THOUGHTSPOT

Data Integration Guide

Version 4.4, October 22, 2017

Table of Contents

Introduction to Data Integration

This guide explains how to integrate ThoughtSpot with other data sources for loading data. It also includes information on installing and using the ThoughtSpot clients (ODBC, JDBC, and Informatica).

There are several ways to load data into ThoughtSpot, depending on your goals and where the data is located. You should also consider requirements for recurring loads when planning how best to bring the data into ThoughtSpot.

ⓘ **Note:** ThoughtSpot displays VARCHAR fields using lower case, regardless of what the original casing of your loaded data is.

Here are the options, with information on where to find the documentation for each method:

Method	Description
ThoughtSpot Data Connect	ThoughtSpot Data Connect is a web interface for connecting to databases and applications to move data into ThoughtSpot. You can choose which tables and columns to import and apply data transformations. You can also set up recurring loads. See the ThoughtSpot Data Connect Guide for details.
ThoughtSpot Loader (tsload)	ThoughtSpot Loader is a command line tool to load CSV files into an existing database schema in ThoughtSpot. This is the fastest way to load extremely large amounts of data, and it can be run in parallel. You can also use this method to script recurring loads. See the ThoughtSpot Administrator Guide for details.
User Data Import	Users can upload a spreadsheet through the web interface with User Data Import. This is useful for giving everyone easy access to loading small amounts of their own data. See the ThoughtSpot Administrator Guide for details.
ODBC	ThoughtSpot provides an ODBC (Open Database Connectivity) driver to enable transferring data from your ETL tool into ThoughtSpot.
JDBC	ThoughtSpot provides a JDBC (Java Database Connectivity) driver to enable transferring data from your ETL tool into ThoughtSpot.
Connect to SSIS	You can use the ODBC driver to connect to SSIS and import data into ThoughtSpot. Basic instructions are included in this guide.
Connect to Pentaho	You can use the JDBC driver to connect to Pentaho and import data into ThoughtSpot. Basic instructions are included in this guide.
Informatica Connector	If your company uses Informatica, you can take advantage of the Informatica Connector. This allows ThoughtSpot to become a target database, into which you can load data.

Login credentials for administration

Summary: You need administrative access to perform the actions discussed in this guide.

You can access ThoughtSpot via SSH at the command prompt and from a Web browser.

Administrative access

There are two separate default administrator users, an operating system user that you type in at the Linux shell prompt, and an application user for access through a browser. Make sure you use the correct login and password for the method you are using to log in. Passwords are case sensitive.

Login Type	User	Access Method	Password
OS user	admin	Via a secure socket shell (SSH)	Contact ThoughtSpot to obtain the default password.
Application user	tsadmin	Access through a Web browser	Contact ThoughtSpot to obtain the default password.

SSH to the appliance

To perform basic administration such as checking network connectivity, starting and stopping services, and setting up email, log in remotely as the Linux administrator user "admin". To log in with SSH from any machine, you can use the command shell or a utility like Putty.

In the following procedure, replace `<hostname_or_IP>` with the hostname or IP address of a node in ThoughtSpot. The default SSH port (22) will be used.

1. Log in to a client machine and open a command prompt.
2. Issue the SSH command, specifying the IP address or hostname of the ThoughtSpot instance:

```
ssh admin@<hostname_or_IP>
```

3. Enter the password for the admin user.

Log in to the ThoughtSpot application

To set up and explore your data, access the ThoughtSpot application from a standard Web browser using a username and password.

Before accessing ThoughtSpot, you need:

- The Web address (IP address or server name) for ThoughtSpot.
- A network connection.
- A Web browser.

- A username and password for ThoughtSpot.

Supported Web browsers include:

Browser	Version	Operating System
Google Chrome	20 and above	Windows 7 or greater, Linux, MacOS
Mozilla Firefox	14 and above	Windows 7 or greater, Linux, MacOS
Internet Explorer	11	Windows 7 or greater

To log in to ThoughtSpot from a browser:

1. Open the browser and type in the Web address for ThoughtSpot: `http://<hostname_or_IP>`
2. Enter your username and password and click **Enter Now**.

ThoughtSpot Clients

Summary: Clients help you to load data easily from your ETL tool or another database.

ThoughtSpot provides certified clients to help you load data easily from your ETL tool or another database. These include ODBC and JDBC drivers.

You can obtain the ThoughtSpot client downloads from the Help Center. Always use the version of the ThoughtSpot clients that corresponds with the version of ThoughtSpot that you are running. When upgrading, make sure to upgrade your clients as well.

⚠ **Important:** The ETL tool must add a data transformation step if the source column data type does not exactly match the target's, ThoughtSpot's, column data type. The driver does not do any implicit conversions.

Where to go next

- **About the ODBC Driver**
  You can use the ThoughtSpot ODBC driver to bring data into ThoughtSpot from your ETL tool or database.
- **About the JDBC Driver**
  Java Database Connectivity (JDBC) is a Java standard API that allows applications to interact with databases in a standard manner. ThoughtSpot has JDBC support via a JDBC driver we provide.
- **About Informatica Connector**
  You can use the ThoughtSpot Informatica Cloud connector to read and write data between ThoughtSpot and Informatica. The connector supports INSERT, UPSERT, and READ operations. Once the connecter is downloaded, you can enter your company's ThoughtSpot cluster information and conduct data transfers.

About the ODBC Driver

Summary: Use the ODBC driver to bring data in from your ETL tool or database.

ThoughtSpot comes packaged with an ODBC (Open Database Connectivity) driver, so that you can transfer data between ThoughtSpot and other databases. Basic knowledge of ODBC data source administration is helpful when setting up ODBC.

Supported operating systems for the ODBC driver are:

- Microsoft Windows 32-bit
- Microsoft Windows 64-bit
- Linux 32-bit
- Linux 64-bit
- Solaris Sparc 32-bit
- Solaris Sparc 64-bit

Version Compatibility

To ensure compatibility, always use the ODBC driver with the same version number as the ThoughtSpot instance to which you are connecting.

Supported Data Types

The ODBC driver supports these data types:

- `INT`
- `BIGINT`
- `BOOLEAN`
- `DOUBLE`
- `FLOAT`
- `DATE`
- `TIME`
- `TIMESTAMP`
- `DATETIME`
- `CHAR`
- `VARCHAR`

Source and target data compatibility

By default, ThoughtSpot takes a permissive approach to datatype compatibility between source and target in ODBC. As much as possible ThoughtSpot automatically converts incoming "compatible" input data to the desired ThoughtSpot target datatype. At installation time, your customer support engineer can assist you in configuring a stricter behavior for ODBC.

Following table describes the conversion matrix between SQL datatypes and ThoughtSpot datatypes.

Source SQL Datatypes	BOOL	INT32	INT64	DOUBLE	FLOAT	CHAR	DATE	TIME	DATETIME
SQL_BIT	Y	Y	Y	Y	Y	Y			
SQL_TINYINT	Y	Y	Y	Y	Y	Y			
SQL_SMALLINT	Y	Y	Y	Y	Y	Y			
SQL_INTEGER	Y	Y	Y	Y	Y	Y			
SQL_BIGINT	Y	Y	Y	Y	Y	Y			
SQL_CHAR	Y	Y	Y	Y	Y	Y	Y	Y	Y
SQL_VARCHAR	Y	Y	Y	Y	Y	Y	Y	Y	Y
SQL_LONGVARCHAR	Y	Y	Y	Y	Y	Y	Y	Y	Y
SQL_BINARY						Y			
SQL_VARBINARY						Y			
SQL_LONGVARBINARY						Y			
SQL_DOUBLE	Y	Y	Y	Y	Y	Y			
SQL_REAL	Y	Y	Y	Y	Y	Y			
SQL_FLOAT	Y	Y	Y	Y	Y	Y			
SQL_NUMERIC	Y	Y	Y	Y	Y	Y			
SQL_GUID						Y			
SQL_INTERVAL_MINUTE_TO_SECOND						Y			
SQL_INTERVAL_HOUR_TO_SECOND						Y			
SQL_INTERVAL_HOUR_TO_MINUTE						Y			
SQL_INTERVAL_DAY_TO_SECOND						Y			
SQL_INTERVAL_DAY_TO_MINUTE						Y			

If a conversion is not possible, an error is returned to the client to indicate conversion failure. The ETL tool must add a data transformation step if the source column data type does not exactly match the target's ThoughtSpot column data type. The driver does not do any implicit conversions.

Install the ODBC Driver on Windows

Summary: Use this procedure to obtain the Microsoft Windows ODBC driver and install it.

The ODBC driver for Windows requires Visual C++ Redistributable for Visual Studio 2014. You will be prompted to install it during installation of the driver if it isn't already installed. It is important to note the following about the ODBC login information:

- Database `username`: This is not the machine login username. This is the name of a ThoughtSpot user with administrator permissions.
- Database `password`: This is not the machine login username. This is the ThoughtSpot user password.

To obtain and install the ODBC driver for Windows:

1. Navigate to the Downloads page in the Help Center to download the ODBC driver onto your Windows workstation.
2. Unzip the file you downloaded.
3. Choose the installer for your version of Windows. There are two different Windows ODBC installers included in the file you downloaded.
   - ThoughtSpotODBC (x86).msi for Windows 32-bit
   - ThoughtSpotODBC (x64).msi for Windows 64-bit
4. Double click the `.msi` file you downloaded to start the installation. You will see a security warning.

