ALIBABA CLOUD

Alibaba Cloud

MaxCompute

JDBC Reference

Document Version: 20220330

Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	**Danger:** Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	**Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.Version updates

This topic describes the latest version updates of Java Database Connectivity (JDBC), including new and enhanced features of a specific version of JDBC.

The following table describes the latest version updates of JDBC. For more information, visit the URL of the related version.

Version	Change type	Description
v3.2.9	New feature	The connection string parameter `useProjectTimeZone` is added. This parameter is optional. Default value: False. If you configure `useProjectTimeZone=true`, the return values of `ResultSet#getDate`, `ResultSet#getTime`, and `ResultSet#getTimestamp` are determined based on the time zone where the current project resides.
	Enhanced feature	The logs that record job failures are optimized.
	Fixed issue	The Maven AspectJ plug-in cannot be used. This avoids possible performance losses.
v3.2.8	New feature	The MaxCompute Query Acceleration (MCQA) feature is supported.
	Enhanced feature	• In MCQA mode, you can add the option `instanceTunnelMaxRecord` to specify the maximum number of rows of results that can be read. The default value is -1, which indicates an unlimited number. This option is supported only for JDBC 3.2.7 or later. • In MCQA mode, you can add the option `instanceTunnelMaxSize` to specify the maximum size of data that can be read. The unit is bytes. The default value is -1, which indicates an unlimited number. This option is supported only for JDBC 3.2.7 or later. • In MCQA mode, you can add the option `disableConnectionSetting` to disable connection-level settings. • If you add the option `alwaysFallback` and set it to True in MCQA mode, the MCQA mode is rolled back to the offline mode in the preceding scenarios. This option is supported only for JDBC 3.2.3 or later.

2.Overview

The MaxCompute Java Database Connectivity (JDBC) driver provides a standard JDBC API. You can perform distributed computing and queries on large amounts of data in MaxCompute by using the JDBC API. You can also use the JDBC driver to connect tools that support JDBC to MaxCompute.

Usage notes

- Use the MaxCompute JDBC driver to connect to MaxCompute. For more information, see Usage notes.
- Use the MaxCompute JDBC driver to connect Business Intelligence (BI) analysis tools to MaxCompute. This way, you can use the tools to analyze data in MaxCompute in a visualized manner. References:
  - Connect Tableau to MaxCompute
  - Connect Davinci to MaxCompute
  - Connect Yonghong BI to MaxCompute
- Use the MaxCompute JDBC driver to connect database management tools to MaxCompute. This way, you can use the tools to manage MaxCompute projects. References:
  - Connect DBeaver to MaxCompute
  - Connect DataGrip to MaxCompute
  - Connect SQL Workbench/J to MaxCompute

Usage notes

- To execute SQL statements and obtain execution results by using the MaxCompute JDBC driver, you must meet the following requirements:
  - You are a member of a project.
  - You have the CREATE INSTANCE permission on the project.
  - You have the SELECT and DOWNLOAD permissions on the table that you want to use.

    > **Note**
    > - When you use MaxCompute JDBC V1.9 or earlier, a temporary table is automatically created for each query. You can use Tunnel commands to obtain query results from the temporary table. To use these versions, you must have the CREATE TABLE permission.
    > - When you use MaxCompute JDBC V2.2 or later, no temporary table is automatically created for each query. You can call the InstanceTunnel interface to obtain query results, regardless of whether you have the CREATE TABLE permission.
    >
    > For more information about MaxCompute permissions, see Authorize users.

- MaxCompute provides the data protection feature. If the data protection feature is enabled for a project, you cannot move data out of the project. If you use MaxCompute JDBC of a version earlier than V2.4, no `result sets` can be obtained. If you use MaxCompute JDBC V2.4 or later, the number of result rows that you obtain cannot exceed the value of the READ_TABLE_MAX_ROW parameter. For more information about this parameter, see 项目空间操作. For more information about the data protection feature, see Project data protection.
- The MaxCompute V2.0 data type edition supports more data types, such as TINYINT, SMALLINT, DATETIME, TIMESTAMP, ARRAY, MAP, and STRUCT. To use these new data types, you must run the following command to enable the MaxCompute V2.0 data type edition. For more information, see

Data type editions.

```
set odps.sql.type.system.odps2=true
```

FAQ

- How do I view the log file of the MaxCompute JDBC driver?
  - By default, the log file of the MaxCompute JDBC driver is stored in the same directory as the JAR package of the MaxCompute JDBC driver. The file name is jdbc.log.
  - If the code and the MaxCompute JDBC driver are in the same uber JAR package, the log file of the MaxCompute JDBC driver is stored in the same directory as the uber JAR package.

