088_DLI Delta SQL 语法参考

文档版本 01
发布日期 2025-01-09

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

1 DLI Delta 表概述

Delta表是一种基于Delta Lake技术实现的数据存储解决方案，它使用基于文件的事务日志扩展了 Parquet 数据文件，可以处理 ACID 事务和可缩放的元数据。 Delta Lake与Apache Spark API完全兼容，并且其设计能够与结构化流式处理紧密集成，可以轻松地将单个数据副本用于批处理和流式处理操作，并提供大规模增量处理。

DLI 中 Delta 的使用限制

- 仅Spark 3.3.1（3.0.0）及以上版本支持Delta。
- DLI支持的Delta版本是Delta 2.3.0。
- Spark 3.3.1（3.0.0）版本的SQL仍有部分不支持Delta表的相关开源语法，具体内容请参考**表1-1**。

表 1-1 Spark 3.3.1-3.0.0 版本 SQL 不支持的 Delta 表相关开源语法

不支持的语法	示例
ALTER TABLE REPLACE COLUMNS替换列	alter table table0 replace columns(id1 int,name1 string);
SHOW CREATE TABLE显示建表语句	show create table table1;
INSERT INTO/OVERWRITE指定静态分区插入表	insert into table1 partition(part='part1') select * from table2;
ALTER TABLE ADD/DROP PARTITION管理分区	alter table test_delta_parts1 add partition('2024-10-28');
CONVERT TO DELTA语法暂不支持parquet.`tablePath`格式的表	convert to delta parquet.`obs://bucket0/db0/table0`;

2 DLI 中使用 Delta 开发作业

2.1 DLI Delta 元数据

- 在DLI中提交Spark SQL作业开发Delta的SQL语法请参考**Delta SQL语法参考**。
- 在DLI中提交Spark Jar作业开发Delta请参考**在DLI使用Delta提交Spark Jar作业**。

DLI Delta 元数据说明

创建Delta表时会在元数据仓创建表的相关元数据信息。

Delta支持对接DLI元数据和Lakeformation元数据（仅Spark 3.3.1及以上版本支持对接Lakeformation元数据），对接方式与Spark一致。

- DLI元数据可在数据湖探索管理控制台的“数据管理 > 库表管理”中查看。
- Lakeformation元数据可在湖仓构建Lakeformation服务的管理控制台中查看。

相关操作

- DLI SQL队列对接DLI元数据方法：
  a. 在DLI管理控制台的SQL编辑器页面的“数据目录”中选择“dli”。
  b. 在“数据库”选项中选择要对接的DLI元数据中的数据库，即可对接到DLI元数据。
- DLI通用队列对接DLI元数据方法：
  请参考**使用Spark作业访问DLI元数据**。
- DLI SQL队列对接Lakeformation元数据方法：
  参考**DLI对接LakeFormation**。
- DLI通用队列对接Lakeformation元数据方法：
  参考**DLI对接LakeFormation**。
- DLI元数据权限管理
  可通过DLI SQL权限管理或者IAM鉴权管理DLI元数据的权限
  - DLI SQL权限管理：
    i. 在“数据湖探索 > 数据管理 > 库表管理”页面，搜索要授权的库/表。

ii. 单击表操作列的“权限管理”，即可查看当前库/表授权信息或者新增授权。

更多信息请参考**在DLI控制台管理数据库资源**。

- IAM鉴权：

参考**权限管理概述**章节中的“IAM鉴权使用场景”。

- Lakeformation元数据权限管理

参考**DLI对接LakeFormation**。

2.2 在 DLI 使用 Delta 提交 Spark Jar 作业

1. 添加如下依赖

```
<dependency>
   <groupId>io.delta</groupId>
   <artifactId>delta-core_2.12</artifactId>
   <version>2.3.0</version>
</dependency>
```

2. SparkSession添加下面两个参数

```
.config("spark.sql.extensions", "io.delta.sql.DeltaSparkSessionExtension")
.config("spark.sql.catalog.spark_catalog", "org.apache.spark.sql.delta.catalog.DeltaCatalog")
```

3. 编写代码（可通过sql或者api两种方式实现）。

1）sql开发示例如下，具体sql语法参考“Delta Sql语法参考”。

```
public static void main( String[] args )
{
   SparkSession spark = SparkSession
         .builder()
         .enableHiveSupport()
         .config("spark.sql.catalog.spark_catalog", "org.apache.spark.sql.delta.catalog.DeltaCatalog")
         .config("spark.sql.extensions", "io.delta.sql.DeltaSparkSessionExtension")
         .appName("DeltaDemo")
         .getOrCreate();
   String sql_create = "CREATE TABLE if not exists dligms.deltaTest1012 (\n" +
         "  id int,\n" +
         "  name string,\n" +
         "  start_year int,\n" +
         "  class string\n" +
         ") USING DELTA\n" +
         " partitioned by(start_year, class)\n" +
         " location 'obs://bucket_name/path/deltaTest1012'";
   spark.sql(sql_create);
   String sql_insert = "insert into dligms.deltaTest1012 values\n" +
         "(1, 'zhangsan', 2024, 'whlg0905')," +
         "(2, 'lisi', 2024, 'whlg0905')," +
         "(3, 'wangwu', 2024, 'whlg0905')," +
         "(4, 'zhaoliu', 2024, 'whlg0905')";
   spark.sql(sql_insert);
   String sql_select = "select * from dligms.deltaTest1012";
   spark.sql(sql_select).show();
   spark.stop();
}
```