5. Select `YES` to continue.
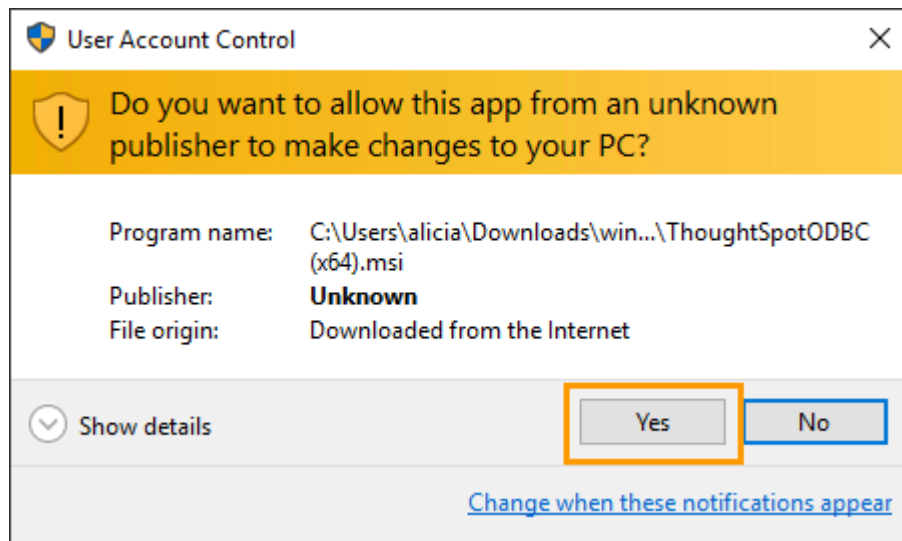
6. Click **Next** to continue.
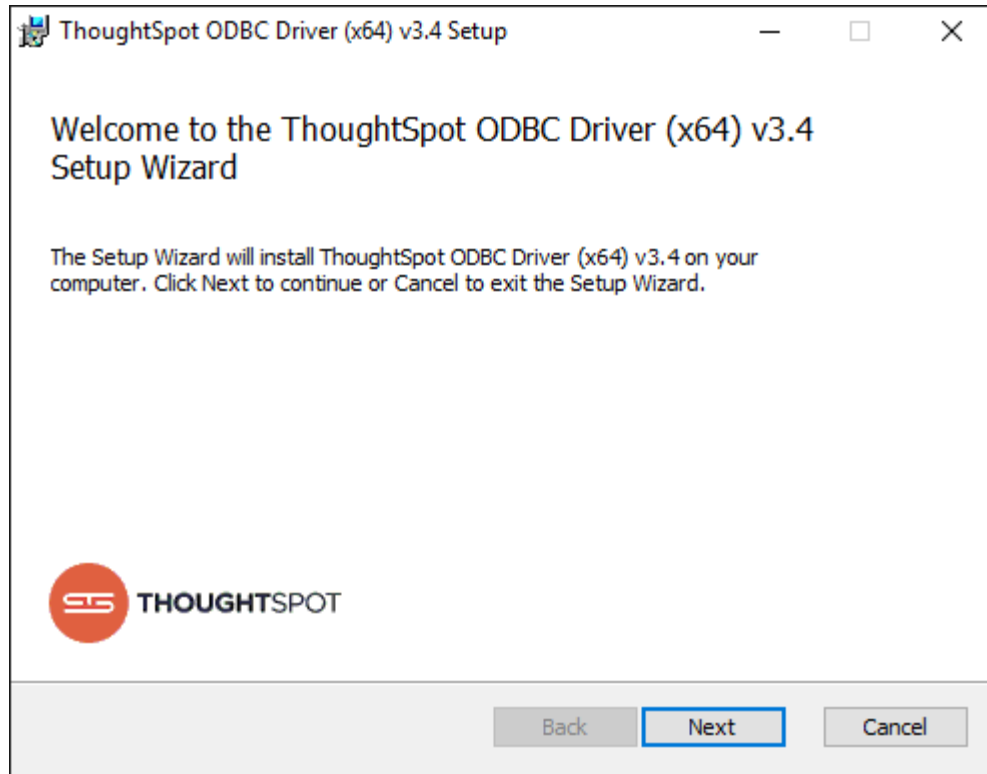
7. Accept the End User License Agreement (EULA), and click **Next**.

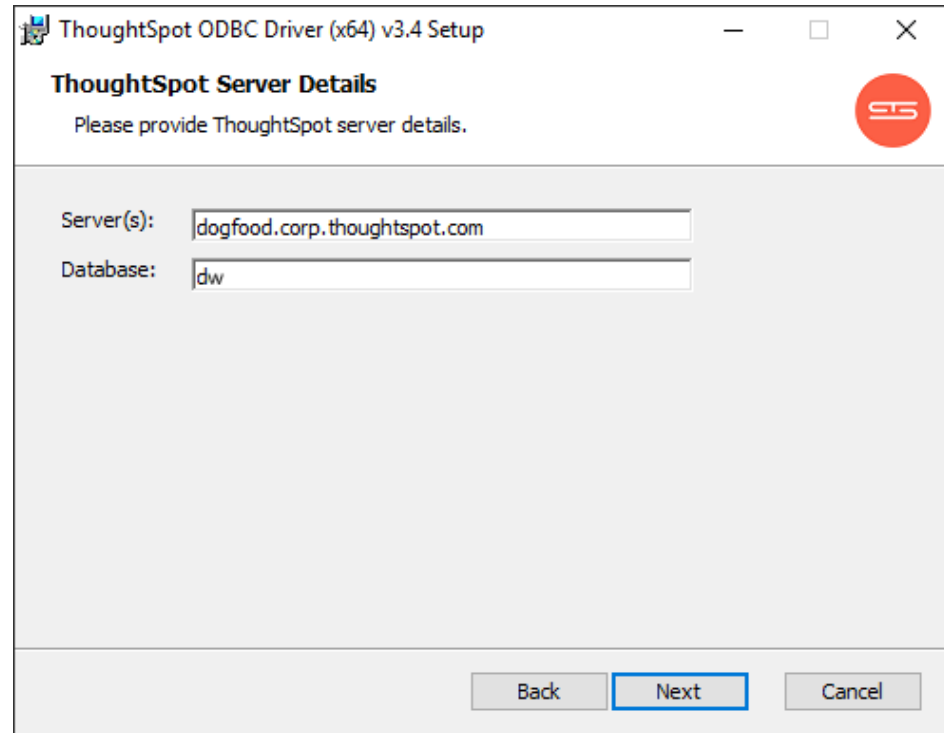8. Specify the destination folder where the driver will be installed.
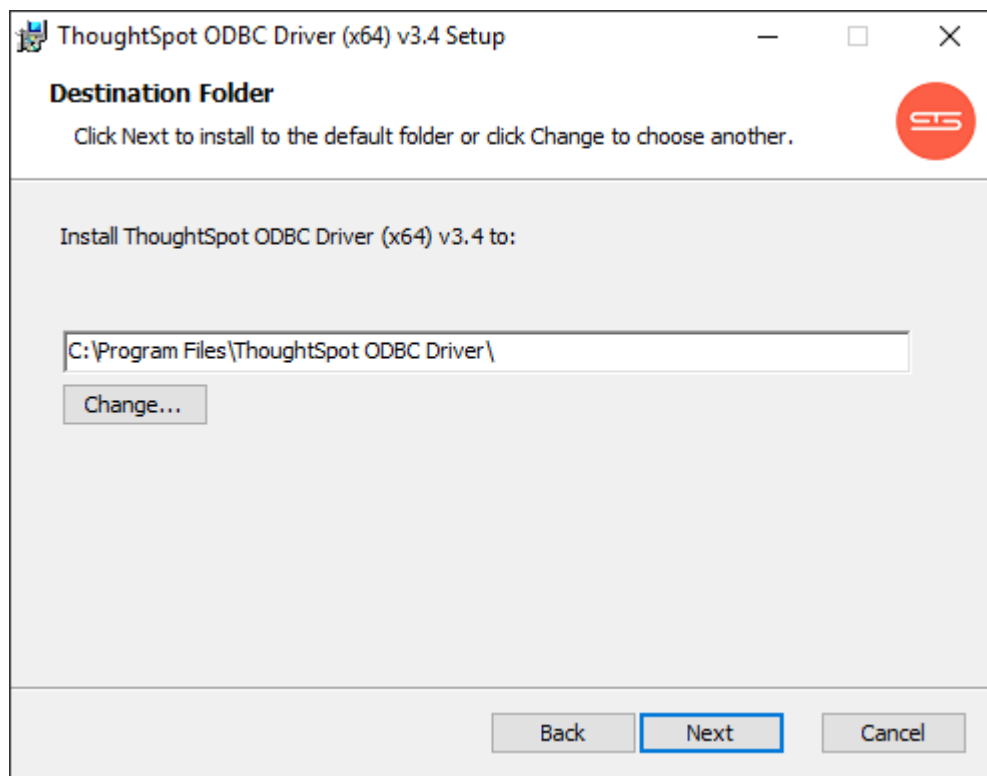
9.   Enter the ThoughtSpot server details, and click **Next**.

- For **Server(s)**, provide a comma separated list of the IP addresses of each node on the ThoughtSpot instance.

  If you need to obtain the IP addresses of the nodes in the cluster, you can run the command `tscli node ls` from the Linux shell on the ThoughtSpot instance.

- For **Database**, optionally specify the database to use. If you skip this entry, you'll need to provide the database each time you connect using ODBC.

10. Confirm that the install can begin by clicking **Install**.

11. You will see a confirmation message when the installation has finished. Click **Finish**.
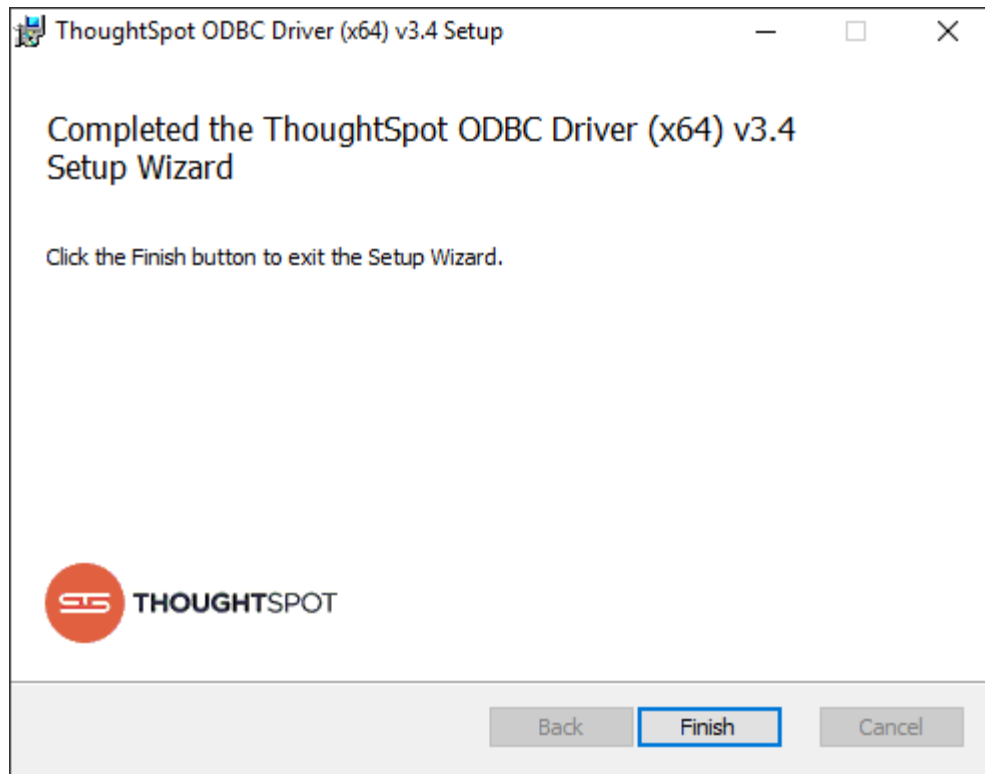
If you need to make changes to the ODBC configuration later, you can Change the ODBC Configuration on Windows. For example, you may want to add a default schema or change the server IP address or the default database. You can also add a new ODBC data source. This capability supports connecting to multiple ThoughtSpot instances.

