Testing JDBC connections

Revision 20221103

Denodo Inc.: 525 University Avenue, Suite 31, Palo Alto, CA 94301. USA www.denodo.com

Goal

Sometimes we may experience some problems connecting from Denodo Virtual DataPort to a JDBC data source due to external issues like network configuration or others that are not Denodo related. When a connection issue occurs, it is a good idea to isolate factors beyond Denodo at first to accelerate the troubleshooting process.

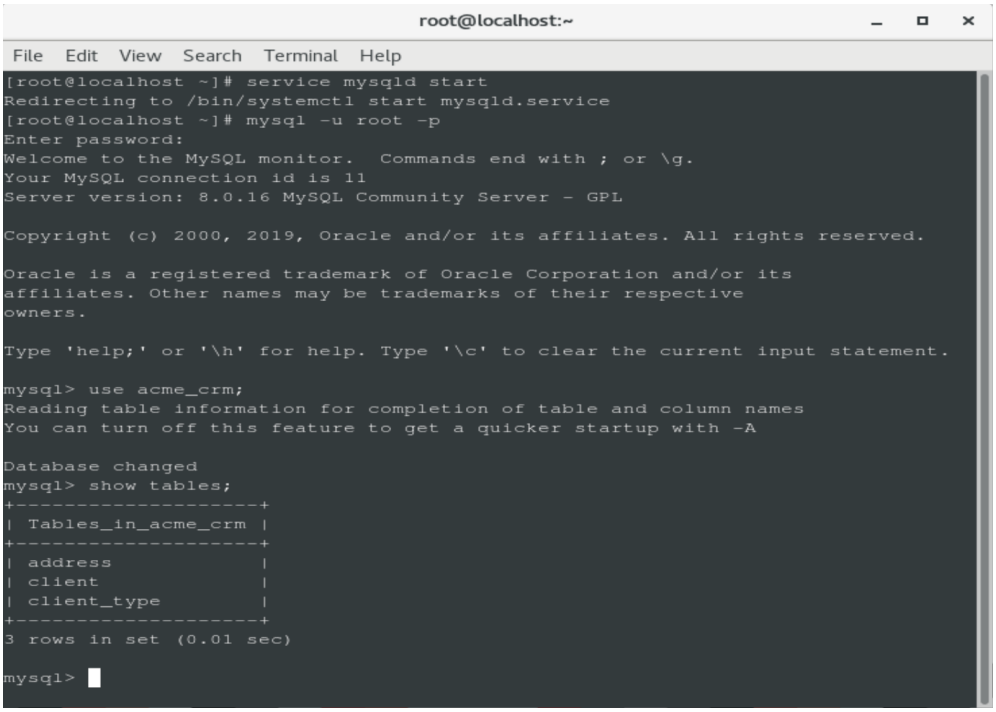
To achieve this goal, this document describes how to test a JDBC connection from a third-party lightweight JDBC client installed on the same host as Denodo Virtual DataPort to a JDBC data source.

A common scenario will be a Denodo installation on a server without a graphical interface or desktop environment. In this document we will use a Linux OS (CentOS Linux 7) system without a desktop environment and we will use the Jisql JDBC client. The same client can be installed and used on Windows systems.

Content

Sample data

The sample data from Data Virtualization Basics Tutorial will be used. A database named `acme_crm` is created in a MySQL database along with three tables: `address`, `client`, `client_type`.



Overview of Jisql

Jisql [Jisql - a Java based interactive SQL application], distributed under the Apache License v2.0, is a Java-based utility that provides a command line interactive session with a SQL server. This SQL client can connect to any database with a JDBC driver so it is a useful and convenient tool in order to interact with any JDBC data source from the command line in non-GUI environments.

Downloading Jisql

The complete Jisql package (source code, Javadoc, build environment) can be downloaded from the project's website at Jisql - a Java based interactive SQL application

According to the online documentation Jisql has been compiled with Java 6 and it works perfectly with this version and newer ones. The reason Java 6 is required is a call to a Java6-only class (the `Console` class) but, thanks to the distributed package containing the source code and ant build files, this dependency could be removed with a simple change in `Jisql.java` to make it work with older versions of Java if needed.

Once you have downloaded the package you must unzip it and then copy to the generated `jisql-2.0.11/lib` folder the JDBC driver, in our example the MySQL connector jar file (`mysql-connector-java.jar` or equivalent) corresponding to the MySQL server version used.

Executing Jisql

The software distribution includes several `runit` script files (e.g. `runit.bat`) that can be used for easily executing Jisql. These batch scripts are for Windows systems. For Linux you can create bash scripts and modify the commands inside to suit your needs, e.g specify your JDBC connection strings and include the necessary `.jar` files into the application's classpath. Here you have an example to run Jisql from Linux to connect to a sample MySQL database.

If you do not use the `-input` parameter, which executes commands from a file, the interactive command line will be opened and it will allow you to execute queries. To close the application you have to use the command `quit` or the command `exit`.

