **Accessing Log in the FusionInsight Manager (MRS 3.x or Later)**.

Step 2 Choose **O&M** > **Alarm** > **Masking**.

Step 3 In the list on the left of the displayed page, select the target service or module.

Step 4 Click **Mask** in the **Operation** column of the alarm you want to mask. In the dialog box that is displayed, click **OK** to change the masking status of the alarm to **Masking**.

Figure 5-2 Masking an alarm



NOTE

- You can search for specified alarms in the list.
- To cancel alarm masking, click **Unmask** in the row of the target alarm. In the dialog box that is displayed, click **OK** to change the alarm masking status to **Display**.
- If you need to perform operations on multiple alarms at a time, select the alarms and click **Mask** or **Unmask** on the top of the list.

----End

FAQ

- **How Do I View Uncleared Alarms in a Cluster?**
  a. Log in to the MRS management console.
  b. Click the name of the target cluster and click the **Alarms** tab.
  c. Click **Advanced Search**, set **Alarm Status** to **Uncleared**, and click **Search**.
  d. Uncleared alarms of the current cluster are displayed.
- **How Do I Clear a Cluster Alarm?**
  You can handle the alarms by referring to the alarm help. To view the help document, perform the following steps:
  – Console: Log in to the MRS management console, click the name of the target cluster, click the **Alarms** tab, and click **View Help** in the **Operation** column of the alarm list. Then, clear the alarm by referring to the alarm handling procedure.
  – Manager: Log in to FusionInsight Manager, choose **O&M** > **Alarm** > **Alarms**, and click **View Help** in the **Operation** column. Then, clear the alarm by referring to the alarm handling procedure.

Monitoring Metric Reference

FusionInsight Manager monitoring metrics are classified as node information metrics and cluster service metrics. **Table 5-2** lists the metrics whose thresholds can be configured a node, and **Table 5-3** lists metrics whose thresholds can be configured for a component.

Table 5-2 Node monitoring metrics and corresponding alarms

Metric Group	Metric	ID	Alarm	Impact on System	Default Threshold
CPU	Host CPU Usage	12016	CPU Usage Exceeds the Threshold	Service processes respond slowly or become unavailable.	90.0%
Disk	Disk Usage	12017	Insufficient Disk Capacity	Service processes become unavailable.	90.0%

Metric Group	Metric	ID	Alarm	Impact on System	Default Threshold
	Disk Inode Usage	12051	Disk Inode Usage Exceeds the Threshold	Data cannot be properly written to the file system.	80.0%
Memory	Host Memory Usage	12018	Memory Usage Exceeds the Threshold	Service processes respond slowly or become unavailable.	90.0%
Host Status	Host File Handle Usage	12053	Host File Handle Usage Exceeds the Threshold	The I/O operations, such as opening a file or connecting to network, cannot be performed and programs are abnormal.	80.0%
	Host PID Usage	12027	Host PID Usage Exceeds the Threshold	No PID is available for new processes and service processes are unavailable.	90%
Network Status	TCP Temporary Port Usage	12052	TCP Temporary Port Usage Exceeds the Threshold	Services on the host fail to establish connections with the external and services are interrupted.	80.0%
Network Reading	Read Packet Error Rate	12047	Read Packet Error Rate Exceeds the Threshold	The communication is intermittently interrupted, and services time out.	0.5%
	Read Packet Dropped Rate	12045	Read Packet Dropped Rate Exceeds the Threshold	The service performance deteriorates or some services time out.	0.5%
	Read Throughput Rate	12049	Read Throughput Rate Exceeds the Threshold	The service system runs abnormally or is unavailable.	80%

Metric Group	Metric	ID	Alarm	Impact on System	Default Threshold
Network Writing	Write Packet Error Rate	12048	Write Packet Error Rate Exceeds the Threshold	The communication is intermittently interrupted, and services time out.	0.5%
	Write Packet Dropped Rate	12046	Write Packet Dropped Rate Exceeds the Threshold	The service performance deteriorates or some services time out.	0.5%
	Write Throughput Rate	12050	Write Throughput Rate Exceeds the Threshold	The service system runs abnormally or is unavailable.	80%
Process	Total Number of Processes in D and Z States	12028	Number of Processes in the D State and Z State on a Host Exceeds the Threshold	Excessive system resources are used and service processes respond slowly.	0
	omm Process Usage	12061	Process Usage Exceeds the Threshold	Switch to user **omm** fails. New omm process cannot be created.	90

Table 5-3 Cluster monitoring metrics and corresponding alarms

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
DBService	Usage of the Number of Database Connections	27005	Database Connection Usage Exceeds the Threshold	Upper-layer services may fail to connect to the DBService database, affecting services.	90%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	Disk Space Usage of the Data Directory	27006	Disk Space Usage of the Data Directory Exceeds the Threshold	Service processes become unavailable. When the disk space usage of the data directory exceeds 90%, the database enters the read-only mode and **Database Enters the Read-Only Mode** is generated. As a result, service data is lost.	80%
Flume	Heap Memory Resource Percentage	24006	Heap Memory Usage of Flume Server Exceeds the Threshold	Heap memory overflow may cause service breakdown.	95.0%
	Direct Memory Usage Statistics	24007	Flume Server Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause service breakdown.	80.0%
	Non-heap Memory Usage	24008	Flume Server Non-Heap Memory Usage Exceeds the Threshold	Non-heap memory overflow may cause service breakdown.	80.0%
	Total GC Duration	24009	Flume Server GC Duration Exceeds the Threshold	Flume data transmission efficiency decreases.	12000 ms
HBase	GC Duration of Old Generation	19007	HBase GC Duration Exceeds the Threshold	If the old generation GC duration exceeds the threshold, HBase data read and write are affected.	5000ms

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	RegionServer Direct Memory Usage Statistics	19009	Direct Memory Usage of the HBase Process Exceeds the Threshold	If the available HBase direct memory is insufficient, a memory overflow occurs and the service breaks down.	90%
	RegionServer Heap Memory Usage Statistics	19008	Heap Memory Usage of the HBase Process Exceeds the Threshold	If the available HBase memory is insufficient, a memory overflow occurs and the service breaks down.	90%
	HMaster Direct Memory Usage	19009	Direct Memory Usage of the HBase Process Exceeds the Threshold	If the available HBase direct memory is insufficient, a memory overflow occurs and the service breaks down.	90%
	HMaster Heap Memory Usage Statistics	19008	Heap Memory Usage of the HBase Process Exceeds the Threshold	If the available HBase memory is insufficient, a memory overflow occurs and the service breaks down.	90%
	Number of Online Regions of a RegionServer	19011	Number of RegionServer Regions Exceeds the Threshold	The data read/write performance of HBase is affected when the number of regions on a RegionServer exceeds the threshold.	2000
	Region in RIT State That Reaches the Threshold Duration	19013	Duration of Regions in RIT State Exceeds the Threshold	Some data in the table is lost or becomes unavailable.	1