Change the Windows ODBC Configuration

Summary: Use the ODBC Administrator to change the ODBC configuration.

Once installation is complete, you can use the ODBC Administrator to change the ODBC configuration. For example, you may want to add a default schema or change the server IP address or the default database.

It is recommended to add a default schema. If you don't specify a default schema, you will need to supply it every time you use the ODBC driver. If you aren't using schemas in ThoughtSpot, you should specify the default schema, which is "falcon_default_schema". If you don't supply a default schema, and you don't specify a schema when using the ODBC driver, you will see an error that says the schema could not be found.

To make changes to the ODBC settings on Windows:

1. Launch the **ODBC Administrator**. You can find it in your programs under **ThoughtSpot ODBC Driver**.
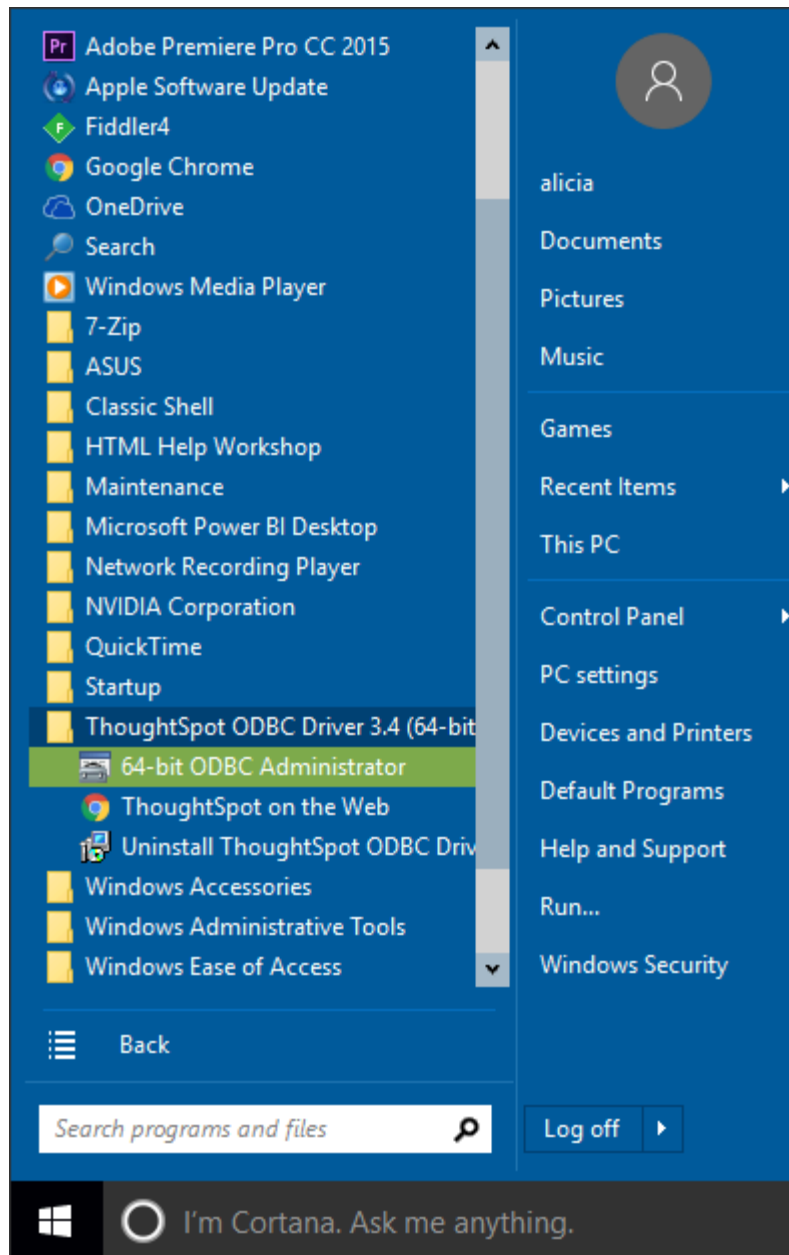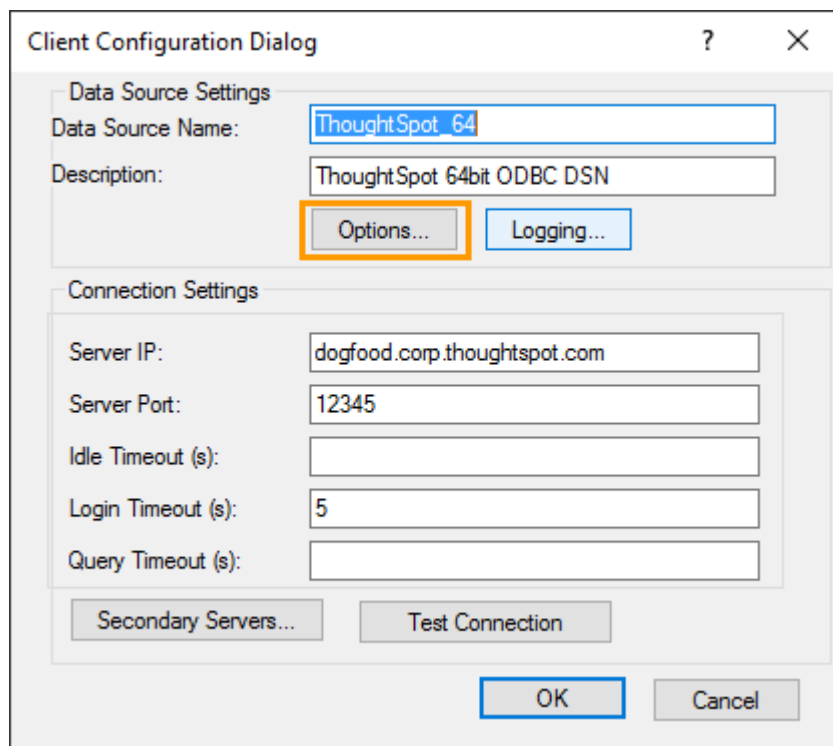
2. Click the **System DSN** tab.

3. Select the data source you want to modify, and click **Configure**.
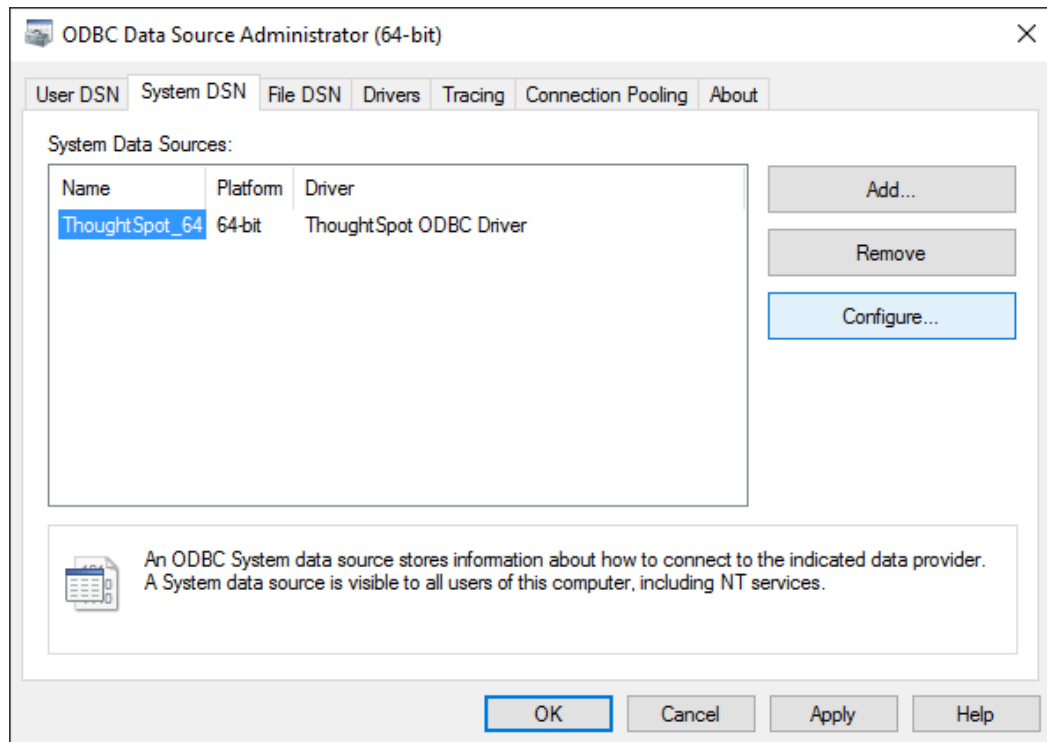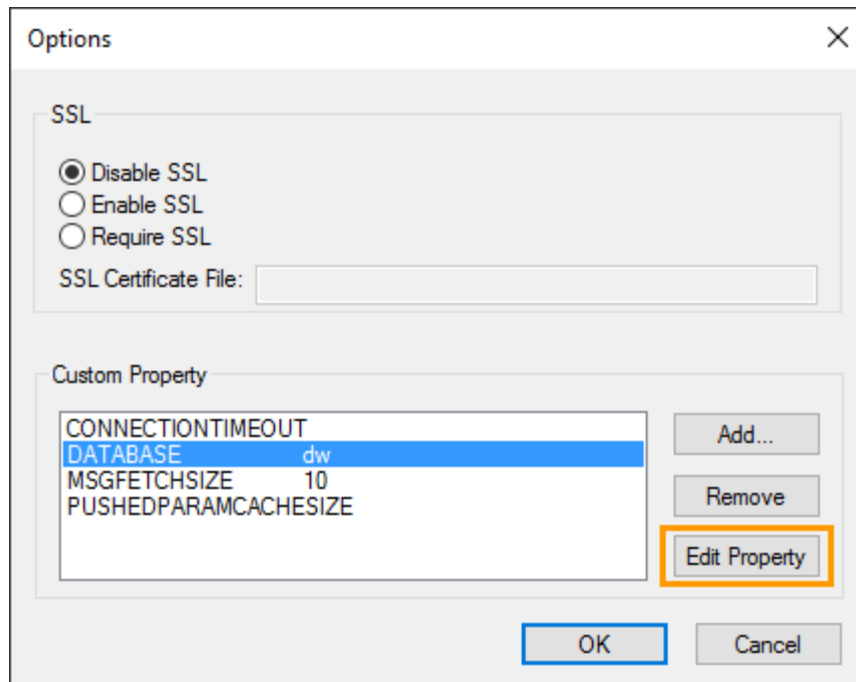
Some properties are exposed through the dialog box, and you can make the settings there.

