MaxCompute

Product Introduction

Product Introduction

What is MaxCompute

MaxCompute (formerly ODPS) provides a fast and fully hosting GB/TB/PB-level data warehouse solution. MaxCompute provides you with a comprehensive data import solution and a variety of classic distributed computing models, enables you to solve your massive data calculation problems more quickly, so as to effectively reduce business costs and protect data security.

In addition, MaxCompute is closely related to DataWorks. DataWorks provides one-stop data synchronization, task development, data workflow development, data operation and maintenance, and data management for MaxCompute. For more information, see DataWorks.

MaxCompute is a big data processing platform developed independently by Alibaba. It is mainly used for batch structural data processing and storage to provide massive data warehouse solutions and Big Data modeling.

Along with the diversified data collection, industrial data has also been increasingly accumulated. The data size has grown up to a massive level (TB, even PB), which the traditional software industry cannot manage. Under the analysis of massive data scenarios, the data analysts usually adopt distributed computing mode due to the limited processing capacity of the single server. But the distributed computing model demands more to the data analysts and is difficult to be maintained. Using the distributed model, data analysts not only must understand the service requirements, but also must be familiar with the underlying computing model.

MaxCompute provides a convenient way to analyze and process big data so that user can analyze big data without concerning details of distributed computing.

> MaxCompute has been widely used in Alibaba group as unified data processing platform for data warehouse, user characteristics and interest mining, data analysis of large Internet enterprises, data sharing processes, the log analysis of the websites, and the transaction analysis of E-commerce websites.

Benefits of MaxCompute

Large-scale computing and storage

MaxCompute is suitable for storage and computing large volumes of data (up to PB-level).

Multiple computation models

MaxCompute supports data processing methods based on SQL, MapReduce, Graph, MPI iteration algorithm, and other programming models.

Strong data security

MaxCompute has supported all offline business analysis of Alibaba Group for more than 7 years, providing multi-layer sandbox protection and monitoring.

Low-cost

MaxCompute can reduce the procurement costs by 20%-30% compared with self-established private cloud models.

Function

MaxCompute Tunnel

- Supports large volumes of historical data channels
  Tunnel provides high concurrency data upload and download services. YYou can use Tunnel to import TB/PB level data from various heterogeneous data sources into MaxCompute or export data from MaxCompute. As the unified channel for MaxCompute data transmission, Tunnel provides stable and high- throughput services. Tunnel provides RESTful APIs and a Java SDK to facilitate programming.

- Real-time and incremental data channels
  For real-time data upload scenarios, MaxCompute provides DataHub services with low latency and convenient usage. It is especially suitable for incremental data import. DataHub also supports a variety of data transmission plug-ins, such as Logstash, Flume, Fluentd, Sqoop.

Computing and Analysis Tasks

MaxCompute provides multiple computing models.

SQL: In MaxCompute, data is stored in forms of tables. MaxCompute provides an SQL query function for the external interface. You can operate MaxCompute similarly to traditional

database software, but still be able to process the massive data up to PB level.

NOTE:

- MaxCompute SQL does not support transactions, index, and Update/Delete operations.
- MaxCompute SQL syntax differs from Oracle and MySQL, so the user cannot migrate SQL statements of other databases into MaxCompute seamlessl, for more information, see SQL syntax.
- After you submit the MaxCompute jobs, jobs can be queued and scheduled for execution. MaxCompute SQL can complete queries in minutes or even seconds, and cannot return results in milliseconds.
- The advantage of MaxCompute SQL is to reduce users' learning cost,because users do not need to understand the concept of distribution. MaxCompute SQL can be understood by users who are familiar with database operations.

UDF: The user-defined function. MaxCompute provides a lot of built-in functions to meet your computing needs, and you can also create custom functions.

MapReduce: MaxCompute MapReduce is the Java MapReduce programming model provided by MaxCompute. It simplifies the development process, and makes it more efficient. Users of MaxCompute MapReduce must have a basic understanding of the concept of distribution and the corresponding programming experience. MaxCompute MapReduce provides Java programming interface.

Graph: MaxCompute Graph is a processing framework designed for iterative graph computing. MaxCompute Graph jobs use graphs to build models. Graphs are composed of vertices and edges. Vertices and edges contain values. After performing iterative graph editing and evolution, you can get the final result. Typical applications include PageRank, SSSP algorithm, and K-Means algorithm.

SDK

Toolkit provided for the developers. For more information, see MaxCompute SDK.

Security

MaxCompute provides a powerful security services and provides protection for the user's data. For more information about each function model, see MaxCompute Security Manual.