2）api开发示例如下（java）。

```
public static void main( String[] args )
{
   SparkSession spark = SparkSession
         .builder()
         .enableHiveSupport()
```

088_DLI Delta SQL 语法参考


```
            .config("spark.sql.catalog.spark_catalog", "org.apache.spark.sql.delta.catalog.DeltaCatalog")
            .config("spark.sql.extensions", "io.delta.sql.DeltaSparkSessionExtension")
            .appName("DeltaDemo")
            .getOrCreate();
    DeltaTable.createIfNotExists(spark)
            .tableName("deltaJava1011")
            .addColumn("id", "INT")
            .addColumn("name", "STRING")
            .addColumn("start_year", "INT")
            .addColumn("class", "STRING")
            .partitionedBy("start_year", "class")
            .location("obs://bucket_name/path/deltaTest1011")
            .execute();
    Dataset<Row> data = spark.read().format("csv")
            .option("header", "true")
            .option("inferSchema", "true")
            .load("obs://bucket_name/path/export/test1011/");
    data.write().insertInto("deltaJava1011");
    spark.stop();
}
```

4. 编译打包后，并执行作业。

3 Delta Time Travel

3.1 查看 Delta 表历史操作记录

命令格式

DESCRIBE HISTORY *[database_name.]table_name|DELTA.`obs_path`* [LIMIT n]

示例

```
DESCRIBE HISTORY delta_table0;
DESCRIBE HISTORY delta.`obs://tablePath` LIMIT 1;
```

系统响应

返回表的历史操作记录，结果指标代表含义见下表。

表 3-1 结果指标说明

指标名称	指标含义
version	对表操作的版本号。
timestamp	当前版本操作的时间戳。
userId	当前版本操作的用户ID。
userName	当前版本操作的用户名。
operation	操作名称（WRITE|CREATE TABLE|UPDATE|DELETE|MERGE|RESTORE等）。
operationParameters	操作参数。
job	运行该操作的作业的详细信息。
notebook	运行操作的笔记的详细信息。
clusterId	集群id。

指标名称	指标含义
readVersion	为执行写操作而读取的表的版本。
isolationLevel	隔离级别。
isBlindAppend	是否追加数据。
operationMetrics	操作的度量（例如，修改的文件数、行数、字节数等信息）。
engineInfo	Spark和Delta版本信息。

3.2 查询 Delta 表历史版本数据

命令格式

查询Delta表历史某一时刻的状态：

SELECT * FROM *[database_name.]table_name*

TIMESTAMP AS OF timestamp_expression

查询Delta表某一历史版本的状态：

SELECT * FROM *[database_name.]table_name* VERSION AS OF version_code

参数描述

表 3-2 查询 Delta 表历史版本参数说明

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
timestamp_expression	时间戳，不能晚于当前时间，格式'yyyy-MM-ddTHH:mm:ss.SSS'。
version_code	**查看Delta表历史操作记录**中查询结果中的版本号。

示例

```
SELECT * FROM delta_table0 TIMESTAMP AS OF '2020-10-18T22:15:12.013Z';
SELECT * FROM delta_table0 VERSION AS OF 2 where part_col='part_value';
```

3.3 还原 Delta 表到早期状态

命令格式

还原Delta表到历史某一时刻的状态：

RESTORE [TABLE] [database_name.]table_name/DELTA.`obs_path`

[TO] TIMESTAMP AS OF timestamp_expression

还原Delta表到某一历史版本的状态：

RESTORE [TABLE] [database_name.]table_name/DELTA.`obs_path`

[TO] VERSION AS OF version_code

参数描述

表 3-3 还原 Delta 表版本参数说明

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
obs_path	Obs路径，表示Delta表的存储位置。
timestamp_expression	时间戳，不能晚于当前时间，格式'yyyy-MM-ddTHH:mm:ss.SSS'
version_code	**查看Delta表历史操作记录**中查询结果对应的版本号。

示例

```
RESTORE delta_table0 TO TIMESTAMP AS OF '2020-10-18T22:15:12.013Z';
RESTORE delta.`obs://bucket_name/db0/delta_table0` VERSION AS OF 2;
```


4 Delta 清理和优化

清理 Delta 表

可以对 Delta 表运行 VACUUM 命令，以删除该表中不再引用且在保留期阈值之前创建的数据文件。

```
VACUUM delta_table0;
VACUUM delta_table0 RETAIN 168 HOURS;--单位只支持HOURS
```

优化 Delta 表

为了提高查询速度，Delta Lake支持优化数据在存储中的布局，这会将许多较小的文件压缩为较大的文件。

```
optimize delta_table0;
optimize delta_table0 where date >= '2020-01-01';
```