4. To change or add a custom property, click **Options**.

5. To change a custom property, click **Edit Property** or to add a new custom property, click **Add**.

6. Type in the key (if needed) and add the value and click **OK**.

   You can add a default schema to use by adding a new custom property with the key "SCHEMA". If you don't use custom schema names in ThoughtSpot, use the value "falcon_default_schema". If you add a default schema, that will save you from having to supply the schema every time you use the ODBC connection.
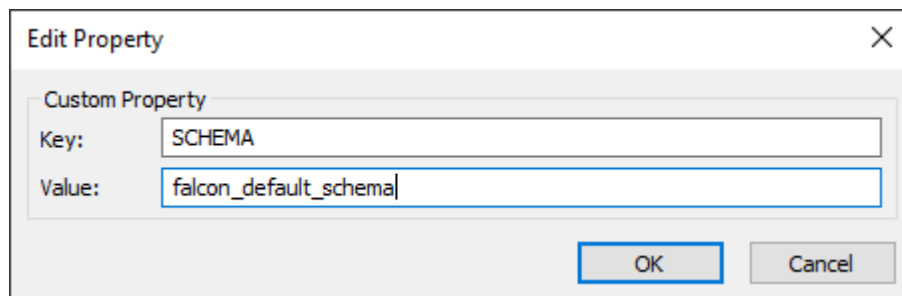
7. Edit any other custom properties you need to change, and click **OK** again.
8. Test the settings by clicking **Test Connection**.

9. If everything is working, click **OK**, to save your settings.

   If not, you may want to enable ODBC logging.

Add a New Data Source to ODBC on Windows

**Summary:** You can add multiple ODBC data sources.

You can add multiple ThoughtSpot data sources to your ODBC configuration. This capability supports connecting to multiple ThoughtSpot instances.

ODBC for Windows needs to have been installed successfully before you can add another ODBC data source.

The main reason for needing to set up multiple ThoughtSpot ODBC data sources is that you have a production cluster and a test or development cluster. The installation procedure for ODBC walks you through the setup of a single data source. Use this procedure if you want to add an additional data source after the installation is successful.
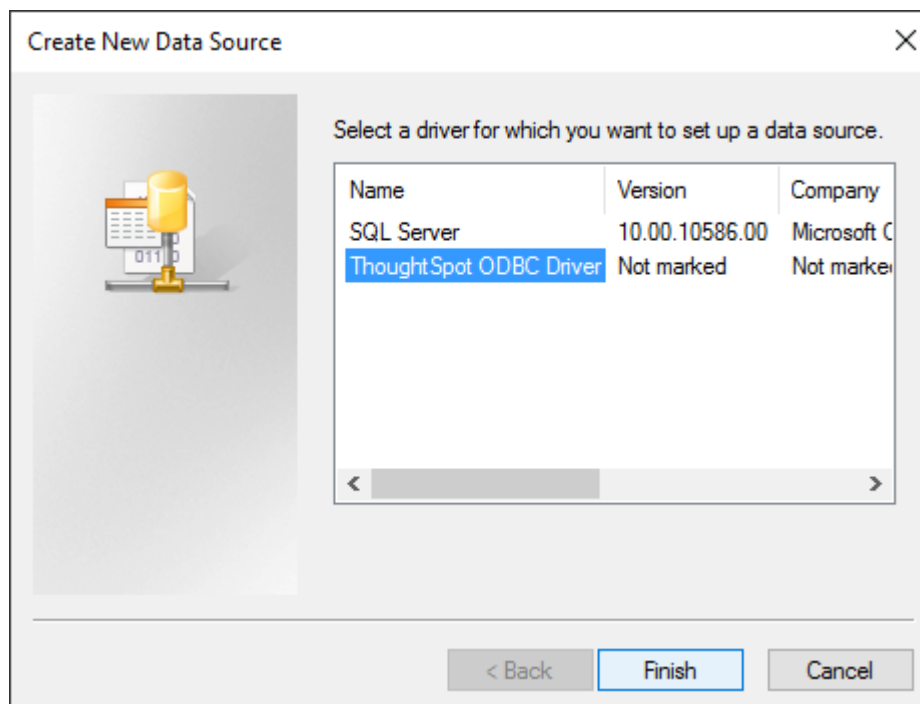
1. Launch the **ODBC Administrator**.

   You can find it in your programs under **ThoughtSpot ODBC Driver**.
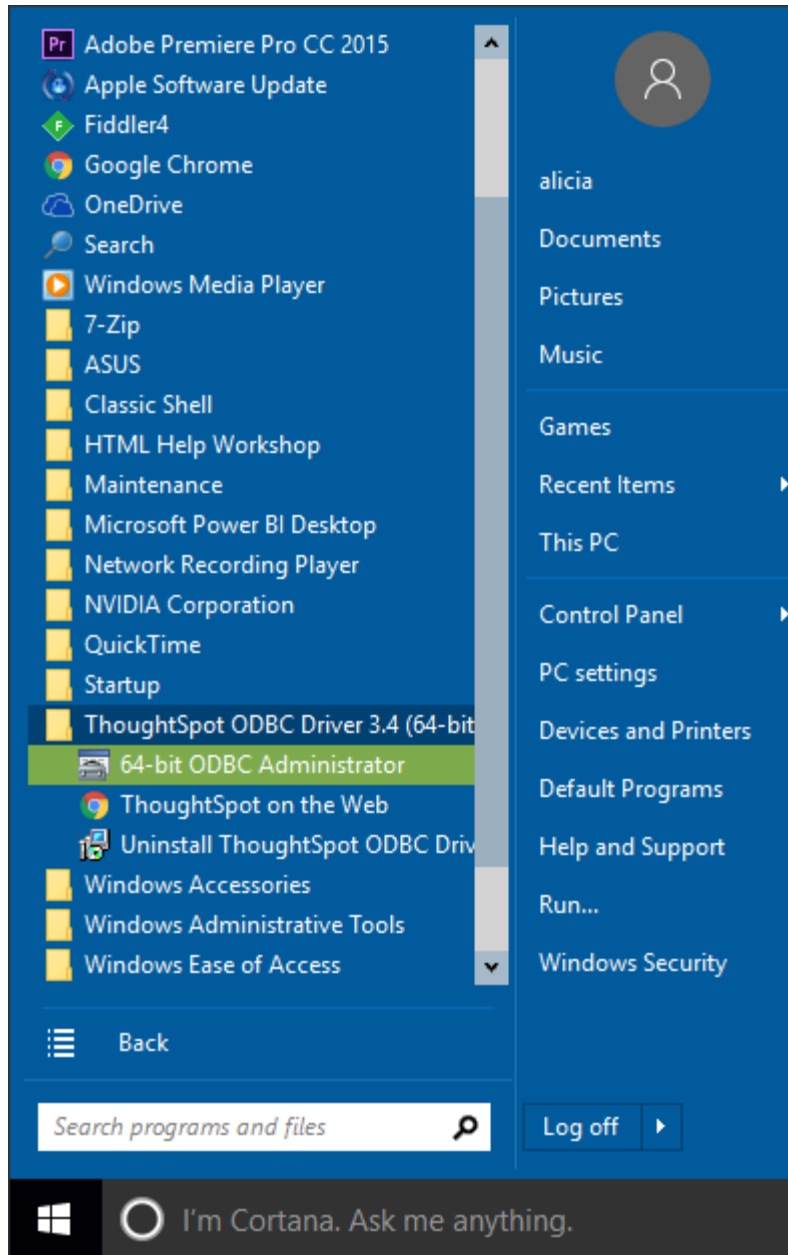
2. Click the **System DSN** tab.

3. Select **Add**.
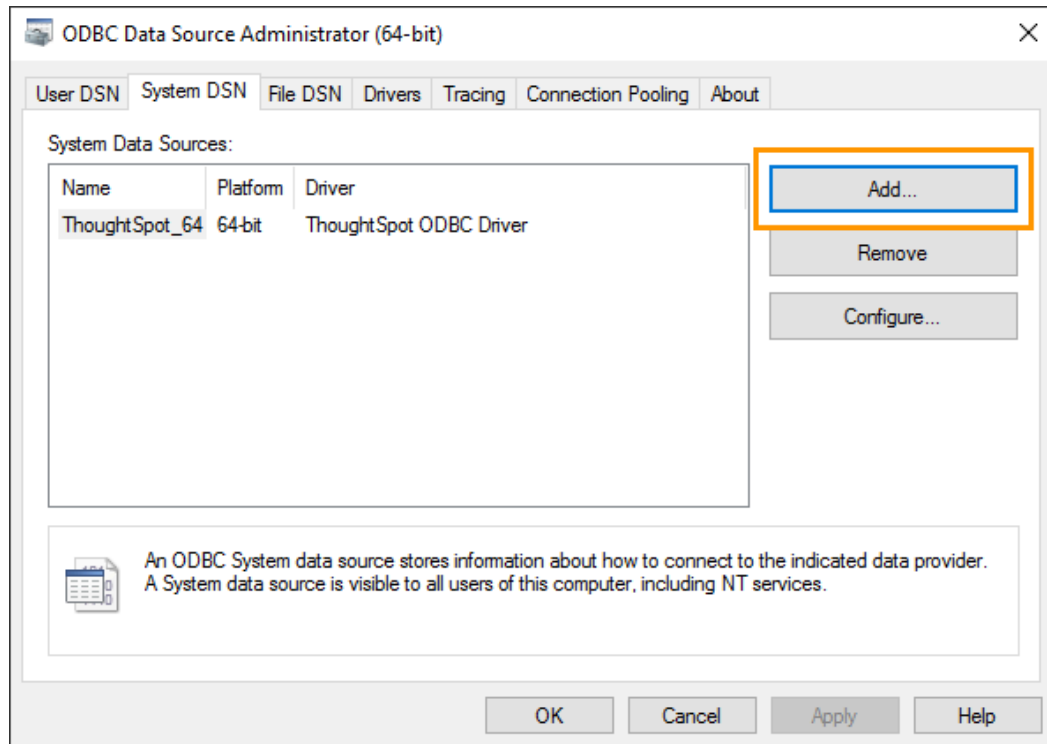
4. Select ThoughtSpot ODBC Driver as the driver to use, and click **Finish**.