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	Handler Usage of RegionServer	19021	Number of Active Handlers of RegionServer Exceeds the Threshold	RegionServers or HBase cannot provide services properly.	90%
	Synchronization Failures in Disaster Recovery	19006	HBase Replication Sync Failed	HBase data in a cluster fails to be synchronized to the standby cluster, causing data inconsistency between active and standby clusters.	1
	Number of Log Files to Be Synchronized in the Active Cluster	19020	Number of HBase WAL Files to Be Synchronized Exceeds the Threshold	If the number of WAL files to be synchronized by a RegionServer exceeds the threshold, the number of ZNodes used by HBase exceeds the threshold, affecting the HBase service status.	128
	Number of HFiles to Be Synchronized in the Active Cluster	19019	Number of HFiles to Be Synchronized Exceeds the Threshold	If the number of HFiles to be synchronized by a RegionServer exceeds the threshold, the number of ZNodes used by HBase exceeds the threshold, affecting the HBase service status.	128

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	Compaction Queue Size	19018	HBase Compaction Queue Size Exceeds the Threshold	The cluster performance may deteriorate, affecting data read and write.	100
HDFS	Lost Blocks	14003	Number of Lost HDFS Blocks Exceeds the Threshold	Data stored in HDFS is lost. HDFS may enter the security mode and cannot provide write services. Lost block data cannot be restored.	0
	Blocks Under Replicated	14028	Number of Blocks to Be Supplemented Exceeds the Threshold	Data stored in HDFS is lost. HDFS may enter the security mode and cannot provide write services. Lost block data cannot be restored.	1000
	Average Time of Active NameNode RPC Processing	14021	Average NameNode RPC Processing Time Exceeds the Threshold	NameNode cannot process the RPC requests from HDFS clients, upper-layer services that depend on HDFS, and DataNode in a timely manner. Specifically, the services that access HDFS run slowly or the HDFS service is unavailable.	100ms

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	Average Time of Active NameNode RPC Queuing	14022	Average NameNode RPC Queuing Time Exceeds the Threshold	NameNode cannot process the RPC requests from HDFS clients, upper-layer services that depend on HDFS, and DataNode in a timely manner. Specifically, the services that access HDFS run slowly or the HDFS service is unavailable.	200ms
	HDFS Disk Usage	14001	HDFS Disk Usage Exceeds the Threshold	The performance of writing data to HDFS is affected.	80%
	DataNode Disk Usage	14002	DataNode Disk Usage Exceeds the Threshold	Insufficient disk space will impact data write to HDFS.	80%
	Percentage of Reserved Space for Replicas of Unused Space	14023	Percentage of Total Reserved Disk Space for Replicas Exceeds the Threshold	The performance of writing data to HDFS is affected. If all unused DataNode space is reserved for replicas, writing HDFS data fails.	90%
	Total Faulty DataNodes	14009	Number of Dead DataNodes Exceeds the Threshold	Faulty DataNodes cannot provide HDFS services.	3
	NameNode Non-Heap Memory Usage Statistics	14018	NameNode Non-Heap Memory Usage Exceeds the Threshold	If the non-heap memory usage of the HDFS NameNode is too high, data read/write performance of HDFS will be affected.	90%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	NameNode Direct Memory Usage Statistics	14017	NameNode Direct Memory Usage Exceeds the Threshold	If the available direct memory of NameNode instances is insufficient, a memory overflow may occur and the service breaks down.	90%
	NameNode Heap Memory Usage Statistics	14007	NameNode Heap Memory Usage Exceeds the Threshold	If the heap memory usage of the HDFS NameNode is too high, data read/write performance of HDFS will be affected.	95%
	DataNode Direct Memory Usage Statistics	14016	DataNode Direct Memory Usage Exceeds the Threshold	If the available direct memory of DataNode instances is insufficient, a memory overflow may occur and the service breaks down.	90%
	DataNode Heap Memory Usage Statistics	14008	DataNode Heap Memory Usage Exceeds the Threshold	The HDFS DataNode heap memory usage is too high, which affects the data read/write performance of the HDFS.	95%
	DataNode Non-Heap Memory Usage Statistics	14019	DataNode Non-Heap Memory Usage Exceeds the Threshold	If the non-heap memory usage of the HDFS DataNode is too high, data read/write performance of HDFS will be affected.	90%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	NameNode GC Duration Statistics	14014	NameNode GC Duration Exceeds the Threshold	A long GC duration of the NameNode process may interrupt the services.	12000 ms
	DataNode GC Duration Statistics	14015	DataNode GC Duration Exceeds the Threshold	A long GC duration of the DataNode process may interrupt the services.	12000 ms
Hive	Hive SQL Execution Success Rate (Percentage)	16002	Hive SQL Execution Success Rate Is Lower Than the Threshold	The system configuration and performance cannot meet service processing requirements.	90.0%
	Background Thread Usage	16003	Background Thread Usage Exceeds the Threshold	There are too many background threads, so the newly submitted task cannot run in time.	90%
	Total GC Duration of MetaStore	16007	Hive GC Duration Exceeds the Threshold	If the GC duration exceeds the threshold, Hive data read and write are affected.	12000 ms
	Total GC Duration of HiveServer	16007	Hive GC Duration Exceeds the Threshold	If the GC duration exceeds the threshold, Hive data read and write are affected.	12000 ms
	Percentage of HDFS Space Used by Hive to the Available Space	16001	Hive Warehouse Space Usage Exceeds the Threshold	The system fails to write data, which causes data loss.	85.0%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	MetaStore Direct Memory Usage Statistics	16006	Direct Memory Usage of the Hive Process Exceeds the Threshold	When the direct memory usage of Hive is overhigh, the performance of Hive task operation is affected. In addition, a memory overflow may occur so that the Hive service is unavailable.	95%
	MetaStore Non-Heap Memory Usage Statistics	16008	Non-heap Memory Usage of the Hive Service Exceeds the Threshold	When the non-heap memory usage of Hive is overhigh, the performance of Hive task operation is affected. In addition, a memory overflow may occur so that the Hive service is unavailable.	95%
	MetaStore Heap Memory Usage Statistics	16005	Heap Memory Usage of the Hive Process Exceeds the Threshold	When the heap memory usage of Hive is overhigh, the performance of Hive task operation is affected. In addition, a memory overflow may occur so that the Hive service is unavailable.	95%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	HiveServer Direct Memory Usage Statistics	16006	Direct Memory Usage of the Hive Process Exceeds the Threshold	When the direct memory usage of Hive is overhigh, the performance of Hive task operation is affected. In addition, a memory overflow may occur so that the Hive service is unavailable.	95%
	HiveServer Non-Heap Memory Usage Statistics	16008	Non-heap Memory Usage of the Hive Service Exceeds the Threshold	When the non-heap memory usage of Hive is overhigh, the performance of Hive task operation is affected. In addition, a memory overflow may occur so that the Hive service is unavailable.	95%
	HiveServer Heap Memory Usage Statistics	16005	Heap Memory Usage of the Hive Process Exceeds the Threshold	When the heap memory usage of Hive is overhigh, the performance of Hive task operation is affected. In addition, a memory overflow may occur so that the Hive service is unavailable.	95%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	Percentage of Sessions Connected to the HiveServer to Maximum Number of Sessions Allowed by the HiveServer	16000	Percentage of Sessions Connected to the HiveServer to Maximum Number Allowed Exceeds the Threshold	If a connection alarm is generated, too many sessions are connected to the HiveServer and new connections cannot be created.	90.0%
Kafka	Percentage of Partitions That Are Not Completely Synchronized	38006	Percentage of Kafka Partitions That Are Not Completely Synchronized Exceeds the Threshold	Too many Kafka partitions that are not completely synchronized affect service reliability. In addition, data may be lost when leaders are switched.	50%
	User Connection Usage on Broker	38011	User Connection Usage on Broker Exceeds the Threshold	If the number of connections of a user is excessive, the user cannot create new connections to the Broker.	80%
	Broker Disk Usage	38001	Insufficient Kafka Disk Capacity	Kafka data write operations fail.	80.0%
	Disk I/O Rate of a Broker	38009	Busy Broker Disk I/Os	The disk partition has frequent I/Os. Data may fail to be written to the Kafka topic for which the alarm is generated.	80%
	Broker GC Duration per Minute	38005	GC Duration of the Broker Process Exceeds the Threshold	A long GC duration of the Broker process may interrupt the services.	12000 ms

