Data Lake Compute

Getting Started

Product Documentation

Contents

Getting Started
Quick Start with Data Analytics in Data Lake Compute

Last updated：2022-09-20 15:03:19

Data Lake Compute allows you to quickly query and analyze COS data. Currently, CSV, ORC, Parquet, JSON, Avro, and text files are supported.

Preparations

Setting necessary internal permissions of Data Lake Compute

> Note：
> If you already have permissions or you are the root account admin, skip this step.

If you are logging in as a sub-account for the first time, in addition to the necessary CAM authorization, you also need Data Lake Compute permissions, which can be granted by a Data Lake Compute admin or root account admin through **Permission management** on the left sidebar in the Data Lake Compute console. For permission details, see Permission Overview.

1. Database and table permission: Permissions to read and write catalogs, databases, tables, and views can be granted.
2. Engine permission: Permissions to use, monitor, and modify compute engines can be granted.

> Note：
> By default, the system will activate the shared public engine based on the Presto kernel, so you can quickly try features out without purchasing a private cluster.

For detailed directions, see Sub-Account Permission Management.

Analysis Steps

Step 1. Create a database

If you are familiar with SQL statements, write the `CREATE DATABASE` statement in the query and skip the creation wizard.

1. Log in to the Data Lake Compute console and select the service region.
2. Select **Data Explore** on the left sidebar.
3. Select **Database & table**, click "+", and select **Create database** as shown below:

4. After selecting an execution engine in the top-right corner, run the `CREATE DATABASE` statement.

Step 2. Create an external table

If you are familiar with SQL statements, write the `CREATE TABLE` statement in the query and skip the creation wizard.

1. Log in to the Data Lake Compute console and select the service region.
2. Select **Data Explore** on the left sidebar.
3. Select **Database & table**, select the created table, and right-click to select **Create external table wizard**.

> Note：
> An external table generally refers to a data file stored in a COS bucket under your account. It can be directly created in Data Lake Compute for analysis with no need to load additional data. It is external, so only its metadata will be deleted when you run `DROP TABLE`, while your original data will remain.

4. Generate the table creation statement based on the wizard, and then complete the steps of setting the basic information, selecting the data format, editing the column, and editing the partition.
   - Step 1. Select the COS path of the data file (which must be a directory in a COS bucket but not a bucket itself). There is also a quick method to upload a file to COS. The operations require relevant COS permissions.

   - Step 2. Select the data file format. In the **Advanced options**, you can select automatic inference, and then the backend will parse the file format and automatically generate the table column information for fast column

inference.

> Note：
> Structure inference is an auxiliary tool for table creation and may not be 100% accurate. You need to check and modify the field names and types as needed.

- Step 3. Skip this step if there is no partition. Proper partitioning helps improve the analysis performance. For more information on partitioning, see Querying Partition Table.

5. Click **Complete** to generate the SQL statement for table creation. Then, select a data engine and run the statement to create a table.

Step 3. Run the SQL analysis

After the data is prepared, write the SQL analysis statement, select an appropriate compute engine, and start data analysis.

Sample

Write a SQL statement with all data query results being `SUCCESS` and run the statement after selecting a compute engine.

```
select * from `DataLakeCatalog`.`demo2`.`demo_audit_table` where _c5 = 'SUCCESS'
```

Quick Start with Permission Management in Data Lake Compute

Last updated：2022-08-16 09:41:59

User and work group

Data Lake Compute manages user permissions through user authorization and work group authorization.

- Work group: You can bind users to a work group to grant them the data and engine permissions of the work group. Users in the same work group have the same permissions.
- User: You can select users in CAM, including sub-accounts and collaborator accounts.

> Note：
> If users are granted different permissions from those in their work groups, all the granted permissions will take effect.

Directions

1. Select **Permission management** on the left sidebar in the Data Lake Compute console.

2. Create a work group.
   Click **Work group > Add work group**. You can bind users to the created work group or create an empty work group. For detailed directions, see User and User Group

3. Authorize the work group.
   After creating the work group, click **Authorize** in the **Operation** column to add permissions, including **Data permission** and **Engine permission**.

i. Data permission

- Data catalog permission: It includes the permissions to create databases in a data catalog and create data catalogs.
- Database and table permission: It includes fine-grained permissions at database and table levels to view and edit databases, tables, views, and functions.

ii. Engine permission

Select a data engine and grant the permissions to use, modify, or delete it.