5. In the **Client Configuration Dialog**, enter the details about your data source.

    - **Data Source Name**: The name you want to call the data source.
    - **Description**: A description of the data source.
    - **Server IP**: A list of the IP addresses for each node, separated by commas.
    - **Server Port**: 12345

- **Idle Timeout**: Time in seconds after which an idle ODBC connection times out.
- **Login Timeout**: Time in seconds after which a login request times out.
- **Query Timeout**: Time in seconds after which a query times out.

6. To configure custom properties, click **Options**.

7. Click **Add**, to add a new custom property.
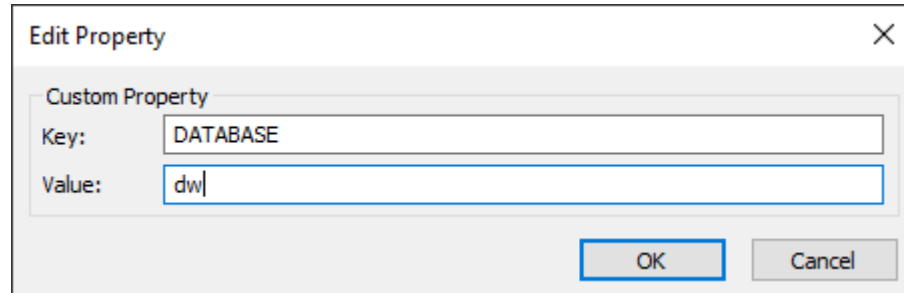
8. Add these properties using the key value pairs shown, clicking **OK** after each entry to save it.

   The key must be defined exactly as it appear here, using all capital letters. You can find other supported

properties in ODBC and JDBC configuration properties.

- **DATABASE**: The default database to connect to.
- **SCHEMA**: Optional. The default schema to connect to.

- **CONNECTIONTIMEOUT**: Optional. Seconds before an idle connection times out.

9. When all the setting have been made, click **Test Connection**.
10. If everything is working, click **OK**, to save your settings. If not, you may want to enable ODBC logging.

Install the ODBC Driver on Linux

Summary: Use this procedure to obtain the Linux ODBC driver and install it.

It is important to note the following about the ODBC login information:

- Database username: This is not the machine login username.This is the name of a ThoughtSpot user with administrator permissions.
- Database password: This is not the machine login password. This is the ThoughtSpot user password.

When you are ready, do the following to install the driver:

1. Create a file on your Linux workstation called /etc/simbaclient.ini and add the following text to it:

   ```
   [Driver]
   ErrorMessagesPath=<path_to_error_messages_directory>
   ```

2. Obtain the ODBC driver:
   a. Navigate to the Downloads page in the Help Center to download the ODBC driver.
   b. Click **ODBC Driver for Linux** to download the file
      `ThoughtSpot_linux_odbc_<version>.tar.gz`.

   c. Unzip and untar the file:

   ```
   gunzip ThoughtSpot_linux_odbc_<version>.tar.gz

   tar -xvf ThoughtSpot_linux_odbc_<version>.tar
   ```

3. Copy the library files from the Lib directory to a safe location on your Linux machine and add the corresponding path to the `LD_LIBRARY_PATH` environment variable.

   For 32-bit users, the library files are located in the directory:

   ```
   /linux/Lib/Linux_x86
   ```

   For 64-bit users, the library is located at:

   ```
   /linux/Lib/Linux_x8664
   ```

4. Open the file `/linux/Setup/odbc.ini` in the editor of your choice.
5. Find the section for the type of Linux you are using (32-bit or 64-bit), by looking at the `Description`.

6. Find the line below it that begins with `ServerList` and replace 127.0.0.1 with a comma separated list of the IP addresses of each node on the ThoughtSpot instance.

   Leave the port number as 12345. The syntax for `ServerList` is:

   ```
   ServerList = <node1_IP> 12345, <node2_IP> 12345 [, <node3_IP> 12345, ...]
   ```

   For example, for the 64-bit ODBC driver:

   ```
   [ThoughtSpot]
   Description = ThoughtSpot 64-bit ODBC Driver
   ```

```
 Driver = ThoughtSpot
 **ServerList = 192.168.2.249 12345, 192.168.2.148 12345, 192.168.2.247 12345**
 Locale = en-US
 ErrorMessagesPath = /usr/local/scaligent/toolchain/local/simba/odbc/linux/
ErrorMessages
 UseSsl = 0
 #SSLCertFile = # Set the SSL certificate file path. The certificate file can
be obtained by extracting the SDK tarball
 #LogLevel = 0 # Set log level to enable debug logging
 #LogPath = # Set the debug log files path
 DATABASE = # Set the default database to connect to
 SCHEMA = # Set the default schema to connect to
```

If you need to obtain the IP addresses of the nodes in the cluster, you can run the command `tscli node ls` from the Linux shell on the ThoughtSpot instance.

7. Open the file `/linux/Setup/odbcinst.ini` in the editor of your choice.

8. Update the the line that starts with `Driver` to have the path to the file `libSimbaClient.so`

    Use the path where you copied the library files. For example, for the 64-bit ODBC driver:

```
 [ThoughtSpot(x64)]
 APILevel           = 1
 ConnectFunctions   = YYY
 Description        = ThoughtSpot 64bit ODBC driver
 **Driver            = /usr/local/scaligent/toolchain/local/simba/odbc/linux/
Bin/Linux_x8664/libSimbaClient.so
 **DriverODBCVer     = 03.52
 SQLLevel           = 1
```

9. Save the file. Now you can test your ODBC connection.

Install the ODBC Driver on Solaris

Summary: Use this procedure to obtain the Solaris ODBC driver and install it.

The Solaris ODBC driver is certified on Solaris Sparc 10. It is important to note the following about the ODBC login information:

- Database username: This is not the machine login username.This is the name of a ThoughtSpot user with administrator permissions.
- Database password: This is not the machine login password. This is the ThoughtSpot user password.

When you are ready, do the following to install the driver:

1. Create a file on your Solaris workstation called `/etc/simbaclient.ini` and add the following text to it:

```
[Driver]
ErrorMessagesPath=<path_to_error_messages_directory>
```

2. Obtain the ODBC driver:
   a. Navigate to the Downloads page in the Help Center to download the ODBC driver.
   b. Click **ODBC Driver for Solaris** to download the file `ThoughtSpot_solaris_sparc_odbc_<version>.tar.gz`.

   c. Unzip and untar the file:

```
gunzip ThoughtSpot_solaris_sparc_odbc_<version>.tar.gz

tar -xvf ThoughtSpot_solaris_sparc_odbc_<version>.tar
```

3. Copy the library files from the Lib directory to a safe location on your Solaris machine.

4. Add the corresponding path to the `LD_LIBRARY_PATH` environment variable.

   For 32-bit users, the library files are located in the directory:

```
/solaris_sparc/Lib/Solaris_sparc_gcc
```

   For 64-bit users, the library is located at:

```
/solaris_sparc/Lib/Solaris_sparc64_gcc
```

5. Open the file `/solaris_sparc/Setup/odbc.ini` in the editor of your choice.

6. Find the section for the type of Linux you are using (32-bit or 64-bit), by looking at the `Description`. Find the line below it that begins with `ServerList`, and replace `127.0.0.1` with a comma separated list of the IP addresses of each node on the ThoughtSpot instance. Leave the port number as `12345`. The syntax for `ServerList` is:

```
ServerList = <node1_IP> 12345, <node2_IP> 12345 [, <node3_IP> 12345, ...]
```

   For example, for the 64-bit ODBC driver:

```
[ThoughtSpot_x64]
```

```
Description         = ThoughtSpot 64-bit ODBC Driver
Driver              = ThoughtSpot(x64)
**ServerList = 192.168.2.249 12345, 192.168.2.148 12345, 192.168.2.247 12345**
Locale              = en-US
UseSsl              = 0
#SSLCertFile        =   # Set the SSL certificate file path. The certificate
fil
e can be obtained by extracting the SDK tarball
#LogLevel           = 0 # Set log level to enable debug logging
#LogPath            =   # Set the debug log files path
DATABASE            =   # Set the default database to connect to
SCHEMA              =   # Set the default schema to connect to
```

If you need to obtain the IP addresses of the nodes in the cluster, you can run the command `tscli node ls` from the Linux shell on the ThoughtSpot instance.

7. Open the file `/solaris_sparc/Setup/odbcinst.ini` in the editor of your choice.

8. Update the the line that starts with `Driver` to have the path to the file `libSimbaClient.so`

   Use the path where you copied the library files. For example, for the 64-bit ODBC driver:

```
[ThoughtSpot(x64)]
APILevel            = 1
ConnectFunctions    = YYY
Description         = ThoughtSpot 64bit ODBC driver
**Driver            = /usr/local/scaligent/toolchain/local/simba/odbc/
solaris_sparc/Lib/Solaris_sparc64_gcc/libSimbaClient.so
**DriverODBCVer     = 03.52
SQLLevel            = 1
```

9. Save the file.

Now you can test your ODBC connection.

Best Practices for Using ODBC

Summary: To successfully use ODBC, following these best practices is recommended.

When developing tools that use the ODBC driver, use these best practices:

- When setting up ODBC for the first time, you should begin by using the ThoughtSpot Loader for the initial data loads. Once those are working properly, then you can switch to ODBC to do incremental loads. This allows you to do more in depth troubleshooting on any initial loading issues.

- After setting up the initial load and ensuring that it works, you may find that there are persistent problems with the incremental load using ODBC. You can enable ODBC logs and send them to ThoughtSpot Support for further investigation.

- You should create the parameterized SQL statement outside of ODBC. Using this method, the SQL statement can be sent to ThoughtSpot in batches by the ODBC driver, so you only have to update the memory itself. ETL tools have this implemented already (end users shouldn't have to actually write the `INSERT` statement). But as a developer, you may be writing code that leverages the ODBC driver, so this tip can help you write your SQL for the best performance with the driver.

- Data can be loaded into a table through multiple parallel connections. This can be achieved by splitting the input data into multiple parts, and loading those individual parts through multiple parallel connections. The parallel loading can be used even while loading to a single table or multiple tables at the same time.

- When doing an incremental data load, note that the same `UPSERT` behavior that occurs via TQL will apply. This means that if you import a row whose primary key matches to an existing row, the existing row will be updated with the new values.

About the JDBC Driver

Summary: Use JDBC to interact with databases in a standard manner.