Z 排序

Zordering是另一种加快查询速度的技术。对数据进行Z排序可以重新组织存储中的数据，当您的数据被适当地排序时，可以跳过更多的文件，读取更少的数据，从而运行得更快。要对Z-Order数据进行排序，请在ZORDER BY中指定要对其进行排序的列。

```
OPTIMIZE delta_table0 ZORDER BY (price);
```

5 Delta SQL 语法参考

5.1 Delta DDL 语法说明

5.1.1 CREATE TABLE

命令功能

CREATE TABLE命令通过指定带有表属性的字段列表来创建Delta Table。

注意事项

- 在该命令中，IF EXISTS和db_name是可选配置。
- 在DLI中delta只支持OBS外表，通过表名创建且未指定location时将会失败。

命令格式

- 通过表名创建Delta表

  CREATE[OR REPLACE] TABLE *[IF NOT EXISTS] [database_name.]table_name*

  [(columnTypeList)]

  USING DELTA

  [COMMENT table_comment]

  [PARTITIONED BY (partColumnList)]

  LOCATION location_path

- 通过delta.`Obs路径`创建Delta表

  CREATE[OR REPLACE] TABLE [IF NOT EXISTS] DELTA. `obs://*bucket_name/tbl_path*`

  [(columnTypeList)]

  USING DELTA

  [COMMENT table_comment]

  [PARTITIONED BY (partColumnList)]

说明

通过表名创建，能通过show tables查到该表，当前版本必须制定location，且只能指定为obs路径；通过delta.`Obs路径`创建，不能通过show tables查到。

参数描述

表 5-1 CREATE TABLE 参数描述

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
columnTypeList	以逗号分隔的带数据类型的列表。列名由字母、数字和下划线（_）组成。
using	参数delta，定义和创建Delta table。
table_comment	表的描述信息。
partColumnList	分区字段列表，列名取自columnTypeList中的列名。
location_path	Delta表的存储位置，当前版本必须指定，且只支持obs路径，指定该路径Delta 表会创建为外表。

所需权限

- SQL权限

  表 5-2 CREATE TABLE 所需权限列表

权限描述
数据库的CREATE_TABLE权限

- 细粒度权限：dli:database:createTable
- LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

- **通过表名创建非分区表**

```
create table if not exists delta_table0 (
id int,
name string,
addr struct<priv:string,city:string>,
price double
) using delta
location 'obs://bucket_name0/db0/delta_table0';
```


- **通过表名创建分区表**

```
create table if not exists delta_table0 (
id bigint,
name string,
ts bigint,
dt string,
hh string
) using delta
partitioned by (dt, hh)
location 'obs://bucket_name0/db0/delta_table0';
```

- **通过delta路径创建非分区表**

```
create table if not exists delta.`obs://bucket_name0/db0/delta_table0`(
id int,
name string,
price double
) using delta;
```

- **通过delta路径创建分区表**

```
create table if not exists delta.`obs://bucket_name0/db0/delta_table0` (
id bigint,
name string,
ts bigint,
dt string,
hh string
) using delta
partitioned by (dt, hh);
```

- **通过create table like创建表**

```
create table delta_table2 like delta_table1
using delta
location 'obs://bucket_name0/db0/delta_table2';
```

系统响应

Table创建成功。

5.1.2 DROP TABLE

命令功能

DROP TABLE的功能是用来删除已存在的Table。

命令格式

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

参数描述

表 5-3 DROP TABLE 参数描述

参数	描述
db_name	Database名称。如果未指定，将选择当前database。
table_name	需要删除的Table名称。

所需权限

- SQL权限

  表 5-4 DROP TABLE 所需权限列表

权限描述
表所在数据库的DROP_TABLE权限

- 细粒度权限：dli:table:dropTable。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

DROP TABLE IF EXISTS db0.delta_table0;

系统响应

执行成功，元数据中表将被删除，无法通过show和describe查询该表。

5.1.3 DESCRIBE

命令功能

DESCRIBE命令用于显示表的详细信息或统计信息。

命令格式

显示表统计信息：

DESCRIBE *[EXTENDED|FORMATTED] [database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`;*

显示表详细信息：

DESCRIBE *DETAIL [database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`;*

所需权限

- SQL权限

  表 5-5 DESCRIBE 所需权限列表

权限描述
表的DESCRIBE_TABLE权限

- 细粒度权限：dli:table:describeTable。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
DESCRIBE FORMATTED delta_table0;

DESCRIBE FORMATTED delta.`obs://bucket_name0/db0/delta_table0`;

DESCRIBE DETAIL delta_table0;
```

系统响应

返回表的详细信息或统计信息。

表 5-6 结果参数描述

参数名	参数含义
format	表的格式，在这里是delta
id	表的唯一id
name	在metaserver中定义的表名
description	关于表的说明
location	表的存储路径
createdAt	建表时间戳
lastModified	最后一次修改的时间戳
partitionColumns	分区列
numFiles	表的最新版本中的文件个数
sizeInBytes	表的最新快照的大小（以字节为单位）
properties	为此表设置的所有属性
minReaderVersion	可以读取该表的最低Reader版本
minWriterVersion	可以写入该表的最低Writer版本

5.1.4 ADD CONSTRAINT

命令功能

ADD CONSTRAINT 命令添加 CHECK 约束。 在将约束添加到表中之前会验证所有现有行是否满足约束。

注意事项

在将约束添加到表中之前会验证所有现有行是否满足约束，如果有行不满足约束，约束将添加失败，添加前需先清理不满足约束的数据。

命令格式

ALTER TABLE *[database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`*

ADD CONSTRAINT constraint_name

CHECK(*boolExpression*)

参数描述

表 5-7 ADD CONSTRAINT 参数描述

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
constraint_name	约束名称。
boolExpression	约束条件表达式。

所需权限

- SQL权限

表 5-8 ADD CONSTRAINT 所需权限列表

权限描述
表的ALTER权限

- 细粒度权限：dli:table:alter。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
alter table delta_table0 add constraint const_price check(price>0);
alter table delta.`obs://bucket1/dbgms/h0` add constraint const_id check(id>0);
```