  The logs of the MaxCompute JDBC driver show details about JDBC API calls, such as class names, method names, parameters, return values, and the number of rows. You can use the information for debugging.
- How do I obtain a MaxCompute Logview URL?

  The MaxCompute JDBC driver is encapsulated based on MaxCompute SDK for Java. Logview URLs are generated when you execute SQL statements in the MaxCompute client, MaxCompute Studio, and DataWorks. Logview URLs are also generated when you use the MaxCompute JDBC driver to execute SQL statements. You can use the Logview URLs to view the job status, track job progresses, and obtain job execution results. A Logview URL is configured by using the properties.log4j parameter. By default, logs are displayed as standard error logs.
- Does MaxCompute support connection pools and the auto-commit mode?

  MaxCompute provides REST services that are different from long connections in traditional databases. It is considered a lightweight task for the MaxCompute JDBC driver to establish a connection. Although the driver supports scenarios in which connection pools are used, connection pools are unnecessary for the MaxCompute JDBC driver.

  MaxCompute does not support transactions. Each query is immediately performed on the server. The auto-commit mode is automatically enabled for the MaxCompute JDBC driver. You cannot disable the auto-commit mode for the MaxCompute JDBC driver.
- How do I obtain partition fields and data types?

  You can use the `Connection.getMetadata()` method to obtain DatabaseMetaData objects and use the `getColumns()` method to obtain the metadata of all columns.

3.Usage notes

This topic describes how to download the MaxCompute JDBC driver and use the driver to connect to MaxCompute. It also provides sample code.

Download the MaxCompute JDBC driver

You can download the JAR packages of various MaxCompute JDBC driver versions from GitHub or Maven. We recommend that you download the JAR package whose name includes jar-with-dependencies.

The following code describes the Project Object Model (POM) file of the MaxCompute JDBC driver downloaded from Maven:

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>odps-jdbc</artifactId>
  <version>3.0.1</version>
  <classifier>jar-with-dependencies</classifier>
</dependency>
```

Note The MaxCompute JDBC driver is an open-source code project that is named aliyun-odps-jdbc.

You are welcome to participate in the development and improvement of the MaxCompute JDBC driver. You can report issues on the **Issues** tab and improve source code on the **Pull requests** tab. Your operations on the **Issues** and **Pull requests** tabs must follow the template requirements for open-source projects.

Connect to MaxCompute

1. Load the MaxCompute JDBC driver.

   ```
   Class.forName("com.aliyun.odps.jdbc.OdpsDriver");
   ```

2. Use DriverManager to establish a connection to MaxCompute.

   ```
   Connection cnct = DriverManager.getConnection(url, accessId, accessKey);
   ```

   - url: The URL must be in the format of `jdbc:odps:<maxcompute_endpoint>?project=<maxcompute_project_name>`. In the URL:
     - <maxcompute_endpoint>: the endpoint of MaxCompute in a specific region. For example, the public endpoint of MaxCompute in the China (Hangzhou) region is `http://service.cn-hangzhou.maxcompute.aliyun.com/api`. For more information, see Endpoints.
     - <maxcompute_project_name>: the name of your MaxCompute project.

     Example:

     ```
     jdbc:odps:http://service.cn-hangzhou.maxcompute.aliyun.com/api?project=test_project
     ```

   - accessId: the AccessKey ID of the account used to create the project.
   - accessKey: the AccessKey secret of the account used to create the project.

> **Note** For more information about how to create and view the AccessKey ID and AccessKey secret, see Create an Alibaba Cloud account.

3. Perform a query.