Java Database Connectivity (JDBC) is a Java standard API that allows applications to interact with databases in a standard manner. ThoughtSpot has JDBC support via a JDBC driver we provide.

When to use JDBC

JDBC can be used whenever you want to connect to ThoughtSpot to insert data programmatically from a Java program or application. You should begin by using the ThoughtSpot Loader for initial data loads and then use JDBC for incremental loads. This is because the ThoughtSpot Loader is generally faster than JDBC. Information on using the ThoughtSpot Loader is available in the ThoughtSpot Administrator Guide.

Version Compatibility

To ensure compatibility, always use the JDBC driver with the same version number as the ThoughtSpot instance to which you are connecting.

Performance Considerations

These are some general recommendations for maximizing the performance of JDBC:

- Insert in batches rather than doing single inserts at a time using the `PreparedStatement::addBatch()` and `PreparedStatement::executeBatch` commands.
- If you need to upload a lot of data, consider running multiple connections with batch inserts in parallel.

ⓘ Note: The ETL tool must add a data transformation step if the source column data type does not exactly match the target's, ThoughtSpot's, column data type. The driver does not do any implicit conversions.

Use the JDBC Driver

Summary: How to configure the JDBC driver.

To use the JDBC driver, include the JDBC library in your path, and provide the connection information. You need this information to configure the JDBC driver:

Information	Description
Driver name	com.simba.client.core.jdbc4.SCJDBC4Driver
Server IP address	The ThoughtSpot appliance URL or IP address. The IP address can be found by going to `http://<server-ip\>:2201/status/service?name=simba_server`
Simba port	The simba port, which is 12345 by default.
Database name	This is not the machine login username. The ThoughtSpot Database name to connect to.
Database username	The name of a ThoughtSpot user with administrator permissions.
Database password	This is not the machine login password. The ThoughtSpot user password.

Install the driver

To obtain the JDBC driver:

1. Log in to the local machine where you want to install the JDBC driver.
2. Click **Here** to download the JDBC driver.
3. Click **JDBC Driver** to download the file `ThoughtSpot_jdbc_<version>.zip`.
4. Move the driver to the desired directory on your local machine.
5. Add the downloaded JDBC driver to your Java class path on the local machine.

Write your application

Using JDBC with ThoughtSpot is the same as using any other JDBC driver with any other database. You need to provide the connection information, create a connection, execute statements, and close the connection.

Specify each of the nodes in the cluster in the connection string, as shown. This enables high availability for JDBC connections. To find out the nodes in the cluster, you can run the command `tscli node ls` from the Linux shell on the ThoughtSpot instance.

The format for the connection is:

```
jdbc:simba://<node1>:12345,<node2>:12345,<node3>:12345[,…];
         LoginTimeout=<seconds>;DATABASE=<db>;SCHEMA=<schema>
```

For example:

```
jdbc:simba://192.168.2.248:12345,192.168.2.249:12345,192.168.2.247:12345;
        LoginTimeout=5;DATABASE=test;SCHEMA=falcon_default_schema
```

As shown, the `DATABASE` and `SCHEMA` parameters need to be in all caps. For the `simba` JDBC driver to work with Spark, the `DATABASE` and `SCHEMA` must be specified in the URL. They cannot be specified as a name/value pair as a map or property. For example:

```
val tssqldf1 = sparkSession.read.format("jdbc").options(Map("url" ->
"jdbc:simba://10.84.78.181:12345;DATABASE=movieratings;SCHEMA=falcon_default_schema",
"driver" ->
"com.simba.client.core.jdbc4.SCJDBC4Driver", "dbtable" -> "Movies", "user" ->
"tsadmin", "password" -> "admin")).load()
```

This `InsertData.java` example shows how to use ThoughtSpot with JDBC. This is an example of a reference JDBC application:

```java
import java.sql.DriverManager;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class InsertData {

  // JDBC class to use.
  private static final String DB_DRIVER = "com.simba.client.core.jdbc4.SCJDBC4Driver";
  // jdbc_example should be an existing database.

  private static final String DB_CONNECTION = "jdbc:simba://192.168.2.129:12345;
     192.168.2.249:12345,192.168.2.247:12345;
     LoginTimeout=5;DATABASE=jdbc_example";SCHEMA=falcon_default_schema

  private static final String TABLE_NAME = "jdbc_example";
  private static final String DB_USER = "<username>";
  private static final String DB_PASSWORD = "<password>";

  // Assuming everything in local directory use:
  //   java -cp .:thoughtspot_jdbc4.jar InsertData
  public static void main(String[] argv) {

    try {
      insertRecordsIntoTable();
    }
    catch (SQLException e) {
      System.out.println(e.getMessage());
    }
  }

  /**
   * Insert some records using batch updates.
   *  Assumes a table exists:  CREATE TABLE "jdbc_example" ( "text" varchar(10) );
   */
  private static void insertRecordsIntoTable() throws SQLException {

    System.out.println("Inserting records.");
    Connection dbConnection = getDBConnection();
    PreparedStatement preparedStatement = null;
    String insertTableSQL = "INSERT INTO falcon_default_schema.jdbc_example (text)
VALUES (?)";

    try {
      preparedStatement = dbConnection.prepareStatement(insertTableSQL);

      // Create multiple statements and add to a batch update.
      for (int cnt = 1; cnt <= 10; cnt++) {
```

```
          preparedStatement.setString(1, "some string " + cnt);
          preparedStatement.addBatch();
          System.out.println("Record " + cnt + " was added to the batch!");
        }
        preparedStatement.executeBatch();  // For large numbers of records, recommend
doing sets of executeBatch commands.
        System.out.println("Records committed");

    }
    catch (SQLException sqle) {
      sqle.printStackTrace();
    }
    finally {

      if (preparedStatement != null) {
        preparedStatement.close();
      }
      if (dbConnection != null) {
        dbConnection.close();
      }
    }
  }

  /** Create a connection to the database. */
  private static Connection getDBConnection() {
    Connection dbConnection = null;
    try {
      Class.forName(DB_DRIVER);
    }
    catch (ClassNotFoundException e) {
      System.out.println(e.getMessage());
    }
    try {
      dbConnection = DriverManager.getConnection(DB_CONNECTION, DB_USER,DB_PASSWORD);
      return dbConnection;
    }
    catch (SQLException sqle) {
      System.out.println(sqle.getMessage());
    }

    return dbConnection;
  }

}
```

About Informatica Connector

Summary: Use the ThoughtSpot Informatica Cloud connector to read and write data.

You can use the ThoughtSpot Informatica Cloud connector to read and write data between ThoughtSpot and Informatica. The connector supports INSERT, UPSERT, and READ operations. Once the connecter is downloaded, you can enter your company's ThoughtSpot cluster information and conduct data transfers.

The Cloud connector can be found through the Informatica Marketplace. Once installed, you can configure a connection between ThoughtSpot and Informatica.
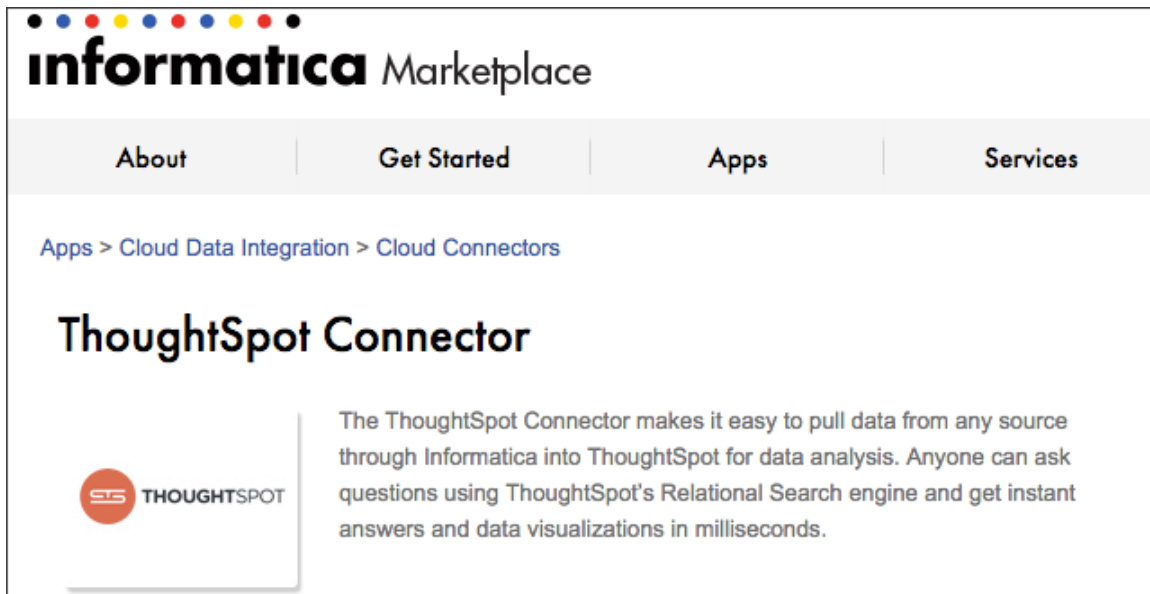
ODBC Data Source Administrator

Summary: Use ODBC to modify log in options and troubleshoot ODBC issues.

The ODBC Data Source Administrator can be used to modify log in options and troubleshoot ODBC issues.

Logging Options

In the ODBC Data Source Administrator, you can specify the log verbosity in the Logging Options. This is done to debug connectivity or failures from the client side.
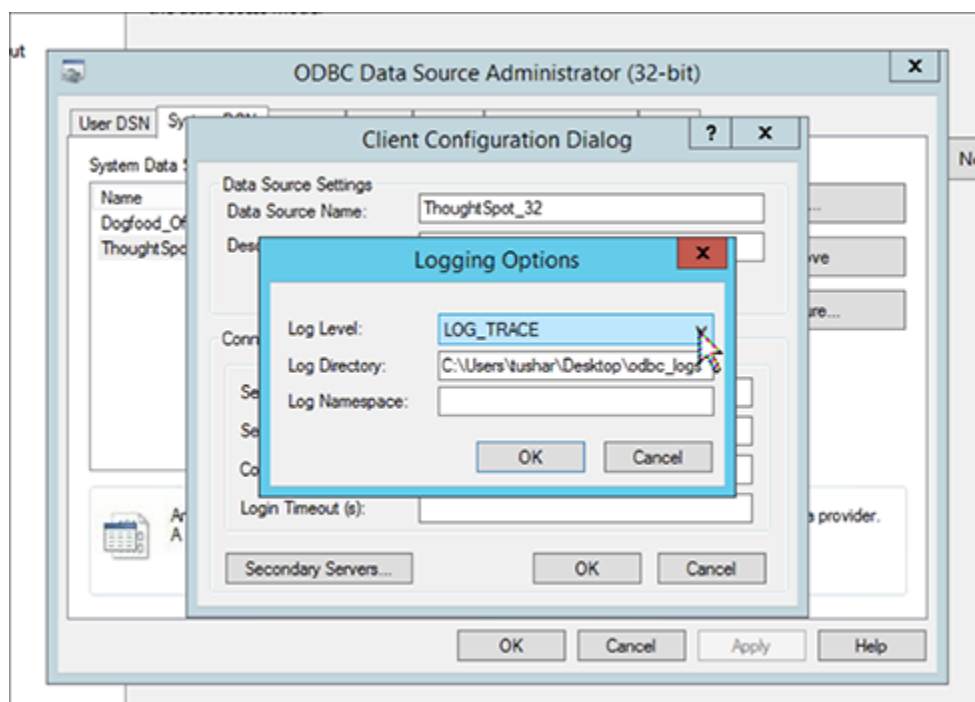
Schema Property

You can provide a schema using the Edit Property. If you do not do this, our system will look in all of the schemas.