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	Heap Memory Usage of Kafka	38002	Kafka Heap Memory Usage Exceeds the Threshold	If the available Kafka heap memory is insufficient, a memory overflow occurs and the service breaks down.	95%
	Kafka Direct Memory Usage	38004	Kafka Direct Memory Usage Exceeds the Threshold	If the available direct memory of the Kafka service is insufficient, a memory overflow occurs and the service breaks down.	95%
Loader	Heap Memory Usage	23004	Loader Heap Memory Usage Exceeds the Threshold	Heap memory overflow may cause service breakdown.	95%
	Direct Memory Usage Statistics	23006	Loader Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause service breakdown.	80.0%
	Non-heap Memory Usage	23005	Loader Non-Heap Memory Usage Exceeds the Threshold	Non-heap memory overflow may cause service breakdown.	80%
	Total GC Duration	23007	GC Duration of the Loader Process Exceeds the Threshold	Loader service response is slow.	12000 ms
MapReduce	GC Duration Statistics	18012	JobHistoryServer GC Duration Exceeds the Threshold	A long GC duration of the JobHistoryServer process may interrupt the services.	12000 ms

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	JobHistoryServer Direct Memory Usage Statistics	18015	JobHistoryServer Direct Memory Usage Exceeds the Threshold	If the available direct memory of the MapReduce service is insufficient, a memory overflow occurs and the service breaks down.	90%
	JobHistoryServer Non-Heap Memory Usage Statistics	18019	Non-Heap Memory Usage of JobHistoryServer Exceeds the Threshold	When the non-heap memory usage of MapReduce JobHistoryServer is overhigh, the performance of MapReduce task submission and operation is affected. In addition, a memory overflow may occur so that the MapReduce service is unavailable.	90%
	JobHistoryServer Heap Memory Usage Statistics	18009	Heap Memory Usage of JobHistoryServer Exceeds the Threshold	When the heap memory usage of MapReduce JobHistoryServer is overhigh, the performance of MapReduce log archiving is affected. In addition, a memory overflow may occur, leading to unavailable YARN service.	95%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
Oozie	Heap Memory Usage	17004	Oozie Heap Memory Usage Exceeds the Threshold	Heap memory overflow may cause service breakdown.	95.0%
	Direct Memory Usage	17006	Oozie Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause service breakdown.	80.0%
	Non-heap Memory Usage	17005	Oozie Non-Heap Memory Usage Exceeds the Threshold	Non-heap memory overflow may cause service breakdown.	80%
	Total GC Duration	17007	GC Duration of the Oozie Process Exceeds the Threshold	Oozie responds slowly when it is used to submit tasks.	12000 ms
Spark2x	JDBCServer2x Heap Memory Usage Statistics	43010	Heap Memory Usage of the JDBCServer2x Process Exceeds the Threshold	If available JDBCServe2x process heap memory is insufficient, a memory overflow occurs and the service breaks down	95%
	JDBCServer2x Direct Memory Usage Statistics	43012	Direct Heap Memory Usage of the JDBCServer2x Process Exceeds the Threshold	If the available JDBCServer2x Process direct heap memory is insufficient, a memory overflow occurs and the service breaks down.	95%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	JDBCServer2x Non-Heap Memory Usage Statistics	43011	Non-Heap Memory Usage of the JDBCServer2x Process Exceeds the Threshold	If the available JDBCServer2x Process non-heap memory is insufficient, a memory overflow occurs and the service breaks down.	95%
	JobHistory2x Direct Memory Usage Statistics	43008	Direct Memory Usage of the JobHistory2x Process Exceeds the Threshold	If the available JobHistory2x Process directmemory is insufficient, a memory overflow occurs and the service breaks down.	95%
	JobHistory2x Non-Heap Memory Usage Statistics	43007	Non-Heap Memory Usage of the JobHistory2x Process Exceeds the Threshold	If the available JobHistory2x Process non-heap memory is insufficient, a memory overflow occurs and the service breaks down.	95%
	JobHistory2x Heap Memory Usage Statistics	43006	Heap Memory Usage of the JobHistory2x Process Exceeds the Threshold	If the available JobHistory2x Process heap memory is insufficient, a memory overflow occurs and the service breaks down.	95%
	IndexServer2x Direct Memory Usage Statistics	43021	Direct Memory Usage of the IndexServer2x Process Exceeds the Threshold	If the available IndexServer2x process direct memory is insufficient, a memory overflow occurs and the service breaks down.	95%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	IndexServer2x Heap Memory Usage Statistics	43019	Heap Memory Usage of the IndexServer2x Process Exceeds the Threshold	If the available IndexServer2x process heap memory is insufficient, a memory overflow occurs and the service breaks down.	95%
	IndexServer2x Non-Heap Memory Usage Statistics	43020	Non-Heap Memory Usage of the IndexServer2x Process Exceeds the Threshold	If the available IndexServer2x process non-heap memory is insufficient, a memory overflow occurs and the service breaks down.	95%
	Full GC Number of JDBCServer2x	43017	JDBCServer2x Process Full GC Number Exceeds the Threshold	The performance of the JDBCServer2x process is affected, or even the JDBCServer2x process is unavailable.	12
	Full GC Number of JobHistory2x	43018	JobHistory2x Process Full GC Number Exceeds the Threshold	The performance of the JobHistory2x process is affected, or even the JobHistory2x process is unavailable.	12
	Full GC Number of IndexServer2x	43023	IndexServer2x Process Full GC Number Exceeds the Threshold	If the GC number exceeds the threshold, IndexServer2x maybe run in low performance or even unavailable.	12