Next step

You have learned about MaxCompute's benefits and functions, now you can continue to learn the

next tutorial in which you will learn about related charges of MaxCompute. For more information, see Pricing.

History

From its incorporation in September 2009, Alibaba Cloud team has envisioned creating the data operations/sharing platform. In April 2010, ODPS (now MaxCompute) officially went into operation alongside the start of Ant Financial' s loans business. In 2012, a centralized data platform has been established. By 2013, the platform possessed ultra-large scale data processing capabilities. From 2014 to 2015, the big data platform gradually became more sophisticated. In 2016, initial vision was realized and MaxCompute 2.0 was produced.

Key milestones

April 2010, ODPS official goes into production operation, Ant Financial steadily launches its loan business.

May 2013, ODPS beta testing.

July 2013, ODPS is officially launched as a commercial service. A single cluster contains 5,000 servers and the service supports multi-level clusters.

September 2016, ODPS is renamed MaxCompute. MaxCompute 2.0 is launched, providing high performance, new functions, and a rich ecosystem.

Definition

Project

Project is the basic unit of operation in MaxCompute. It is similar to the concept of Database or

Schema in traditional databases, and sets the boundary for MaxCompute multi-users isolation and access control. User can have multiple project permissions at the same time. By means of security authorization, users can access the objects of another project in their own project, such as Table, Resource, Function and Instance.

To enter the project run the **Use Project** command, as follows:

```
use my_project   -- Enter a project named 'my_project'.
```

After running this command, you can enter a project named **my project** and all objects in this project can be operated, such as Table, Resource, Function, Instance. **Use Project** is a command provided by MaxCompute client. For more commands, see Common Commands.

Table

Table is a data storage unit in MaxCompute. Table is a two-dimensional data structure composed of rows and columns. Each row represents a record, each column represents a field with the same data type. One record can contain one or more columns. Column name and data type consist of the schema of this table.

The operating objects (input, output) of various computing tasks in MaxCompute are tables. You can create a table, delete a table, and import data into a table.

> NOTE:
>
> The data management module of DataWorks can perform operations such as creating, collecting, modifying data lifecycle, modifying the table structure, and managing table/resource/function permissions on the MaxCompute table. For more information, see Data Management Overview.

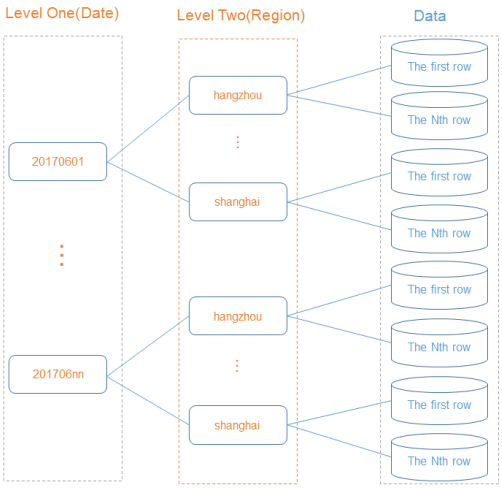
Two types of MaxCompute tables exist: internal tables and external tables (external tables are supported in MaxCompute 2.0).

- For internal tables, all data is stored in MaxCompute tables, the columns in the table can be any of the data types supported by MaxCompute.
- For external tables, the data is not really stored in MaxCompute. The table data can be stored in OSS or Table Store. MaxCompute only records meta information of the table. You can use MaxCompute’ s external table to process unstructured data on OSS or Table Store, such as video, audio, genetics, weather, and geographic information.

Partition

To improve the processing efficiency, you can specify the partition when creating a table. That is, several fields in the table are specified as partition columns. In most cases, you can consider the partition to be the directory under the file system.

In MaxCompute each value of partition column is used as a partition (directory). You can specify multiple field of the table as a partition, so that make partitions to be multi-level directories. If the partitions to be accessed are specified when you use data, then only corresponding partitions are read and full table scan is avoided, which can improve the processing efficiency and reduce costs.



Partition types

MaxCompute 2.0 supports more partition types. Currently, MaxCompute supports the following partition types: TINYINT, SMALLINT, INT, BIGINT, VARCHAR, and STRING.

NOTE:

In earlier versions of MaxCompute, only STRING partition type was supported. Although the partition type can be specified as BIGINT, it is still handled as STRING in fact, only the schema of the table is indicated as a BIGINT type.