4. Create a user.

Add a user and bind the user to a work group: Click **User > Add user**. Set **User type** to **General user** and bind the user to a work group, so that the user can get all the permissions of the work group. If you set **User type** to **Admin**, you don't need to bind the user to a work group.

5. Authorize a user.

Authorize a user in the user list. **Data permission** and **Engine permission** can be granted, just like with a work group.

For detailed directions, see Sub-Account Permission Management.

Quick Start with Partition Table

Last updated：2022-08-16 09:41:59

Data Lake Compute Partition Table

With the partition catalog feature, you can store data with different characteristics in different catalogs. In this way, when exploring data, you can filter data by partition through the `where` condition. This greatly reduces the scanned data volume and improves the query efficiency.

Note：

- Partitions in the same table should adopt the same data type and format.
- Internal tables in Data Lake Compute are implemented as implicit partitions, so you don't need to care about the partition catalog structure.

Creating a Partition Table

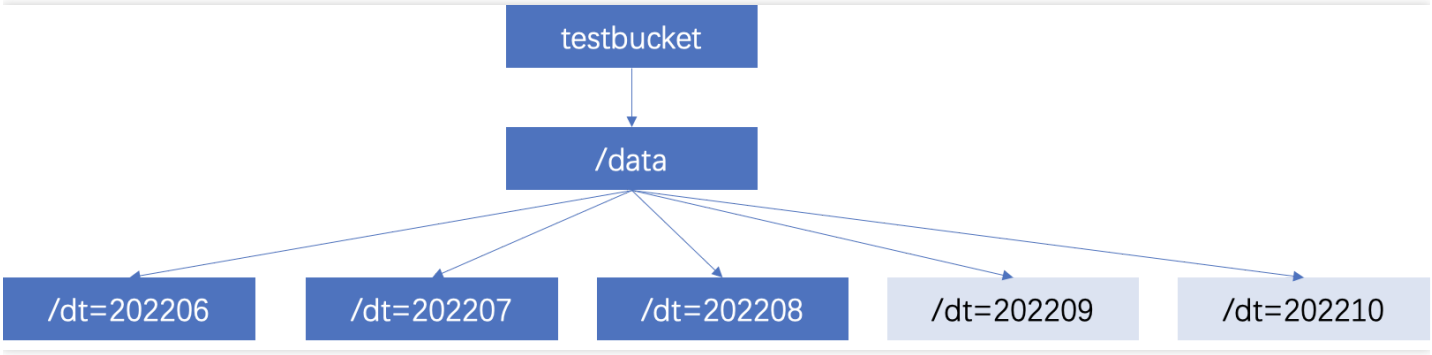
Specify the partition field through the `PARTITIONED BY` parameter in the table creation statement.
Example: Creating the `test_part` partition table

```
CREATE EXTERNAL TABLE IF NOT EXISTS `DataLakeCatalog`.`test_a_db`.`test_part` (
`_c0` int,
`_c1` int,
`_c2` string,
`dt` string
) USING PARQUET PARTITIONED BY (dt) LOCATION 'cosn://testbucket/data/';
```

Adding a Partition

Adding a partition through `ALTER TABLE ADD PARTITION`

If your data partition catalog uses the Hive partitioning rule (partition column name=partition column value), the rule can be used to add partitions. The catalog is organized as follows:




```
ALTER TABLE `DataLakeCatalog`.`test_a_db`.`test_part` add PARTITION (dt = '202206
')
ALTER TABLE `DataLakeCatalog`.`test_a_db`.`test_part` add PARTITION (dt = '202207
')
ALTER TABLE `DataLakeCatalog`.`test_a_db`.`test_part` add PARTITION (dt = '202208
')
ALTER TABLE `DataLakeCatalog`.`test_a_db`.`test_part` add PARTITION (dt = '202209
')
ALTER TABLE `DataLakeCatalog`.`test_a_db`.`test_part` add PARTITION (dt = '202210
')
```

Adding a partition by specifying the location through `ALTER TABLE`

If your data adopts a general COS catalog (not in the "partition column name=partition column value" format), you can specify a catalog when adding a partition.
Sample SQL:

```
ALTER TABLE `DataLakeCatalog`.`test_a_db`.`test_part` add PARTITION (dt = '20221
1') LOCATION='cosn://testbucket/data2/202211'
ALTER TABLE `DataLakeCatalog`.`test_a_db`.`test_part` add PARTITION (dt = '20221
2') LOCATION='cosn://testbucket/data2/202212'
```