系统响应

可在执行历史或作业列表中查看任务运行成功或失败。

5.1.5 DROP CONSTRAINT

命令功能

DROP CONSTRAINT 命令删除 CHECK 约束。

命令格式

ALTER TABLE *[database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`*

DROP CONSTRAINT constraint_name

参数描述

表 5-9 DROP CONSTRAINT 参数描述

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
constraint_name	约束名称。

所需权限

- SQL权限

表 5-10 DROP CONSTRAINT 所需权限列表

权限描述
表的ALTER权限

- 细粒度权限：dli:table:alter。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
alter table delta_table0 drop constraint const_price;

alter table delta.`obs://bucket1/dbgms/h0` drop constraint const_id;
```

系统响应

可在执行历史或作业列表中查看任务运行成功或失败。

5.1.6 CONVERT TO DELTA

命令功能

CONVERT TO DELTA 命令将现有的 Parquet 表就地转换为 Delta 表。 此命令会列出目录中的所有文件，创建 Delta Lake 事务日志来跟踪这些文件，并通过读取所有 Parquet 文件的页脚来自动推断数据架构。 转换过程会收集统计信息，以提升转换后的 Delta 表的查询性能。 如果提供表名，则元存储也将更新，以反映该表现在是 Delta 表。

注意事项

分区表转换需要设置参数
spark.sql.forcePartitionPredicatesOnPartitionedTable.enabled为false。

命令格式

CONVERT TO DELTA *[database_name.]table_name* [NO STATISTICS]

参数描述

表 5-11 CONVERT TO DELTA 参数描述

参数	描述
database_name	Database名称，由字母、数字和下划线（ _ ）组成。
table_name	Database中的表名，由字母、数字和下划线（ _ ）组成。
NO STATISTICS	表示在转换过程中绕过统计信息收集，以更快的速度完成转换。

所需权限

- SQL权限

表 5-12 CONVERT TO DELTA 所需权限列表

权限描述
表所在数据库的CREATE_TABLE权限
表的ALTER权限
表的INSERT_INTO_TABLE权限
表的DROP_TABLE权限

- 细粒度权限：dli:database:createTable, dli:table:alter, dli:table:insertIntoTable, dli:table:dropTable。

- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
create table if not exists parquet_table0 (id int,name string,price double) using parquet location 'obs://
bucket_name0/db0/parquet_table0';
convert to delta parquet_table0;
```

系统响应

执行成功。

5.1.7 SHALLOW CLONE

命令功能

SHALLOW CLONE 命令在特定版本创建现有Delta表的浅拷贝。被克隆的信息包括：schema、分区信息、数据文件路径等。

对克隆表所做的任何更改都只会影响克隆本身，而不会影响源表，只要它们不触及源数据。注意克隆表可能仍会指向源表的数据文件，当源表做了vacuum操作时，可能导致克隆表找不到文件。

命令格式

CREATE TABLE *[target_db.]target_table*

SHALLOW CLONE *[source_db.]source_table*|*DELTA.`obs://bucket_name/tbl_path`*

location 'obs://bucket_name/tbl_path'

参数描述

表 5-13 SHALLOW CLONE 参数描述

参数	描述
target_db	目标Database名称，由字母、数字和下划线（_）组成。
target_table	目标表名，由字母、数字和下划线（_）组成。
source_db	源Database名称，由字母、数字和下划线（_）组成。
source_table	源表名，由字母、数字和下划线（_）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
constraint_name	约束名称。
boolExpression	约束条件表达式。

所需权限

- SQL权限

表 5-14 SHALLOW CLONE 所需权限列表

权限描述
目标库的CREATE_TABLE权限
源表的SELECT权限

- 细粒度权限：dli:database:createTable, dli:table:select。
- LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
CREATE OR REPLACE TABLE delta_table1 SHALLOW CLONE delta_table0 LOCATION 'obs://bucket0/db0/
delta_table1';
CREATE TABLE delta.`obs://bucket0/db0/delta_table1` SHALLOW CLONE delta_table0 VERSION AS OF 10;
```

系统响应

可在执行历史或作业列表中查看任务运行成功或失败。

5.2 Delta DML 语法说明

5.2.1 INSERT

命令功能

INSERT命令用于将SELECT查询结果加载到Delta表中。

命令格式

追加模式：

INSERT INTO *[database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`*

select query;

覆盖模式：

INSERT OVERWRITE *[database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`*

select query;

参数描述

表 5-15 INSERT INTO 参数

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
select query	查询语句。

所需权限

- SQL权限

表 5-16 INSERT INTO 所需权限列表

权限描述
表的INSERT_INTO_TABLE权限

- 细粒度权限：dli:table:insertIntoTable
- LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
insert into delta_table0 values(1, 'a1', 20);

insert into delta_table0 select 1, 'a1', 20;

insert into test_delta_parts1 PARTITION(dt) select id,name,dt from test_delta1;

-- insert overwrite table
insert overwrite table delta_table0 select 1, 'a1', 20;
```