An example is as follow:

```
create table parttest (a bigint) partitioned by (pt bigint);
insert into parttest partition(pt) select 1, 2 from dual;
insert into parttest partition(pt) select 1, 10 from dual;
select * from parttest where pt >= 2;
```

After the execution, the returned result is only one line, because 10 was treated as a string and

compared with 2, so no result can be returned.

Restrictions

The partition has the following usage restrictions:

- Partition level of a single table is up to 6 levels.
- The maximum number of single table partitions is 60,000.
- The maximum number of query partitions for a query is 10,000.

For example, to create a two-level partition table with the date as the level one partition and the region as the level two partition:

```
create table src (key string, value bigint) partitioned by (pt string,region string);
```

When querying, use the partition column as a filter condition in the **Where** condition filter:

```
select * from src where pt='20170601' and region='hangzhou';      -- This is the correct way to use it. When MaxCompute generates a query plan, only data which region is 'hangzhou'  under the '20170601' partition is accessed.
select * from src where pt = 20170601; -- This is the wrong way to use it. In this way, the effectiveness of the partition filter cannot be guaranteed. Pt is a String type. When the String type is compared with Bigint type(20170601), MaxCompute converts both to Double type, and loss of precision occurs.
```

Some of SQL operations on the partitions are less efficient and may cause higher billing, for example, using dynamic partition.

For some MaxCompute commands, there is a difference in syntax when performing operations on partitioned and non-partitioned tables. For more information, see DDL SQL and DML SQL.

Data type

The basic data types supported by MaxCompute2.0 are listed in the following table, new types include Tinyint, Smallint, Int, Float, Varchar, Timestamp, and Binary data type. The columns in the MaxCompute table must be any of the listed types.

Notes:

- Once data type such as Tinyint, Smallint, Int, Float, Varchar, Timestamp, or Binary is involved when running an SQL command, set odps.sql.type.system.odps2=true; must be added before the SQL command. The set command and SQL command are submitted

simultaneously. When INT type is involved, if the set command is not added, the Int type is converted to Bigint which is 64 bits.

Type	New	Constant	Description
Bigint	No	100000000000L, -1L	64-bit signed integer, range -263 + 1 to 263 - 1
String	No	“abc”,’bcd’,”alibaba” ‘inc’ (Note3)	The threshold of single string length is 8M
Boolean	No	TRUE,FALSE	True/False, Boolean type
Double	No	3.1415926 1E+7	64-bit binary floating point
Datetime	No	DATETIME ‘2017-11-11 00:00:00’	0001-01-01 00:00:00 ~ 9999-12-31 23:59:59, Date type, use UTC+8 as the standard time system
Tinyint	Yes	Y,-127Y	8-bit signed integer, range -128 to 127
Smallint	Yes	32767S, -100S	16-bit signed integer, range -32768 to 32767
Int	Yes	1000,-15645787 (Note1)	32-bit signed integer-231 to 231 - 1
Float	Yes	None	32-bit binary floating point
Varchar	Yes	None (Note2)	Variable-length character type, n is the length, and the range is 1 to 65535.
Binary	Yes	None	Binary data type, the threshold of single string length is 8M
Timestamp	Yes	TIMESTAMP ‘2017-11-11 00:00:00.123456789’	It is independent of the time zone and ranges from January 1st 0000 to December 31, 9999 23.59:59.999999999, which is accurate to nanosecond.

All data types in the preceding table can be NULL.

NOTES:

- **Note1**: For INT constant, if the range of Int is exceeded, Int is converted into Bigint; if the range of Bigint is exceeded, it is converted into Double. In the earlier versions of MaxCompute, all Int types in SQL script were converted to Bigint , for example:

```
create table a_bigint_table(a int);
select cast(id as int) from mytable;
```

  To be compatible with the earlier MaxCompute versions, MaxCompute 2.0 still keeps this conversion without setting odps.sql.type.system.odps2 as true, but a warning that INT is treated as BIGINT is triggered. If you face such a situation, we recommend that you change an Int to a Bigint to avoid confusion.
- **Note2**: VARCHAR constants can be expressed by STRING constants of implicit transformation.
- **Note3**: STRING constants support connections, for example, 'abc' 'xyz' is parsed as 'abcxyz', and different parts can be written on different lines.

Complex Data Types

All complex data types that MaxCompute 2.0 supports are listed in the following table.

NOTE:

Once complex data type is involved when you run a SQL command, set odps.sql.type.system.odps2=true; must be added before the SQL command. The set command and SQL command are submitted simultaneously.