ODBC Tracing

Windows shows ODBC specific tracing in the ODBC Data Source Administrator Tracing tab. You can start tracing there by clicking Start Tracing Now. This logs every ODBC call from this system, and prints the input and output for the call.
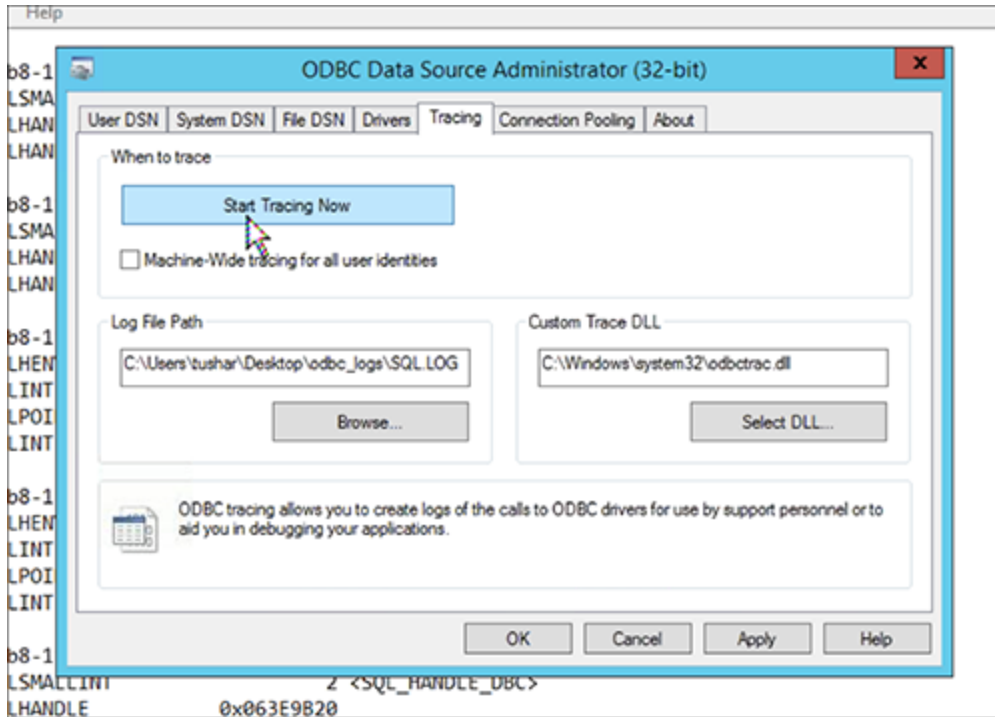
Although this is lower level information, it can still be helpful in troubleshooting. When you are not sure if it is our driver or the tool causing an issue, doing this trace will help narrow the inquiry.
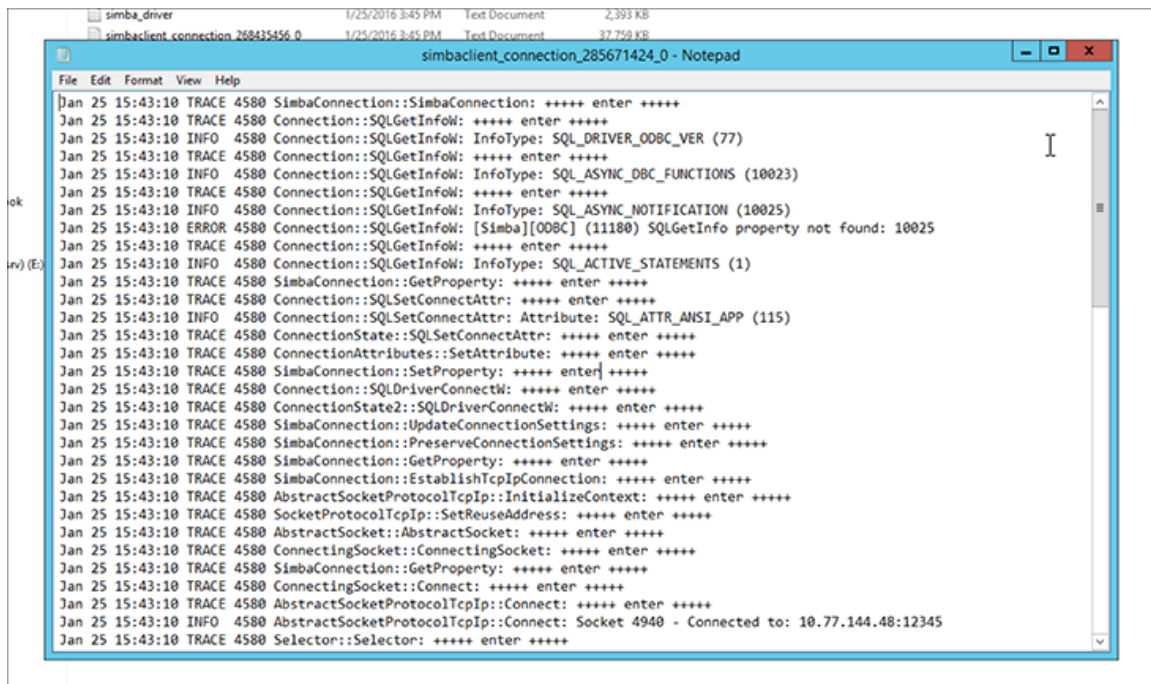
If you start or stop tracing, make sure you do not have the SSIS client open. Close it, change the trace, and reopen.

Set up the ODBC Driver for SSIS

Summary: Use SSIS to set up the ODBC Driver.

Use SSIS to set up the ODBC Driver by creating a connection manager. This manager is used to create a connection between your OLE DB Source and the ODBC Destination.

On Windows 64-bit, you have to install both the 32-bit and 64-bit ThoughtSpot ODBC drivers. In addition, they must be named the same, such as ThoughtSpot. By default they are named ThoughtSpot-32 and ThoughtSpot-64. This is required because the 64-bit SSIS shows a list of 32-bit ODBC drivers when you configure an ODBC target. However, it executes the 64-bit driver. If the drivers aren't named the same, then you'll get an error saying the driver doesn't exist.

Set up the driver

To set up the ODBC driver using SSIS:

1. Open your SQL Server visual development tool that is based on Microsoft Visual Studio.
2. Select **OLE DB Source**, and click **New**.
3. Here you must add the server by name from the machine accessible list.

4. Enter the authentication information: db name, user name, password, and test connection.

   You can add the UID and password by clicking on **Options**.

5. Click **File** and select **New**, then **Project**.
6. Select the **Integration Services** tab under **Installed > Templates > Business Intelligence**.