系统响应

可在执行历史或作业列表中查看任务运行成功或失败。

5.2.2 CREATE TABLE AS SELECT

命令功能

CREATE TABLE As SELECT命令通过指定带有表属性的字段列表来创建Hudi Table。

命令格式

CREATE[OR REPLACE] TABLE *[IF NOT EXISTS] [database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`*

USING DELTA

[COMMENT table_comment]

[PARTITIONED BY (partColumnList)]

[LOCATION location_path]

[AS query_statement]

参数描述

表 5-17 CREATE TABLE AS SELECT 参数描述

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
using	参数delta，定义和创建Delta table
table_comment	表的描述信息。
location_path	Delta表的存储位置，当前版本通过表名创建Delta表时必须指定，且只支持obs路径，指定该路径Delta 表会创建为外表。
query_statement	select查询表达式

所需权限

- SQL权限

  表 5-18 CREATE TABLE AS SELECT 所需权限列表

权限描述
数据库的CREATE_TABLE权限
查询表的SELECT权限

- 细粒度权限：dli:database:createTable, dli:table:select。

- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

- 创建分区表

```
create table if not exists delta_table0
using delta
partitioned by (dt)
location 'obs://bucket_name0/db0/delta_table0'
as
select 1 as id, 'a1' as name, 10 as price, 1000 as dt;
```

- 创建非分区表

```
create table if not exists delta_table0
using delta
location 'obs://bucket_name0/db0/delta_table0'
as
select 1 as id, 'a1' as name, 10 as price;

create table delta.`obs://bucket_name0/db0/delta_table0`
using delta
partitioned by (part_col1, part_col2)
as select id,name,year,class_name from table1 where part_col1=2024;
```

5.2.3 MERGE INTO

命令功能

通过MERGE INTO命令，根据一张表或子查询的连接条件对另外一张表进行查询，连接条件匹配上的进行UPDATE或DELETE，无法匹配的执行INSERT。这个语法仅需要一次全表扫描就完成了全部同步工作，执行效率要高于INSERT + UPDATE。

注意事项

分区表合并需要设置参数
spark.sql.forcePartitionPredicatesOnPartitionedTable.enabled为false。

命令格式

MERGE INTO *[database_name.]table_name AS target_alias*

USING *(sub_query | tableIdentifier) AS source_alias*

ON *<merge_condition>*

[WHEN MATCHED [AND <condition>] THEN <matched_action>]

[WHEN MATCHED [AND <condition>] THEN <matched_action>]

[WHEN NOT MATCHED [AND <condition>] THEN <not_matched_action>]

<merge_condition> =A equal bool condition

<matched_action> =

DELETE |

*UPDATE SET * |*

UPDATE SET column1 = expression1 [, column2 = expression2 ...]

<not_matched_action> =

*INSERT * |*

INSERT (column1 [, column2 ...]) VALUES (value1 [, value2 ...])

参数描述

表 5-19 MERGE INTO 参数

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
target_alias	目标表的别名。
sub_query	子查询。
source_alias	源表或源表达式的别名。
merge_condition	将源表或表达式和目标表关联起来的条件
condition	过滤条件，可选。
matched_action	当满足条件时进行Delete或Update操作
not_matched_action	当不满足条件时进行Insert操作

所需权限

- SQL权限

  表 5-20 MERGE INTO 所需权限列表

权限描述
表的UPDATE权限
表的INSERT_INTO_TABLE权限
表的DELETE权限

- 细粒度权限：dli:table:update, dli:table:insertIntoTable, dli:table:delete。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

- 部分字段更新

```
create table h0(id int, comb int, name string, price int) using delta location 'obs://bucket_name0/db0/h0';
create table s0(id int, comb int, name string, price int) using delta location 'obs://bucket_name0/db0/s0';
insert into h0 values(1, 1, 'h1', 1);
insert into s0 values(1, 1, 's1', 1),(2, 2, 's2', 2);
//写法1
merge into h0 using s0
on h0.id = s0.id
when matched then update set h0.id = s0.id, h0.comb = s0.comb, price = s0.price * 2;
//写法2
merge into h0 using s0
on h0.id = s0.id
when matched then update set id = s0.id,
name = s0.name,
comb = s0.comb + h0.comb,
price = s0.price + h0.price;
```

- 缺省字段更新和插入

```
create table h0(id int, comb int, name string, price int, flag boolean) using delta location 'obs://bucket_name0/db0/h0';
create table s0(id int, comb int, name string, price int, flag boolean) using delta location 'obs://bucket_name0/db0/s0';
insert into h0 values(1, 1, 'h1', 1, false);
insert into s0 values(1, 2, 's1', 1, true);
insert into s0 values(2, 2, 's2', 2, false);

merge into h0 as target
using (
select id, comb, name, price, flag from s0
) source
on target.id = source.id
when matched then update set *
when not matched then insert *;
```