```
Statement stmt = cnct.createStatement();
ResultSet rset = stmt.executeQuery("SELECT foo FROM bar");
while (rset.next()) {
    // process the results
}
rset.close();
stmt.close();
conn.close();
```

Sample code

- Delete a table, create a table, and obtain metadata

```
import java.sql.Connection;
import java.sql.DatabaseMetaData;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
public class Main {
    private static final String DRIVER_NAME = "com.aliyun.odps.jdbc.OdpsDriver";
    public static void main(String[] args) throws SQLException {
        try {
            Class.forName(DRIVER_NAME);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
            System.exit(1);
        }
        Connection conn = DriverManager.getConnection(
            "jdbc:odps:<maxcompute_endpoint>? project=<maxcompute_project>",
            "aliyun accessId", "aliyun accessKey");
        // create a table
        Statement stmt = conn.createStatement();
        final String tableName = "jdbc_test";
        stmt.execute("DROP TABLE IF EXISTS " + tableName);
        stmt.execute("CREATE TABLE " + tableName + " (key BIGINT, value STRING)");
        // get meta data
        DatabaseMetaData metaData = conn.getMetaData();
        System.out.println("product = " + metaData.getDatabaseProductName());
        System.out.println("jdbc version = "
            + metaData.getDriverMajorVersion() + ", "
            + metaData.getDriverMinorVersion());
        ResultSet tables = metaData.getTables(null, null, tableName, null);
        while (tables.next()) {
            String name = tables.getString("TABLE_NAME");
            System.out.println("inspecting table: " + name);
            ResultSet columns = metaData.getColumns(null, null, name, null);
            while (columns.next()) {
                System.out.println(
                    columns.getString("COLUMN_NAME") + "\t" +
                        columns.getString("TYPE_NAME") + "(" +
                        columns.getInt("DATA_TYPE") + ")");
            }
            columns.close();
        }
        tables.close();
        stmt.close();
        conn.close();
    }
}
```

The following result is returned:

```
product = MaxCompute/ODPS
jdbc version = 3, 0
inspecting table: jdbc_test
key    BIGINT(-5)
value    STRING(12)
```

- Update a table

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
public class Main {
    private static final String DRIVER_NAME = "com.aliyun.odps.jdbc.OdpsDriver";
    public static void main(String[] args) throws SQLException {
        try {
            Class.forName(DRIVER_NAME);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
            System.exit(1);
        }
        Connection conn = DriverManager.getConnection(
            "jdbc:odps:<maxcompute_endpoint>? project=<maxcompute_project>",
            "aliyun accessId", "aliyun accessKey");
        Statement stmt = conn.createStatement();
        // The following DML also works
        //String dml = "INSERT INTO jdbc_test SELECT 1, \"foo\"";
        String dml = "INSERT INTO jdbc_test VALUES(1, \"foo\")";
        int ret = stmt.executeUpdate(dml);
        assert ret == 1;
        stmt.close();
        conn.close();
    }
}
```

- Update multiple tables at a time

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
public class Main {
    private static final String DRIVER_NAME = "com.aliyun.odps.jdbc.OdpsDriver";
    public static void main(String[] args) throws SQLException {
        try {
            Class.forName(DRIVER_NAME);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
            System.exit(1);
        }
        Connection conn = DriverManager.getConnection(
            "jdbc:odps:<maxcompute endpoint>? project=<maxcompute project>",
            "aliyun accessId", "aliyun accessKey");
        PreparedStatement pstmt = conn.prepareStatement("INSERT INTO jdbc_test VALUES(?,
?)") ;
        pstmt.setLong(1, 1L);
        pstmt.setString(2, "foo");
        pstmt.addBatch();
        pstmt.setLong(1, 2L);
        pstmt.setString(2, "bar");
        pstmt.addBatch();
        int[] ret = pstmt.executeBatch();
        assert ret[0] == 1;
        assert ret[1] == 1;
        pstmt.close();
        conn.close();
    }
}
```

- Query a table

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
public class Main {
    private static final String DRIVER_NAME = "com.aliyun.odps.jdbc.OdpsDriver";
    public static void main(String[] args) throws SQLException {
        try {
            Class.forName(DRIVER_NAME);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
            System.exit(1);
        }
        Connection conn = DriverManager.getConnection(
            "jdbc:odps:<maxcompute endpoint>? project=<maxcompute project>",
            "aliyun accessId", "aliyun accessKey");
        ResultSet rs;
        Statement stmt = conn.createStatement();
        String sql = "SELECT * FROM JDBC_TEST";
        stmt.executeQuery(sql);
        ResultSet rset = stmt.getResultSet();
        while (rset.next()) {
            System.out.println(String.valueOf(rset.getInt(1)) + "\t" + rset.getString(2))
;
        }
    }
}
```