Automatically adding a partition through `MSCK REPAIR TABLE`

Use the `MSCK REPAIR TABLE` statement to scan the data catalog specified during table creation. If there is a new partition catalog, the system will automatically add the partitions to the metadata of the data table.
Sample SQL:

```
MSCK REPAIR TABLE `DataLakeCatalog`.`test_a_db`.`test_part`
```

We recommend you use `ALTER TABLE` to add a partition preferably, as automatic adding through `MSCK REPAIR TABLE` has the following restraints:

- `MSCK REPAIR TABLE` only adds partitions to the metadata of the data table but does not delete them.
- `MSCK REPAIR TABLE` is not recommended if the data volume is large, as it will scan all the data, which may cause a timeout.
- If your partition catalog doesn't use the Hive partitioning rule (partition column name=partition column value), `MSCK REPAIR TABLE` cannot be used.

Quick Start with UDFs

Last updated：2022-08-16 09:41:59

UDF description

You can write a user-defined function (UDF), package it into a JAR file, and define it as a function in Data Lake Compute for use in query analysis. Currently, UDFs in Data Lake Compute are in the Hive format and inherit `org.apache.hadoop.hive.ql.exec.UDF` to implement the `evaluate` method.

Example: A simple array UDF

```
public class MyDiff extends UDF {
public ArrayList<Integer> evaluate(ArrayList<Integer> input) {
ArrayList<Integer> result = new ArrayList<Integer>();
result.add(0, 0);
for (int i = 1; i < input.size(); i++) {
result.add(i, input.get(i) - input.get(i - 1));
}
return result;
}
}
```

Sample POM file

```
<dependencies>
<dependency>
<groupId>org.slf4j</groupId>
<artifactId>slf4j-log4j12</artifactId>
<version>1.7.16</version>
<scope>test</scope>
</dependency>
<dependency>
<groupId>org.apache.hive</groupId>
<artifactId>hive-exec</artifactId>
<version>1.2.1</version>
</dependency>
</dependencies>
```

Creating a function

If you are familiar with the SQL syntax, you can create a function by running the `CREATE FUNCTION` statement on the **Data Explore** page. You can also create a function visually in the following steps:

> Note：
> The data management page of Data Lake Compute is currently in beta test. To try it out, submit a ticket for application.

1. Log in to the Data Lake Compute console and select the service region.
2. Select **Data management** on the left sidebar and select the target database.
3. Click **Function** to enter the function management page.
4. Click **Create function**.

You can also upload a local UDF program package or select a COS path (which requires COS permissions). The following example shows how to create a function by selecting a COS path.
The function class name includes the package information and the function execution class name.

Using a function

1. Log in to the Data Lake Compute console and select the service region.
2. Select **Data Explore** on the left sidebar and select a compute engine to use a SQL function.

Cross-Source Analysis of EMR Hive Data

Last updated：2022-08-16 09:41:59

Data Lake Compute allows you to configure an EMR Hive data source for multi-source federated data analysis.

Preparations

- Get the EMR Hive address.
- Use an account with the permission to create data catalogs. For more information on permissions, see Permission Overview.

Creating an EMR Hive data source

1. Log in to the Data Lake Compute console and select the service region.
2. Select **Data Explore** on the left sidebar, click **+** in the **Database & table** column, and select **Create data catalog**.

3. Select **EMR Hive (HDFS)** for **Connection type** and select the target EMR instance. The VPC information will be populated by default after the instance is selected. **EMR versions supported by EMR Hive are 2.3.5, 2.3.7, 3.1.1, and 3.1.2.**

   Note：
   Relevant permissions are required for you to select the EMR Hive instance.

4. Select the **Run cluster**. Currently, you can only select a private data engine of Presto. If there is no engine, create one on the **Data engine** page. For more information on the purchase process, see Purchasing Private Data Engine.

   Note：
   The IP range of the selected data engine cannot be the same as that of the EMR instance; otherwise, a network conflict will occur, and you cannot query or analyze data.

5. Click **Confirm**.

Querying the EMR Hive data

After the data catalog is created, you can switch to it from the **Data catalog** menu on the **Data Explore** page.

At this point, you can query and analyze the data catalog with SQL statements.
Select the data engine bound when the data catalog is created and click **Run** to get the query result.

> Note：
> You can only query the data catalog with its bound data engine. To change the bound engine, click the set icon next to the data catalog.