- 多条件更新和删除

```
create table h0(id int, comb int, name string, price int, flag boolean) using delta location 'obs://bucket_name0/db0/h0';
create table s0(id int, comb int, name string, price int, flag boolean) using delta location 'obs://bucket_name0/db0/s0';
insert into h0 values(1, 1, 'h1', 1, false);
insert into h0 values(2, 2, 'h2', 1, false);
insert into s0 values(1, 1, 's1', 1, true);
insert into s0 values(2, 2, 's2', 2, false);
insert into s0 values(3, 3, 's3', 3, false);

merge into h0
using (
select id, comb, name, price, flag from s0
) source
on h0.id = source.id
when matched and h0.flag = false then update set id = source.id, comb = h0.comb + source.comb, price = source.price * 2
when matched and h0.flag = true then delete
when not matched then insert *;
```

系统响应

可在driver日志和客户端中查看命令运行成功或失败。

5.2.4 UPDATE

命令功能

UPDATE命令根据列表达式和可选的过滤条件更新Delta表。

命令格式

UPDATE *[database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`*

SET column = EXPRESSION(,column = EXPRESSION)

[WHERE boolExpression]

参数描述

表 5-21 UPDATE 参数

参数	描述
database_name	Database名称，由字母、数字和下划线（ _ ）组成。
table_name	Database中的表名，由字母、数字和下划线（ _ ）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
column	待更新的目标列。
EXPRESSION	需在目标表中更新的源表列值的表达式。
boolExpression	过滤条件表达式。

所需权限

- SQL权限

表 5-22 UPDATE 所需权限列表

权限描述
表的UPDATE权限

- 细粒度权限：dli:table:update。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
update delta_table0 set price = price + 20 where id = 1;

update delta.`obs://bucket0/db0/delta_table1` set price = price *2, name = 'a2' where part0='xx' and id = 2;
```

系统响应

可在driver日志和客户端中查看命令运行成功或失败。

5.2.5 DELETE

命令功能

DELETE命令从Delta表中删除记录。

命令格式

DELETE from *[database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`*

[WHERE boolExpression]

参数描述

表 5-23 DELETE 参数

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
boolExpression	删除项的过滤条件

所需权限

- SQL权限

表 5-24 DELETE 所需权限列表

权限描述
表的DELETE权限

- 细粒度权限：dli:table:delete。

- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
delete from delta_table0 where column1 = 'value1';

delete from delta_table0 where column1 IN ('value1', 'value2');

delete from delta.`obs://bucket_name0/db0/delta_table0` where column1 = 'value1';
```

系统响应

可在driver日志和客户端中查看命令运行成功或失败。

5.2.6 VACUUM

命令功能

VACUUM命令用于删除表目录中不由 Delta 管理的所有文件，并删除不再处于表事务日志最新状态且超过保留期阈值的数据文件。默认阈值为 7 天。

注意事项

RETAIN num HOURS表示保留期阈值，建议设置为至少 7 天。

如果对 Delta 表运行VACUUM，则将无法再回头查看在指定数据保留期之前创建的版本。

Delta Lake 具有一项安全检查，用于防止运行危险的VACUUM命令，当指定保留期阈值少于168小时时会报错限制该操作。 如果确定指定保留期阈值进行vacuum操作，可通过将 Spark 配置属性 spark.databricks.delta.retentionDurationCheck.enabled 设置为 false 来关闭此安全检查。

命令格式

VACUUM *[database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`* [RETAIN num HOURS];

支持通过DRY RUN参数模拟执行vacuum操作，返回vacuum将要删除的文件列表：

VACUUM *[database_name.]table_name|DELTA.`obs://bucket_name/tbl_path`* [RETAIN num HOURS] [DRY RUN];

参数描述

表 5-25 VACUUM 参数

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。

参数	描述
bucket_name	obs桶名称。
tbl_path	Delta表在obs桶中的存储位置。
num	保留期时长

所需权限

- SQL权限

表 5-26 VACUUM 所需权限列表

权限描述
表的UPDATE权限

- 细粒度权限：dli:table:update。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
VACUUM delta_table0 RETAIN 168 HOURS;

VACUUM delta_table0 RETAIN 48 HOURS DRY RUN;

VACUUM delta.`obs://bucket_name0/db0/delta_table0` RETAIN 168 HOURS;
```

系统响应

可在driver日志和客户端中查看命令运行成功或失败。

5.2.7 RESTORE

命令功能

RESTORE命令用于将Delta表还原到早期状态，支持还原到较早的版本号或者时间戳。

命令格式

还原Delta表到历史某一时刻的状态：

RESTORE [TABLE] [database_name.]table_name|DELTA.`obs_path`

[TO] TIMESTAMP AS OF timestamp_expression

还原Delta表到某一历史版本的状态：

RESTORE [TABLE] [database_name.]table_name|DELTA.`obs_path`

[TO] VERSION AS OF version_code

参数描述

表 5-27 参数描述

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
obs_path	Obs路径，表示Delta表的存储位置。
timestamp_expression	时间戳，不能晚于当前时间，格式'yyyy-MM-ddTHH:mm:ss.SSS'
version_code	1.3.1中查询结果中的版本号

所需权限

- SQL权限

表 5-28 RESTORE 所需权限列表

权限描述
表的UPDATE权限

- 细粒度权限：dli:table:update。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
RESTORE delta_table0 TO TIMESTAMP AS OF '2020-10-18 22:15:12.013';
RESTORE delta.`obs://bucket_name/db0/delta_table0` VERSION AS OF 2;
```