Type	Definition method	Construction method

ARRAY	array< int >;
	array< struct< a:int, b:string >>
		array(1, 2, 3);
		array(array(1, 2);
		array(3, 4))

MAP	map< string, string >; map< smallint, array< string>>	map("k1" , "v1" , "k2" , "v2"); map(1S, array('a' , 'b'), 2S, array('x' , 'y'))
STRUCT	struct< x:int, y:int>; struct< field1:bigint, field2:array< int>, field3:map< int, int>>	named_struct('x' , 1, 'y' , 2); named_struct('field1' , 100L, 'field2' , array(1, 2), 'field3' , map(1, 100, 2, 200)

Lifecycle

The lifecycle of a MaxCompute table or partition is counted from the last update time. If the table or partition remains unchanged after a specified time, MaxCompute automatically recycles it. The **specified time** is lifecycle.

Lifecycle units: days, positive integers only.

When a lifecycle is specified for a non-partition table, the lifecycle is counted from the last time the table data was modified (LastDataModifiedTime). If table data has not been changed, MaxCompute recycles the table automatically without manual operation (similar to the drop table operation).

> **NOTE**:
>
> Lifecycle scanning is started on schedule every day. The entire partitions are be scanned. If the partition remains unchanged after its lifecycle, MaxCompute automatically recycles it.
> Suppose that the lifecycle of a partition is 1 day, and the latest time the partition was updated is June 18,2017 PM 03:07:00. It was discovered that the partition has been recycled by MaxComputeon on June 20,2017 PM 01:07:00 after scanning.

When a lifecycle is specified for a partition table, MaxCompute determines whether to recycle the partition based on its LastDataModifiedTime. Unlike non-partition tables, a partition table cannot be deleted after all its partitions have been recycled.

You can only set the lifecycle of tables, but not of partitions. The lifecycle can be specified when the table is created.

If no lifecycle is specified, the table or partition cannot be recycled by MaxCompute automatically according to lifecycle rules.

For more information on specifying/modifying lifecycle during table creation and modifying a table’ s LastDataModifiedTime, see DDL documentation.

Resources

Resources is a unique concept of MaxCompute. If you want to use user-defined function UDF or MapReduce, you must use recources to accomplish tasks.

SQL UDF: After writing a UDF, you must upload the compiled Jar package to MaxCompute as a resource. When you run this UDF, MaxCompute automatically downloads this Jar package to obtain the code you wrote. To upload the jar package is a course to create MaxCompute resource, so jar package is a kind of MaxCompute resource. The process of uploading Jar package is to create a resource in MaxCompute. The Jar package is one type of MaxCompute resource.

MapReduce: After writing a MapReduce program, you must upload the compiled Jar package to MaxCompute as a resource. When running a MapReduce job, the MapReduce framework will automatically download this Jar package and obtain the written code. Similarly, you can upload text files and MaxCompute tables to MaxCompute as different types of resources. Then, you can read or use these resources when running UDF or MapReduce.

MaxCompute provides interfaces for you to read and use resources. For more information, see Use Resourse Example and UDTF Usage .

NOTE:

There are limits on resource reading by MaxCompute’ s user-defined function (UDF) or MapReduce function. For more information, see Application Restriction.

Types of MaxCompute resources include:

- File type

  Tablet ype, which are tables in MaxCompute

  NOTE: Currently, only Bigint, Double, String, Datetime, Boolean fields are supported in tables referenced by MapReduce.

  Jar type, which is compiled Java Jar packages

  Archive type, which is the compression type, and is determined by the resource name suffix. Supported compression types include: .zip/.tgz/.tar.gz/.tar/jar

For more information about resources, see Add Resource, Drop Resource, List Resources and Describe Resource.

Function

MaxCompute provides SQL computing capabilities. In MaxCompute SQL, you can use the system's built-in functions to perform certain computing and counting tasks. However, if such SQL functions do not meet your requirements, you can use the Java programming interface provided by MaxCompute to develop user-defined functions (UDF). User-defined functions (UDFs) can be further divided into scalar valued functions (UDFs), user-defined aggregate functions (UDAFs), and user-defined tables functions (UDTFs).

After writing the code for a UDF, you must compile the code into a Jar package and upload this package to MaxCompute. Then, you can register this UDF in MaxCompute.

> **NOTE:**
>
> When using UDFs, you must only specify the UDF name and input the relevant parameters in SQL. UDFs are used in the same way as the built-in functions provided by MaxCompute.

For more information, see Function introduction.

Task

Task is the basic computing units of MaxCompute. Both SQL and MapReduce implement functions using tasks.