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	Total GC Duration (in Milliseconds) of JDBCServer2x	43013	JDBCServer2x Process GC Duration Exceeds the Threshold	If the GC duration exceeds the threshold, JDBCServer2x maybe run in low performance.	12000 ms
	Total GC Duration (in Milliseconds) of JobHistory2x	43009	JobHistory2x Process GC Duration Exceeds the Threshold	If the GC duration exceeds the threshold, JobHistory2x may run in low performance.	12000 ms
	Total GC Duration (in Milliseconds) of IndexServer2x	43022	IndexServer2x Process GC Duration Exceeds the Threshold	If the GC duration exceeds the threshold, IndexServer2x may run in low performance or even unavailable.	12000 ms
Storm	Number of Available Supervisors	26052	Number of Available Supervisors of the Storm Service Is Less Than the Threshold	Existing tasks in the cluster cannot be performed. The cluster can receive new Storm tasks, but cannot perform these tasks.	1
	Slot Usage	26053	Storm Slot Usage Exceeds the Threshold	New Storm tasks cannot be performed.	80.0%
	Nimbus Heap Memory Usage	26054	Nimbus Heap Memory Usage Exceeds the Threshold	When the heap memory usage of Storm Nimbus is overhigh, frequent GCs occur. In addition, a memory overflow may occur so that the Yarn service is unavailable.	80%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
Yarn	NodeManager Direct Memory Usage Statistics	18014	NodeManager Direct Memory Usage Exceeds the Threshold	If the available direct memory of NodeManager is insufficient, a memory overflow occurs and the service breaks down.	90%
	NodeManager Heap Memory Usage Statistics	18018	NodeManager Heap Memory Usage Exceeds the Threshold	When the heap memory usage of Yarn NodeManager is overhigh, the performance of Yarn task submission and operation is affected. In addition, a memory overflow may occur so that the Yarn service is unavailable.	95%
	NodeManager Non-Heap Memory Usage Statistics	18017	NodeManager Non-heap Memory Usage Exceeds the Threshold	When the heap memory usage of Yarn NodeManager is overhigh, the performance of Yarn task submission and operation is affected. In addition, a memory overflow may occur so that the Yarn service is unavailable.	90%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	ResourceManager Direct Memory Usage Statistics	18013	ResourceManager Direct Memory Usage Exceeds the Threshold	If the available direct memory of ResourceManager is insufficient, a memory overflow occurs and the service breaks down.	90%
	ResourceManager Heap Memory Usage Statistics	18008	ResourceManager Heap Memory Usage Exceeds the Threshold	When the heap memory usage of Yarn ResourceManager is overhigh, the performance of Yarn task submission and operation is affected. In addition, a memory overflow may occur so that the Yarn service is unavailable.	95%
	ResourceManager Non-Heap Memory Usage Statistics	18016	ResourceManager Non-Heap Memory Usage Exceeds the Threshold	When the non-heap memory usage of Yarn ResourceManager is overhigh, the performance of Yarn task submission and operation is affected. In addition, a memory overflow may occur so that the Yarn service is unavailable.	90%
	NodeManager GC Duration Statistics	18011	NodeManager GC Duration Exceeds the Threshold	A long GC duration of the NodeManager process may interrupt the services.	12000 ms

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	ResourceManager GC Duration Statistics	18010	ResourceManager GC Duration Exceeds the Threshold	A long GC duration of the ResourceManager process may interrupt the services.	12000 ms
	Number of Failed Tasks in the Root Queue	18026	Number of Failed Yarn Tasks Exceeds the Threshold	A large number of application tasks fail to be executed. Failed tasks need to be submitted again.	50
	Terminated Applications of the Root Queue	18025	Number of Terminated Yarn Tasks Exceeds the Threshold	A large number of application tasks are forcibly stopped.	50
	Pending Memory	18024	Pending Yarn Memory Usage Exceeds the Threshold	It takes long time to end an application. A new application cannot run after submission.	8388608MB
	Pending Tasks	18023	Number of Pending Yarn Tasks Exceeds the Threshold	It takes long time to end an application. A new application cannot run for a long time after submission.	60
ZooKeeper	ZooKeeper Connections Usage	13001	Available ZooKeeper Connections Are Insufficient	Available ZooKeeper connections are insufficient. When the connection usage reaches 100%, external connections cannot be handled.	80%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	ZooKeeper Heap Memory Usage	13004	ZooKeeper Heap Memory Usage Exceeds the Threshold	If the available ZooKeeper memory is insufficient, a memory overflow occurs and the service breaks down.	95%
	ZooKeeper Direct Memory Usage	13002	ZooKeeper Direct Memory Usage Exceeds the Threshold	If the available ZooKeeper memory is insufficient, a memory overflow occurs and the service breaks down.	80%
	ZooKeeper GC Duration per Minute	13003	GC Duration of the ZooKeeper Process Exceeds the Threshold	A long GC duration of the ZooKeeper process may interrupt the services.	12000 ms
Ranger	UserSync GC Duration	45284	UserSync GC Duration Exceeds the Threshold	UserSync responds slowly.	12000 ms
	PolicySync GC Duration	45292	PolicySync GC Duration Exceeds the Threshold	PolicySync responds slowly.	12000 ms
	RangerAdmin GC Duration	45280	RangerAdmin GC Duration Exceeds the Threshold	RangerAdmin responds slowly.	12000 ms
	TagSync GC Duration	45288	TagSync GC Duration Exceeds the Threshold	TagSync responds slowly.	12000 ms

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	UserSync Non-Heap Memory Usage	45283	UserSync Non-Heap Memory Usage Exceeds the Threshold	Non-heap memory overflow may cause service breakdown.	80.0%
	UserSync Direct Memory Usage	45282	UserSync Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause service breakdown.	80.0%
	UserSync Heap Memory Usage	45281	UserSync Heap Memory Usage Exceeds the Threshold	Heap memory overflow may cause service breakdown.	95.0%
	PolicySync Direct Memory Usage	45290	PolicySync Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause service breakdown.	80.0%
	PolicySync Heap Memory Usage	45289	PolicySync Heap Memory Usage Exceeds the Threshold	Heap memory overflow may cause service breakdown.	95.0%
	PolicySync Non-Heap Memory Usage	45291	PolicySync Non-Heap Memory Usage Exceeds the Threshold	Non-heap memory overflow may cause service breakdown.	80.0%
	RangerAdmin Non-Heap Memory Usage	45279	RangerAdmin Non-Heap Memory Usage Exceeds the Threshold	Non-heap memory overflow may cause service breakdown.	80.0%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	RangerAdmin Heap Memory Usage	45277	RangerAdmin Heap Memory Usage Exceeds the Threshold	Heap memory overflow may cause service breakdown.	95.0%
	RangerAdmin Direct Memory Usage	45278	RangerAdmin Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause service breakdown.	80.0%
	TagSync Direct Memory Usage	45286	TagSync Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause service breakdown.	80.0%
	TagSync Non-Heap Memory Usage	45287	TagSync Non-Heap Memory Usage Exceeds the Threshold	Non-heap memory overflow may cause service breakdown.	80.0%
	TagSync Heap Memory Usage	45285	TagSync Heap Memory Usage Exceeds the Threshold	Heap memory overflow may cause service breakdown.	95.0%
ClickHouse	Clickhouse Service Quantity Quota Usage in ZooKeeper	45426	ClickHouse Service Quantity Quota Usage in ZooKeeper Exceeds the Threshold	After the ZooKeeper quantity quota of the ClickHouse service exceeds the threshold, you cannot perform cluster operations on the ClickHouse service on FusionInsight Manager. As a result, the ClickHouse service cannot be used.	90%