Note: Running Jisql from Linux

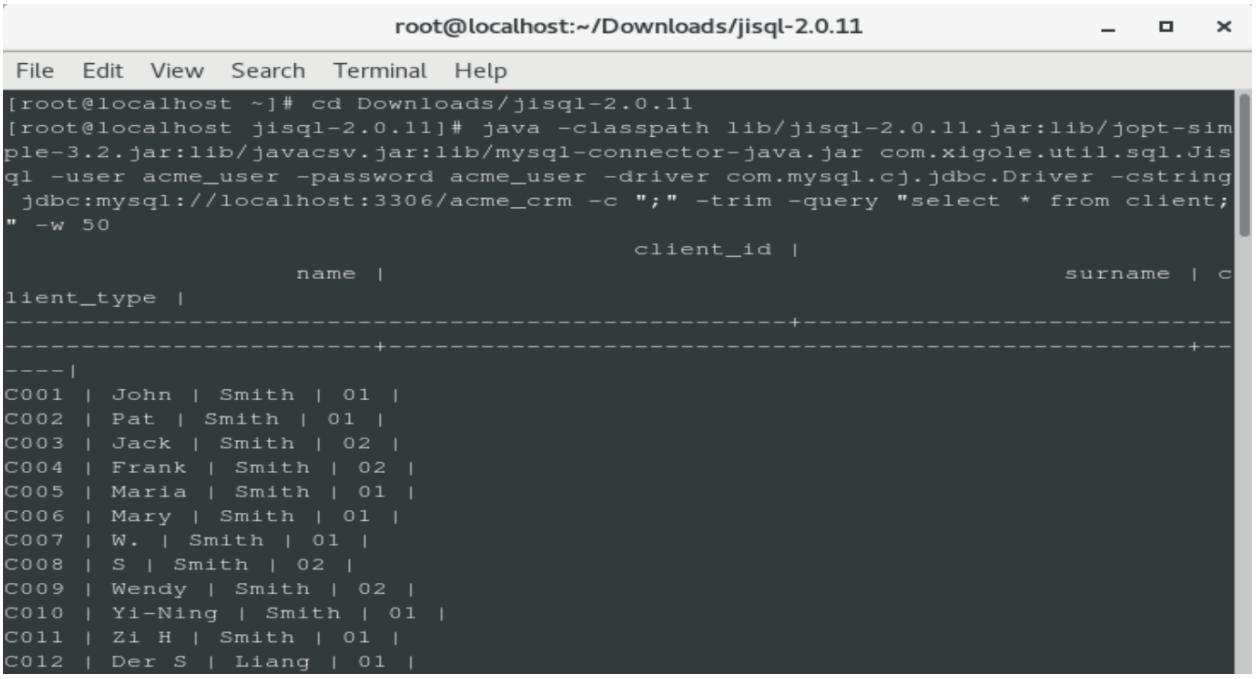
For the -classpath parameter the jar files need to be separated with a colon (semicolon on Windows systems), and in the -c parameter the semicolon needs to be surrounded with quotation marks (no quotation marks on Windows).

```
java -classpath lib/jisql-2.0.11.jar:lib/jopt-simple-3.2.jar:lib/javacsv.jar:lib/mysql-connector-java.jar com.xigole.util.sql.Jisql -user acme_user -password acme_user -driver com.mysql.cj.jdbc.Driver -cstring jdbc:mysql://localhost:3306/acme_crm -c ";"
```

Running queries

In order to run queries you can use the interactive command line or the `-query` parameter. The former example is shown in the last section . The latter opens a connection, executes the query, shows the results in the command line and closes the connection.

Here is an example of command line using `-query` parameter:



Available parameters

`-classpath` The elements that you have to add to your classpath. If it is in a different directory you have to write the full path.

`-user` The username used to log in to the database.

`-password` The password used to log in to the database. If this option is missing the program asks for the password.

`-driver` The JDBC driver class name of the driver. In our case it is `com.mysql.cj.jdbc.Driver` .

`-cstring` The connection string to the database. In our case it should be something similar to `jdbc:mysql://localhost:3306/<database_name>`.

Some interesting optional parameters:

`-c` The command terminator to use. By default it uses the string “go” on a line by itself to determine when to send the string buffer to the database. The semi-colon character “;” may be a better option so you should add “`-c ;`” to the command line.

`-input` The name or the full path of a file to read commands from instead of System.in.

`-query` An optional single query to run instead of interacting with the command line or a file. The query must be between quotation marks. For example, “`-query "select * from client;""`.

`-formatter` There are three formatters that are included.

- **`default`** Output the format in table format (it does not have to be specified). This option supports the following command line options:
  - **`-noheader`** It does not print the header info.
  - **`-spacer`** The character to use for "empty" space. This defaults to the space character.
  - **`-w`** Specifies the maximum field width for a column. By default Jisql shows columns to a maximum width of 2048. By specifying a value for this, Jisql will truncate the output of columns that are wider than this parameter.
  - **`-delimiter`** Specify a single character delimiter for columns.
  - **`-trim`** Trim the spaces from columns.
  - **`-nonull`** Print an empty string instead of the word "NULL" when there is a null value.
  - **`-left`** Left justify the output
  - **`-debug`** Print debugging information about the result set.
- **`csv`** Output the data in CSV format. This option supports the following command line options:
  - **`-delimiter`** Specifies the delimiter to use. By default a comma is used
  - **`-colnames`** Column names are printed as the first line of output. By default they are not included
- **`xml`** Output the data in XML format.

Troubleshooting

Bash:lib/jopt-simple-3.2.jar :Permission denied

After making sure there is no issue with user permissions, this error could be related to the wrong command syntax for Linux systems. E.g Take into account that the jar files in the -classpath parameter need to be combined with colon instead of semicolon when running Jisql command from a Linux shell.

Could not create connection to database server. ErrorCode: 0

This error message occurs when the wrong JDBC driver is used in the Jisql command. You need to download and use the JDBC driver with the correct version for the database you will be connecting to.

References

Jisql Installation Guide
Denodo Tutorials