When you submit a large number of tasks, especially computing tasks, such as SQL DML statements, MapReduce, and other such tasks, MaxCompute parses them the tasks to create a task execution plan. The execution plan is composed of multiple inter-dependent execution stages.

Currently, the execution plan logic can be viewed as a directed graph, with the vertices representing stages and the edges representing the dependencies between stages. MaxCompute executes the various stages according to the dependencies in the graph (execution plan).

A single stage comprises multiple threads, also known as workers, which perform the computing task in this execution stage. The different workers of a single stage process different data, but have identical execution logic. When executed, a computing task is turned into an instance. You can then

use the information of this instance, for example, Status Instance and Kill Instance.

However, not all MaxCompute tasks are computing tasks (such as DDL statements in SQL). Essentially, these tasks only must read or modify metadata in MaxCompute. Therefore, these tasks cannot be parsed into an execution plan.

> NOTE:
>
> Not all the requests are converted into tasks in MaxCompute, for example, the operations of Project, Resource, UDF and Instance can be completed without MaxCompute tasks.

Instance

In MaxCompute, some tasks are turned into MaxCompute instances for execution. These instances experience two stages: Running and Terminated. In the Running stage, the instance status is Running. In the Terminated stage, the instance status can be Success, Failed, or Canceled. You can query or modify instance status using the instance ID assigned by MaxCompute. For example:

```
status <instance_id>;          --View the satatus of a certain instance.
kill <instance_id>; --Stop an instance and set its status to be  'Canceled' .
wait <instance_id>; --View the running logs of a certain instance.
```

Reading guidance

For abecedarians

If you are learning MaxCompute for the first time, we recommend that you begin by reading the following sections:

MaxCompute Summary — This chapter is a general introduction of MaxCompute and its major features. By reading this chapter, you can get a general understanding of MaxCompute.

Quick Start — This chapter provides detailed examples, and guides you step-by-step how to

apply for an account, install the client, create a table, authorize a user, export/import data, run SQL tasks, UDFs, and Mapreduce programs.

Basic Introduction — This chapter introduces basic concepts and common commands of MaxCompute. You can can become more familiar with how to operate MaxCompute.

Tools — Before analyzing the data, you must know how to download, configure, and use MaxCompute’ s common tools. The following tool is provided:

- MaxCompute Client: You can use this tool to operate MaxCompute.

As you become familiar with the preceding modules, we recommend that you continue reading for a deeper understanding of MaxCompute.

For data analysts

If you are a data analyst, we recommend that you read the following section about MaxCompute SQL.

MaxCompute SQL: You can query and analyze massive data that stored on MaxCompute. The main features are as following:

It supports DDL statements. You can use **Create**, **Drop**, and **Alter** statements to manage tables and partitions.

You can select a few records from a table by using **Select** clause. You can query records which meet the conditions in the **Where** clause.

You can achieve the association between the two tables by the **Join** equivalent connection.

You can perform the aggregation operation using **Group By** clause.

You can insert the result records into another table using **Insert overwrite/into** statement.

You can use built-in functions and user-defined functions (UDFs) to implement a series of computing tasks.

For developers

If you are an experienced developer and have a knowledge of distributed system, considering that some data analysis cannot be completed by SQL, we recommend that you learn advanced modules of MaxCompute, as following:

MapReduce: MaxCompute provides MapReduce programming interface. You can use the Java API (provided by MapReduce) to write MapReduce program for processing data in MaxCompute.

Graph: A set of framework for iterative graph processing. You can use the graph for data modeling. Graphs are composed of vertices and edges. Vertices and edges contain values. Process outputs a final solution after performing iterative graph editing and evolution.

Eclipse Plugin: Facilitates using of Java SDK of MapReduce, UDF, and Graph.

Tunnel: You can use Tunnel to import data from various heterogeneous data sources into MaxCompute or export data from MaxCompute

SDK:

- Java SDK: Provides developers with Java interfaces.
- Python SDK: Provides developers with Python interfaces.

NOTE:

Currently, MapReduce and Graph functions are still in open beta. If you want to use these functions, submit an application by the work order system, specify your project name, and we will manage it within 7 working days.

For project owners or project administrators

If you are a project owner or administrator, we recommend that you read the following sections:

- Security: By reading this chapter, you can understand how to authorize users, share resources across the projects, set data protection features, and perform policy authorization.
- Billing: Chapter introduce the charging model of MaxCompute.
- Commands that only the project owner can use. For example, the SetProject operation in Others of *Common Commands*.Some commands are only available to the project owner, for example, **SetProject**, for more infoormation about this kind of commands, see Others of Common commands.