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	ClickHouse Service Capacity Quota Usage in ZooKeeper	45427	ClickHouse Service Capacity Quota Usage in ZooKeeper Exceeds the Threshold	After the ZooKeeper capacity quota of the ClickHouse service exceeds the threshold, you cannot perform cluster operations on the ClickHouse service on FusionInsight Manager. As a result, the ClickHouse service cannot be used.	90%
IoTDB	Maximum Merge (Intra-Space Merge) Latency	45594	IoTDBServer Intra-Space Merge Duration Exceeds the Threshold	Data write is blocked and the write operation performance is affected.	300000 ms
	Maximum Merge (Flush) Latency	45593	IoTDBServer Flush Execution Duration Exceeds the Threshold	Data write is blocked and the write operation performance is affected.	300000 ms
	Maximum Merge (Cross-Space Merge) Latency	45595	IoTDBServer Cross-Space Merge Duration Exceeds the Threshold	Data write is blocked and the write operation performance is affected.	300000 ms
	Maximum RPC (executeStatement) Latency	45592	IoTDBServer RPC Execution Duration Exceeds the Threshold	Running performance of the IoTDBServer process is affected.	10000s
	Total GC Duration of IoTDBServer	45587	IoTDBServer GC Duration Exceeds the Threshold	A long GC duration of the IoTDBServer process may interrupt the services.	12000 ms

Service	Metric	ID	Alarm Name	Impact on System	Default Threshold
	Total GC Duration of ConfigNode	45590	ConfigNode GC Duration Exceeds the Threshold	A long GC duration of the ConfigNode process may interrupt services.	12000 ms
	IoTDBServer Heap Memory Usage	45586	IoTDBServer Heap Memory Usage Exceeds the Threshold	If the available IoTDBServer process heap memory is insufficient, a memory overflow occurs and the service breaks down.	90%
	IoTDBServer Direct Memory Usage	45588	IoTDBServer Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause service breakdown.	90%
	ConfigNode Heap Memory Usage	45589	ConfigNode Heap Memory Usage Exceeds the Threshold	If the available ConfigNode process heap memory is insufficient, a memory overflow occurs and the service breaks down.	90%
	ConfigNode Direct Memory Usage	45591	ConfigNode Direct Memory Usage Exceeds the Threshold	Direct memory overflow may cause the IoTDB instance to be unavailable.	90%

5.2 Submitting Spark Tasks to New Task Nodes

You can add task nodes to an MRS cluster to increase compute capability. Task nodes are mainly used to process data instead of permanently storing data.

This section describes how to bind a new task node using tenant resources and submit Spark tasks to the new task node. You can get started by reading the following topics:

1. **Adding Task Nodes**
2. **Creating a Resource Pool**
3. **Creating a Tenant**
4. **Configuring Queues**
5. **Configuring Resource Distribution Policies**
6. **Creating a User**
7. **Using spark-submit to Submit a Task**
8. **Deleting Task Nodes**

Adding Task Nodes

1. On the cluster details page, click **Nodes** and click **Add Node Group**. The **Add Node Group** page is displayed.
2. On the **Add Node Group** page that is displayed, set parameters as needed.

   Table 5-4 Parameters for adding a node group

Parameter	Description
Instance Specifications	Select the flavor type of the hosts in the node group.
Nodes	Configure the number of nodes in the node group.
System Disk	Configure the specifications and capacity of the system disks on the new nodes.
Data Disk (GB)/Disks	Set the specifications, capacity, and number of data disks of the new nodes.
Deploy Roles	Select **NM** to add a NodeManager role.

3. Click **OK**.

Creating a Resource Pool

Step 1 On the cluster details page, click **Tenants**.

Step 2 Click **Resource Pools**.

Step 3 Click **Create Resource Pool**.

Step 4 On the **Create Resource Pool** page, set the properties of the resource pool.

- **Name**: Enter the name of the resource pool, for example, **test1**.
- **Resource Label**: Enter the resource pool label, for example, **1**.
- **Available Hosts**: Enter the node added in **Adding Task Nodes**.

Step 5 Click **OK**.

----End

Creating a Tenant

Step 1 On the cluster details page, click **Tenants**.

Step 2 Click **Create Tenant**. On the page that is displayed, configure tenant properties. The following table takes MRS 3.x versions as an example.

Table 5-5 Tenant parameters

Parameter	Description
Name	Set the tenant name, for example, **tenant_spark**.
Tenant Type	Select **Leaf**. If **Leaf** is selected, the current tenant is a leaf tenant and no sub-tenant can be added. If **Non-leaf** is selected, sub-tenants can be added to the current tenant.
Compute Resource	If **Yarn** is selected, the system automatically creates a task queue using the tenant name in Yarn. If **Yarn** is not selected, the system does not automatically create a task queue.

Parameter	Description
Configuration Mode	If **Yarn** is selected for **Compute Resource**, this parameter can be set to **Basic** or **Advanced**. ● **Basic**: Configure the percentage of compute resources used by the tenant in the default resource pool by specifying **Default Resource Pool Capacity (%)**. ● **Advanced**: Configure the following parameters for advanced settings: – **Weight**: Tenant resource weight. The value ranges from 0 to 100. Tenant resource weight = Tenant weight/Total weight of tenants at the same level – **Minimum Resources**: resources preempted by the tenant. The value is a percentage or absolute value of the parent tenant's resources. When a tenant's workload is light, their resources are automatically lent to other tenants. When available resources are fewer than **Minimum Resources**, the tenant can preempt the resources that were lent out. – **Maximum Resources**: maximum resources that can be used by a tenant. The value is a percentage or absolute value of the parent tenant's resources. – **Reserved Resources**: resources reserved for the tenant. The value is a percentage or absolute value of the parent tenant's resources.
Default Resource Pool Capacity (%)	Set the percentage of computing resources used by the current tenant in the **default** resource pool, for example, **20%**.
Storage Resource	If HDFS is selected, the system automatically creates the **/tenant** directory under the root directory of the HDFS when a tenant is created for the first time. If **HDFS** is not selected, the system does not create a storage directory under the root directory of the HDFS.
Maximum Number of Files/Directories	Set the maximum number of files or directories, for example, **100000000000**.