系统响应

可在driver日志和客户端中查看命令运行成功或失败。

5.2.8 OPTIMIZE

命令功能

OPTIMIZE命令用于优化数据在存储中的布局，提高查询速度。

注意事项

- 由于Optimize是一项耗时的活动，因此需要根据更好的最终用户查询性能与优化计算时间之间的权衡来确定运行Optimize的频率。

- 分区表优化需要设置参数 spark.sql.forcePartitionPredicatesOnPartitionedTable.enabled为false。

命令格式

OPTIMIZE [database_name.]table_name

[WHERE boolExpression]

Z排序：

OPTIMIZE [database_name.]table_name

[WHERE boolExpression]

ZORDER BY (*columnList*);

参数描述

表 5-29 参数描述

参数	描述
database_name	Database名称，由字母、数字和下划线（_）组成。
table_name	Database中的表名，由字母、数字和下划线（_）组成。
boolExpression	过滤条件表达式。
columnList	z排序指定的字段列表，Z顺序列应与分区列不同。

所需权限

- SQL权限

表 5-30 OPTIMIZE 所需权限列表

权限描述
表的UPDATE权限

- 细粒度权限：dli:table:update。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
OPTIMIZE delta_table0;

optimize delta_table0 where dt >= '2020-01-01';
```



```
OPTIMIZE delta_table0 WHERE dt >= current_timestamp() - INTERVAL 1 day ZORDER BY (price);
```

系统响应

可在driver日志和客户端中查看命令运行成功或失败。

5.3 Schema 演进语法说明

功能介绍

该能力用于支持SparkSql对Hudi表的列进行Alter变更，使用该能力前必须开启Schema演进。

Schema 演进支持的范围

Schema演进支持范围：

- 支持列（包括嵌套列）相关的增、删、改、位置调整等操作。
- 不支持对分区列做演进。
- 不支持对Array类型的嵌套列进行增、删、列操作。

5.3.1 ALTER COLUMN

命令功能

ALTER TABLE ... ALTER COLUMN语法用于修改当前列属性包括列comment、空约束，当前不支持修改列类型、列位置。

注意事项

- 目前不支持修改列类型。
- 目前不支持修改已存在列的顺序。
- 目前不支持指定顺序添加列。

命令语法

ALTER TABLE *tableName* **ALTER**

[COLUMN] *col_name*

[COMMENT] *col_comment*

参数描述

表 5-31 ALTER COLUMN 参数描述

参数	描述
tableName	表名。

参数	描述
col_name	待修改的列名。
column_type	目标列类型。
col_comment	列comment。
column_name	位置修改参照列，例如：AFTER column_name的语义是要将待修改列放到参照列column_name之后。

所需权限

- SQL权限

表 5-32 ALTER TABLE 所需权限列表

权限描述
表的ALTER权限

- 细粒度权限：dli:table:alter。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

- 其他修改

```
ALTER TABLE table1 ALTER COLUMN col_a DROP NOT NULL

ALTER TABLE table1 ALTER COLUMN col_a COMMENT 'new comment'
```

系统响应

通过运行**DESCRIBE**命令，可显示修改的列。

5.3.2 ADD COLUMNS

命令功能

ADD COLUMNS命令用于为现有表添加新列。

命令语法

ALTER TABLE *tableName* **ADD COLUMNS***(col_spec[, col_spec ...])*

参数描述

表 5-33 ADD COLUMNS 参数描述

参数	描述
tableName	表名。
col_spec	可由[col_name][col_type][nullable][comment]四部分组成。 ● col_name：新增列名，必须指定。 暂不支持给嵌套列添加新的子列 ● col_type：新增列类型，必须指定。 ● nullable：新增列是否可以为空，可以缺省。 ● comment：新增列comment，可以缺省。

所需权限

- SQL权限

表 5-34 ALTER TABLE 所需权限列表

权限描述
表的ALTER权限

- 细粒度权限：dli:table:alter。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
alter table h0 add columns(ext0 string);

alter table h0 add columns(new_col int comment 'add new column');

alter table delta.`obs://bucket_name0/db0/delta_table0` add columns(new_col string);
```

系统响应

通过运行**DESCRIBE**命令，可显示新添加的列。

5.3.3 RENAME COLUMN

命令功能

ALTER TABLE ... RENAME COLUMN语法用于修改列名称。

注意事项

- 如果您的表已经在所需的协议版本上，需要先执行如下语句才能修改成功：
  ```
  ALTER TABLE table_name SET TBLPROPERTIES ('delta.columnMapping.mode' = 'name');
  ```
- 如果您的表不在所需的协议版本上，需要先执行如下语句才能修改成功：
  ```
  ALTER TABLE table_name SET TBLPROPERTIES (
  'delta.columnMapping.mode' = 'name',
  'delta.minReaderVersion' = '2',
  'delta.minWriterVersion' = '5')
  ```