Parameter	Description
Storage Space Quota	Quota for the HDFS storage space used by the current tenant The minimum value is **1**, and the maximum value is the total storage quota of the parent tenant. The unit is MB or GB. Set the quota for using the storage space, for example, 50000 MB. This parameter indicates the maximum HDFS storage space that can be used by a tenant, but not the actual space used. If its value is greater than the size of the HDFS physical disk, the maximum space available is the full space of the HDFS physical disk. **NOTE** To ensure data reliability, the system automatically generates one backup file when a file is stored in the HDFS. That is, two replicas of the same file are stored by default. The HDFS storage space indicates the total disk space occupied by all these replicas. For example, if the value is set to **500** MB, the actual space for storing files is about 250 MB (500/2 = 250).
Storage Path	Set the storage path, for example, **tenant/spark_test**. The system automatically creates a folder named after the tenant under the **/tenant** directory by default, for example, **spark_test**. The default HDFS storage directory for tenant **spark_test** is **tenant/spark_test**. When a tenant is created for the first time, the system creates the **/tenant** directory in the HDFS root directory. The storage path is customizable.
Services	Set other service resources associated with the current tenant. HBase is supported. To configure this parameter, click **Associate Services**. In the displayed dialog box, set **Service** to **HBase**. If **Association Mode** is set to **Exclusive**, service resources are occupied exclusively. If **share** is selected, service resources are shared.
Description	Enter the description of the current tenant.

Step 3 Click **OK** to save the settings.

It takes a few minutes to save the settings. If the **Tenant created successfully** is displayed in the upper-right corner, the tenant is added successfully.

NOTE

- Roles, computing resources, and storage resources are automatically created when tenants are created.
- The new role has permissions on the computing and storage resources. The role and its permissions are controlled by the system automatically and cannot be controlled manually under **Manage Role**.
- If you want to use the tenant, create a system user and assign the Manager_tenant role and the role corresponding to the tenant to the user.

----End

Configuring Queues

Step 1 On the cluster details page, click **Tenants**.

Step 2 Click the **Queue Configuration** tab.

Step 3 In the tenant queue table, click **Modify** in the **Operation** column of the specified tenant queue.

NOTE

- In the tenant list on the left of the **Tenant Management** page, click the target tenant.

  In the displayed window, choose **Resource**. On the displayed page, click  to open the queue modification page (for versions earlier than MRS 3.x).
- A queue can be bound to only one non-default resource pool.

By default, the resource tag is the one specified in **Creating a Resource Pool**. Set other parameters based on the site requirements.

Step 4 Click **OK**.

----End

Configuring Resource Distribution Policies

Step 1 On the cluster details page, click **Tenants**.

Step 2 Click **Resource Distribution Policies** and select the resource pool created in **Creating a Resource Pool**.

Step 3 Locate the row that contains **tenant_spark**, and click **Modify** in the **Operation** column.

- **Weight**: **20**
- **Minimum Resource**: **20**
- **Maximum Resource**: **80**
- **Reserved Resource**: **10**

Step 4 Click **OK**.

----End

Creating a User

Step 1 Log in to FusionInsight Manager. For details, see **Accessing FusionInsight Manager**.

Step 2 Choose **System** > **Permission** > **User**. On the displayed page, click **Create User**.

- **Username**: **spark_test**
- **User Type**: **Human-Machine**
- **User Group**: hadoop and hive
- **Primary Group**: **hadoop**
- **Role**: **tenant_spark**

Step 3 Click **OK** to add the user.

----End

Using spark-submit to Submit a Task

1. Log in to the client node as user **root** and run the following commands:

   cd *Client installation directory*

   source bigdata_env

   source Spark2x/component_env

   For a cluster with Kerberos authentication enabled, run the **kinit spark_test** command. For a cluster with Kerberos authentication disabled, skip this step.

   Enter the password for authentication. Change the password upon the first login.

   cd Spark2x/spark/bin

   sh spark-submit --queue tenant_spark --class org.apache.spark.examples.SparkPi --master yarn-client ../examples/jars/spark-examples_*.jar

Deleting Task Nodes

1. On the cluster details page, click **Nodes**.
2. Locate the row that contains the target task node group, and click **Scale In** in the **Operation** column.
3. Set the **Scale-In Type** to **Specific node** and select the target nodes.

   NOTE

   Only nodes in the stopped, lost, unknown, isolated, or faulty state can be selected for scale-in.

4. Select **I understand the consequences of performing the scale-in operation**, and click **OK**.

5.3 Configuring Auto Scaling for an MRS Cluster

In big data application scenarios, especially real-time data analysis and processing, the number of cluster nodes needs to be dynamically adjusted according to data

volume changes to provide proper resources. The auto scaling function of MRS enables clusters to be automatically scaled out or in based on cluster load.

- Auto scaling rules: You can increase or decrease Task nodes based on real-time cluster loads. Auto scaling will be triggered when the data volume changes but there may be some delays.
- Resource plan (setting the task node quantity based on the time range): If the data volume changes periodically, you can create resource plans to resize the cluster before the data volume changes, thereby avoiding delays in increasing or decreasing resources.

You can configure either auto scaling rules or resource plans or both of them to trigger the auto scaling.

Scenario

The following example describes how to use both auto scaling rules and resource plans:

A real-time processing service sees an unstable increase in data volume from 7:00 to 13:00 on Monday, Tuesday, and Saturday. For example, 5 to 8 task nodes are required from 7:00 to 13:00 on Monday, Tuesday, and Saturday, and 2 to 4 are required beyond this period.

You can set an auto scaling rule based on a resource plan. When the data volume exceeds the expected value, the number of Task nodes changes with resource loads, without exceeding the node range specified in the resource plan. When a resource plan is triggered, the number of nodes changes within the specified range with minimum affect. That is, increase nodes to the upper limit and decrease nodes to the lower limit.

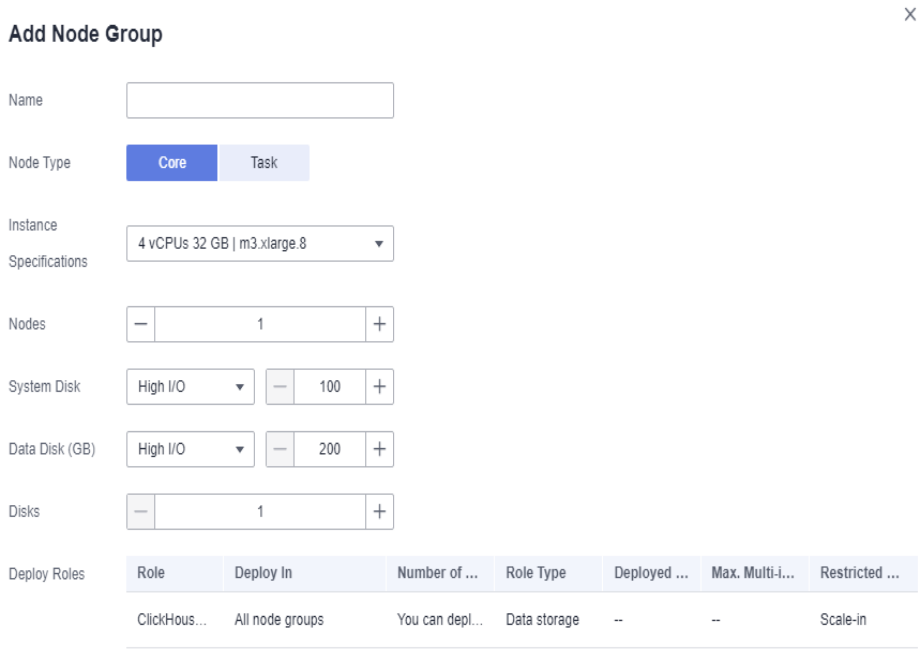
Adding a Task Node

You can scale out an MRS cluster by manually adding task nodes.

To add a task node to a custom cluster, perform the following steps:

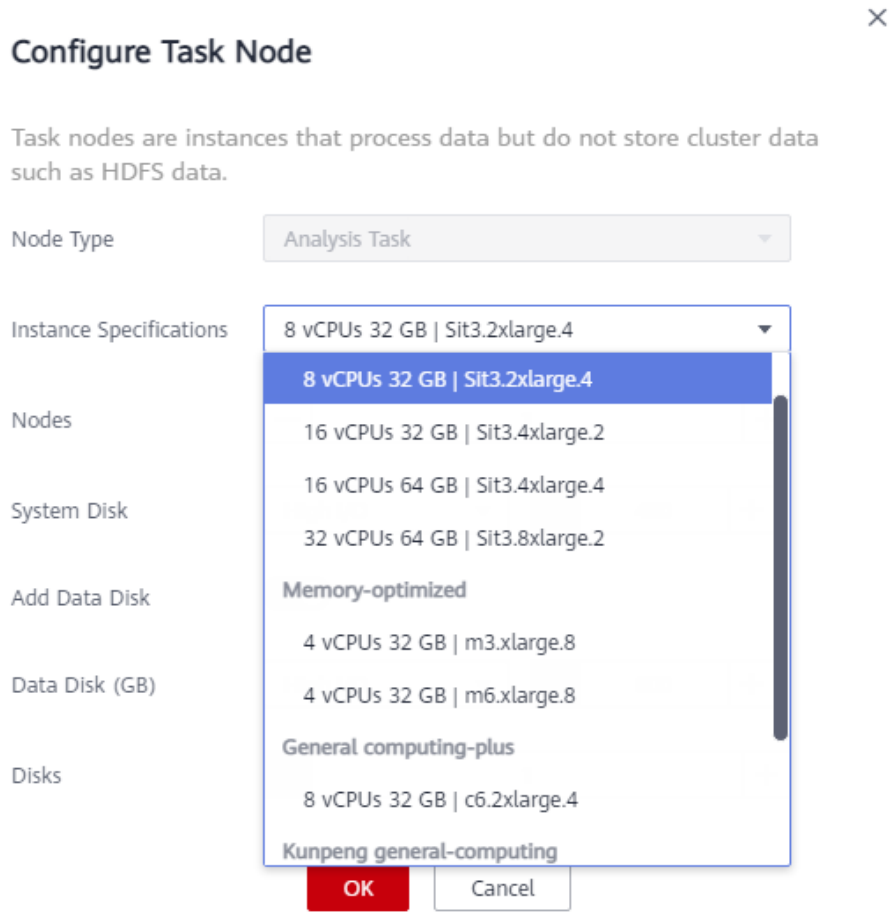
1. On the cluster details page, click the **Nodes** tab and click **Add Node Group**. The **Add Node Group** page is displayed.
2. Select **Task** for**Node Type**. Retain the default value **NM** for **Deploy Roles**. To deploy the NodeManager role, the node type must be **Task**. Set other parameters as required.

Figure 5-3 Adding a task node group



To add a task node to a non-custom cluster, perform the following steps:

1. On the cluster details page, click the **Nodes** tab and click **Configure Task Node**. The **Configure Task Node** page is displayed.
2. On the **Configure Task Node** page, set **Node Type**, **Instance Specifications**, **Nodes**, **System Disk**. In addition, if **Add Data Disk** is enabled, configure the storage type, size, and number of data disks.



3. Click **OK**.

Using Auto Scaling Rules and Resource Plans Together

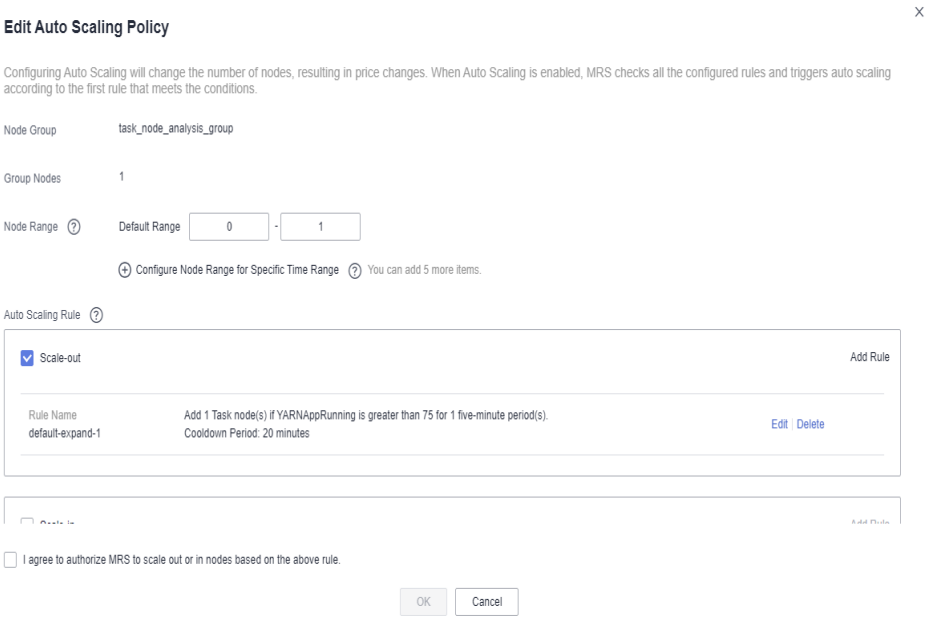
Step 1 Log in to the MRS management console.

Step 2 On the **Active Clusters** page, and click the name of the cluster to be operated. The cluster details page is displayed.

Step 3 On the page that is displayed, click the **Auto Scaling** tab.

Step 4 Click **Add Auto Scaling Policy** and set **Node Range** to **2-4**.

Figure 5-4 Configuring auto scaling



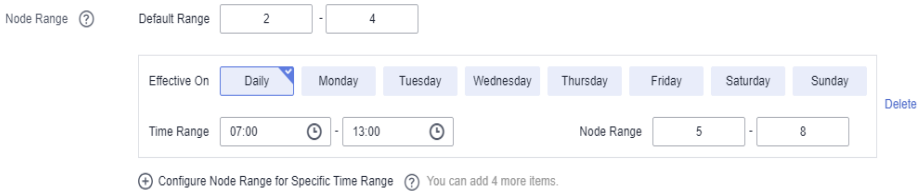
Step 5 Configure a resource plan.