命令语法

ALTER TABLE *tableName* **RENAME COLUMN** *old_columnName* **TO** *new_columnName*

参数描述

表 5-35 参数描述

参数	描述
tableName	表名。
old_columnName	旧列名。
new_columnName	新列名。

所需权限

- SQL权限

表 5-36 ALTER TABLE 所需权限列表

权限描述
表的ALTER权限

- 细粒度权限：dli:table:alter。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
ALTER TABLE table1 RENAME COLUMN addr to address

ALTER TABLE table1 RENAME COLUMN addr.priv to province
```

a.b.c 表示嵌套列全路径，嵌套列具体规则见**ADD COLUMNS**。

系统响应

通过运行**DESCRIBE**命令查看表列修改。

5.3.4 RENAME TABLE

命令功能

ALTER TABLE ... RENAME语法用于修改表名。

命令语法

ALTER TABLE *tableName* **RENAME TO** *newTableName*

参数描述

表 5-37 RENAME 参数描述

参数	描述
tableName	表名。
newTableName	新表名。

所需权限

- SQL权限

表 5-38 ALTER TABLE 所需权限列表

权限描述
表的ALTER权限

- 细粒度权限：dli:table:alter。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
ALTER TABLE table1 RENAME TO table2
```

系统响应

通过运行**SHOW TABLES**查看新的表名。

5.3.5 DROP COLUMN

命令功能

ALTER TABLE ... DROP COLUMN语法用于删除列。

命令语法

ALTER TABLE *tableName* **DROP COLUMN|COLUMNS** *cols*

参数描述

表 5-39 DROP COLUMN 参数描述

参数	描述
tableName	表名。
cols	待删除列，可以指定多个。

所需权限

- SQL权限

表 5-40 ALTER TABLE 所需权限列表

权限描述
表的ALTER权限

- 细粒度权限：dli:table:alter。
- 由LakeFormation提供的元数据服务，权限配置详见LakeFormation文档。

示例

```
ALTER TABLE table1 DROP COLUMN a.b.c
ALTER TABLE table1 DROP COLUMNS a.b.c, x, y
```

a.b.c 表示嵌套列全路径，嵌套列具体规则见**ADD COLUMNS**。

系统响应

通过运行**DESCRIBE**命令，可查看删除列。

088_DLI Delta SQL 语法参考

6 Delta 常见配置参数

提交DLI Spark SQL作业时，在“SQL编辑器”界面右上角的“设置 > 参数设置”中配置Delta参数。

表 6-1 Delta 常见配置项

参数	描述	默认值
spark.databricks.delta.retentionDurationCheck.enabled	vacuum清理不再引用的文件时是否进行保留期检查。	true
spark.databricks.delta.properties.defaults.deletedFileRetentionDuration或delta.deletedFileRetentionDuration	Delta不再引用的文件的保留期。当spark.databricks.delta.retentionDurationCheck.enabled为true时，清空未超过保留期的文件将会抛出异常。	168小时（1周）
spark.databricks.delta.properties.defaults.logRetentionDuration或delta.logRetentionDuration	Delta log文件过期时间。每当Delta log进行checkpoint动作时，会检查是否有需要删除的过期文件，如果有，则删除这些过期文件以防Delta log文件无限增长。	30天

7 DLI Delta 常见问题

执行 insert into/overwrite table_name partition(part_key='part_value') select ...报错 DLI.0005: DeltaAnalysisException: Partition column `dt` not found in schema [id, name]

根因分析：不支持insert into/overwrite table_name partition(part_key='part_value')语法。

执行 sql 报错 DLI.0005: There should be at least one partition pruning predicate on partitioned table `777dd`.`test_delta_parts1`

解决方案：在console页面设置中添加参数spark.sql.forcePartitionPredicatesOnPartitionedTable.enabled值为false。

show create table 查看建表语句报错 DLI.0005: Operation not allowed: `SHOW CREATE TABLE` is not supported for Delta tables

根因分析：不支持该语法，可通过Describe formatted查看表结构。

执行 vacuum 报错 DLI.0001: IllegalArgumentException: requirement failed: Are you sure you would like to vacuum files with such a low retention period?

根因分析：RETAIN 保留期过短（少于168小时），需要确认是否能清理该时间前的数据，将无法再回头查看在指定数据保留期之前创建的版本。确认清理可在console页面设置中添加参数spark.databricks.delta.retentionDurationCheck.enabled值为false。

执行 rename/drop column 报错 DLI.0005: DeltaAnalysisException: Column rename is not supported for your Delta table...

解决方案：先执行如下语句，再执行rename。

```
ALTER TABLE delta_perms1 SET TBLPROPERTIES (
'delta.columnMapping.mode' = 'name',
'delta.minReaderVersion' = '2',
'delta.minWriterVersion' = '5');
```


088_DLI Delta SQL 语法参考