1. Click **Configure Node Range for Specific Time Range** under **Default Range**.
2. Configure the **Time Range** and **Node Range** parameters.

   Time Range: Set it to **07:00-13:00**.

   Node Range: Set it to **5-8**.

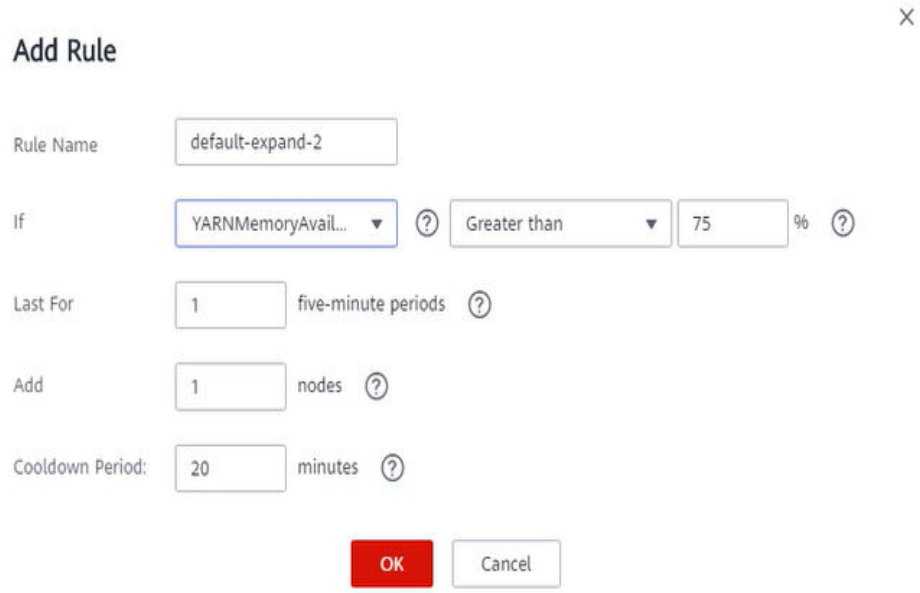
   Figure 5-5 Auto scaling



Step 6 Configure an auto scaling rule.

1. Select **Scale-out**.
2. Click **Add Rule** on the right.

Figure 5-6 Adding a rule



Rule Name: **default-expand-2**.

If: Select the rule objects and constraints from the drop-down list boxes, for example, **YARNAppRunning** is greater than 75.

Last For: Set it to **1 five-minute periods**.

Add: Set it to **1 node**.

Cooldown Period: Set it to **20 minutes**.

3. Click **OK**.

Step 7 Select **I agree to authorize MRS to scale out or in nodes based on the above rule**.

Step 8 Click **OK**.

----End

Reference Information

When adding a rule, you can refer to **Table 5-6** to configure the corresponding metrics.

NOTE

- Hybrid clusters support all metrics of analysis and streaming clusters.
- The accuracy of different value types in **Table 5-6** is as follows:
  - **Integer**: integer
  - **Percentage**: 0.01
  - **Ratio**: 0.01

Table 5-6 Auto scaling metrics

Cluster Type	Metric	Value Type	Description
Streaming cluster	StormSlotAvailable	Integer	Number of available Storm slots. Value range: 0 to 2147483646.
	StormSlotAvailablePercentage	Percentage	Percentage of available Storm slots, that is, the proportion of the available slots to total slots. Value range: 0 to 100.
	StormSlotUsed	Integer	Number of used Storm slots. Value range: 0 to 2147483646.
	StormSlotUsedPercentage	Percentage	Percentage of the used Storm slots, that is, the proportion of the used slots to total slots. Value range: 0 to 100.
	StormSupervisorMemAverageUsage	Integer	Average memory usage of the Supervisor process of Storm. Value range: 0 to 2147483646.
	StormSupervisorMemAverageUsagePercentage	Percentage	Average percentage of the used memory of the Supervisor process of Storm to the total memory of the system. Value range: 0 to 100.
	StormSupervisorCPUAverageUsagePercentage	Percentage	Average percentage of the used CPUs of the Supervisor process of Storm to the total CPUs. Value range: [0, 6000].
Analysis cluster	YARNAppPending	Integer	Number of pending tasks on Yarn. Value range: 0 to 2147483646.
	YARNAppPendingRatio	Ratio	Ratio of pending tasks on YARN, that is, the ratio of pending tasks to running tasks on YARN. Value range: 0 to 2147483646.
	YARNAppRunning	Integer	Number of running tasks on Yarn. Value range: 0 to 2147483646.
	YARNContainerAllocated	Integer	Number of containers allocated to YARN. Value range: 0 to 2147483646.

Cluster Type	Metric	Value Type	Description
	YARNContainerPending	Integer	Number of pending containers on Yarn. Value range: 0 to 2147483646.
	YARNContainerPendingRatio	Ratio	Ratio of pending containers on Yarn, that is, the ratio of pending containers to running containers on Yarn. Value range: 0 to 2147483646.
	YARNCPUAllocated	Integer	Number of virtual CPUs (vCPUs) allocated to Yarn. Value range: 0 to 2147483646.
	YARNCPUAvailable	Integer	Number of available vCPUs on Yarn. Value range: 0 to 2147483646.
	YARNCPUAvailablePercentage	Percentage	Percentage of available vCPUs on Yarn, that is, the proportion of available vCPUs to total vCPUs. Value range: 0 to 100.
	YARNCPUPending	Integer	Number of pending vCPUs on Yarn. Value range: 0 to 2147483646.
	YARNMemoryAllocated	Integer	Memory allocated to Yarn. The unit is MB. Value range: 0 to 2147483646.
	YARNMemoryAvailable	Integer	Available memory on Yarn. The unit is MB. Value range: 0 to 2147483646.
	YARNMemoryAvailablePercentage	Percentage	Percentage of available memory on Yarn, that is, the proportion of available memory to total memory on Yarn. Value range: 0 to 100.
	YARNMemoryPending	Integer	Pending memory on Yarn. Value range: 0 to 2147483646.

When adding a resource plan, you can set parameters by referring to **Table 5-7**.

Table 5-7 Configuration items of a resource plan

Parameter	Description
Effective On	The effective date of a resource plan. **Daily** is selected by default. You can also select one or multiple days from Monday to Sunday.
Time Range	Start time and end time of a resource plan are accurate to minutes, with the value ranging from **00:00** to **23:59**. For example, if a resource plan starts at 8:00 and ends at 10:00, set this parameter to **8:00-10:00**. The end time must be at least 30 minutes later than the start time. Time ranges configured for different resource plans cannot overlap.
Node Range	The number of nodes in a resource plan ranges from **0** to **500**. In the time range specified in the resource plan, if the number of task nodes is less than the specified minimum number of nodes, it will be increased to the specified minimum value of the node range at a time. If the number of task nodes is greater than the maximum number of nodes specified in the resource plan, the auto scaling function reduces the number of task nodes to the maximum value of the node range at a time. The minimum number of nodes must be less than or equal to the maximum number of nodes.