7. Enter a name in the **Name** field and click **OK**.
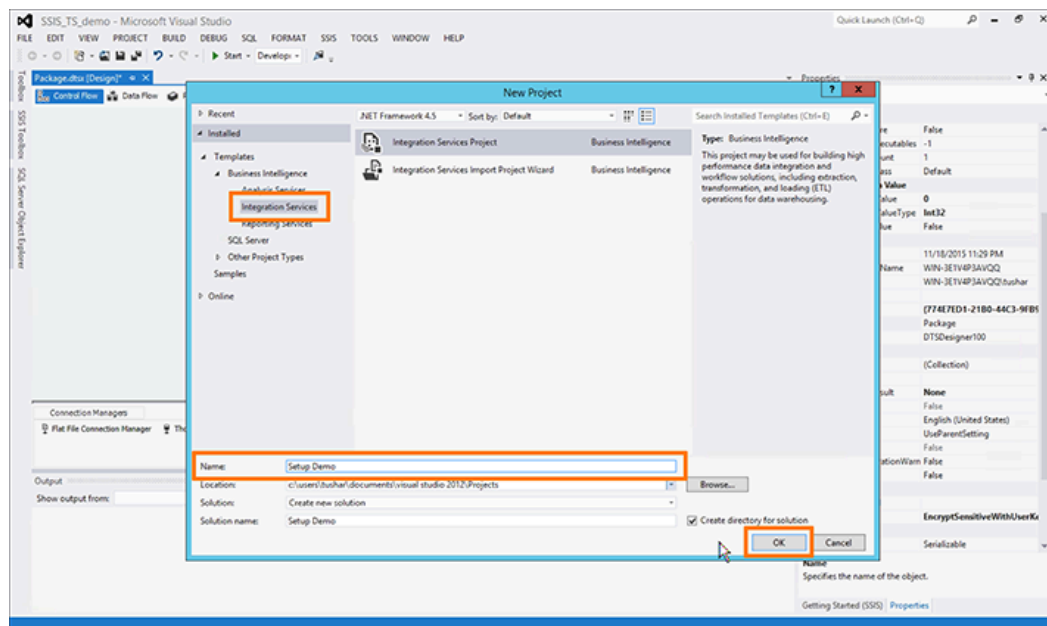
8. Select the **SSIS Toolbox** tab on the left hand side of the platform, and drag and drop **Data Flow Task** to the main window.
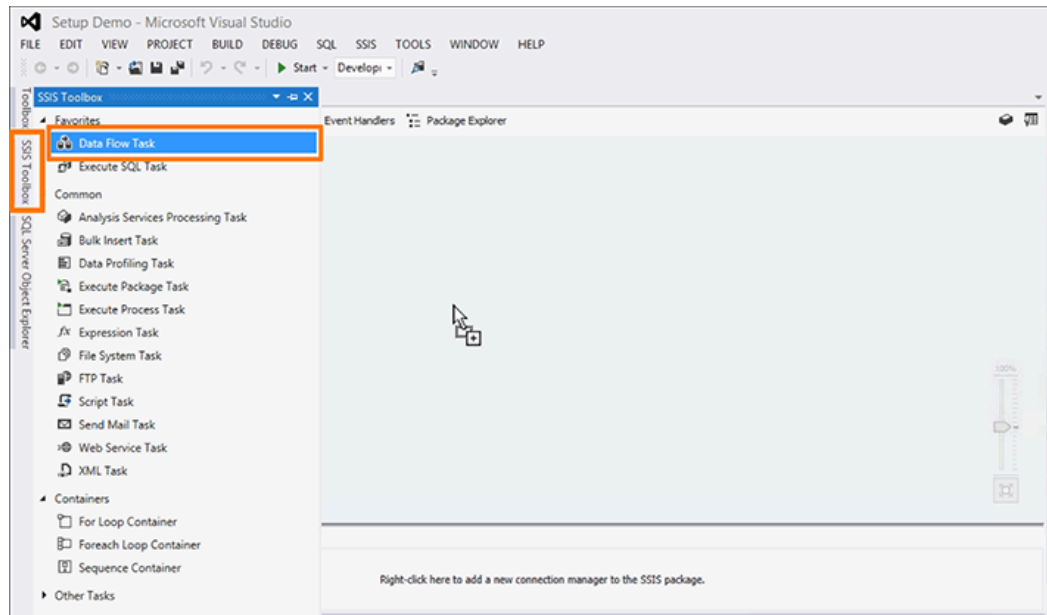
9. Double click the **Data Flow Task** icon when it appears in the center of the page.
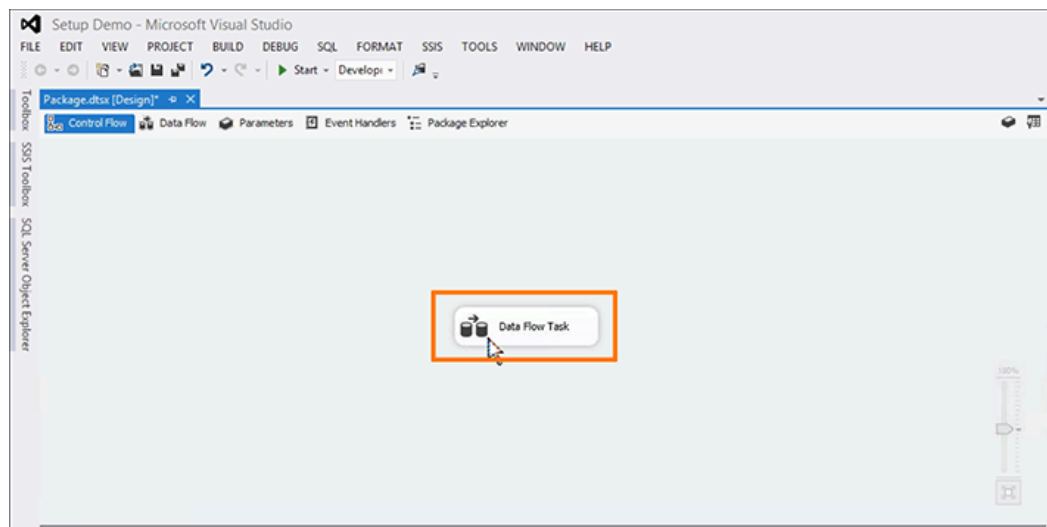
10. Navigate back to the **SSIS Toolbox** tab. You now want to create sources and destinations.

Create sources and destinations

1. Under **Other Sources**, find **OLE DB Source** and drag and drop it to the main window.
2. Double click the **OLE DB Source** icon when it appears in the center of the page to open the OLE DB Source Editor.
3. Select a new OLE DB connection manager by clicking **New**.

4. In the Configure OLE DB Connection Manager window, select your **Data connection** and click **OK**.
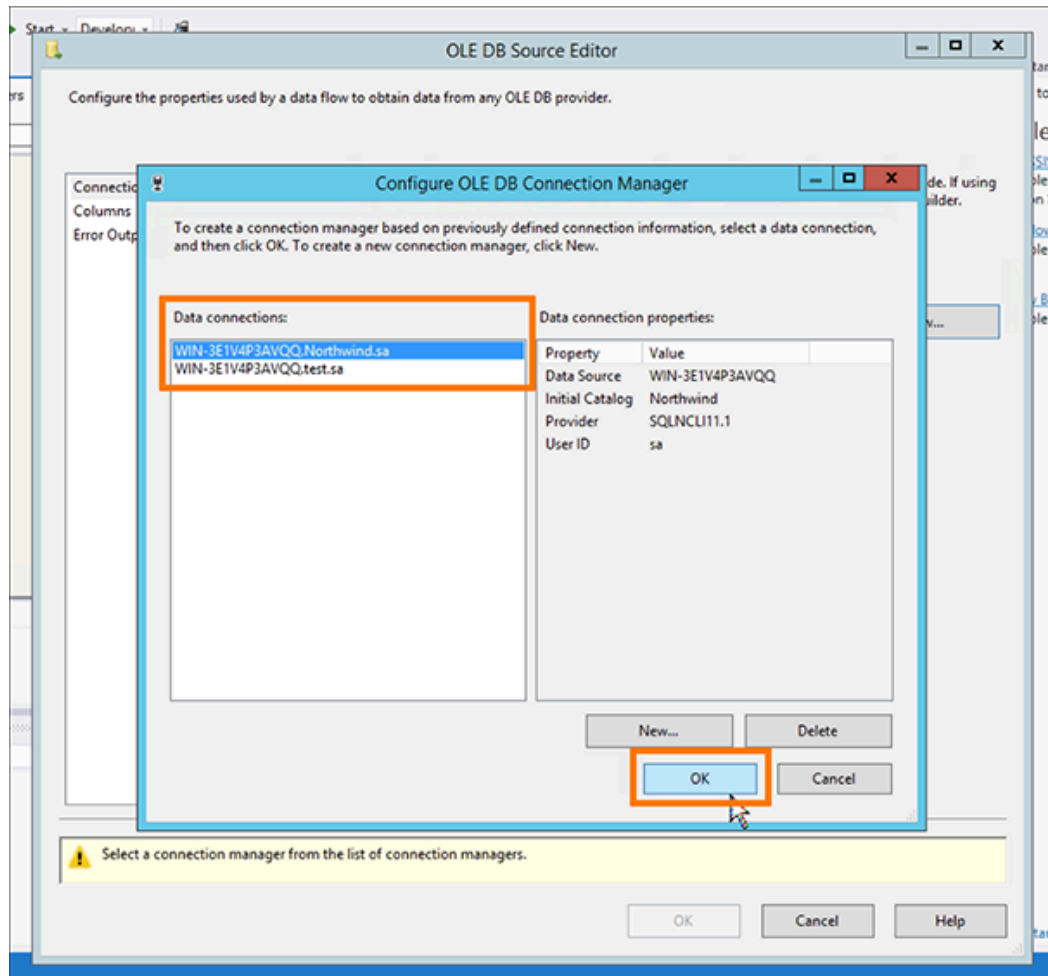
If you do not see your data connection, you will have to create a new one in the Connection Manager by clicking **New**.

5. Back in the OLE DB Source Editor, select the **Name of the table or the view**, and click **OK**.
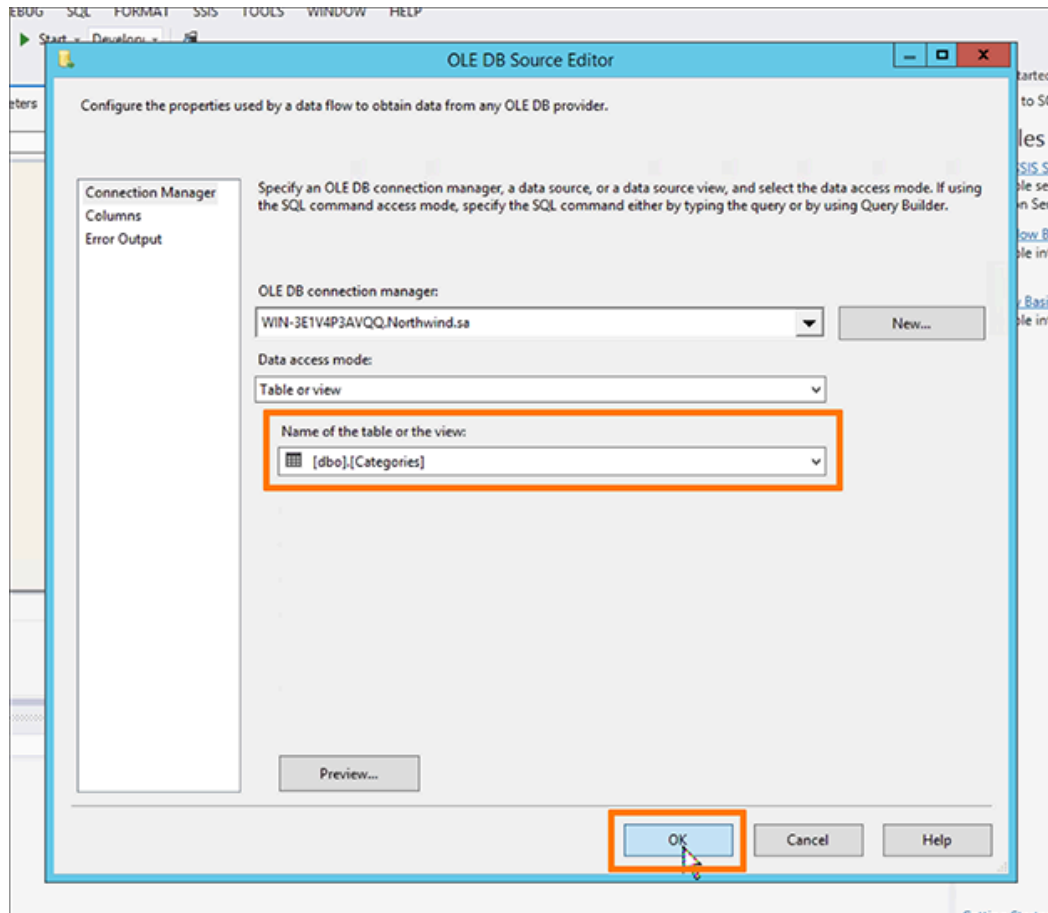
6. Select the table, and see what columns are in it.

    In this example, a single column, `c1`, is selected.
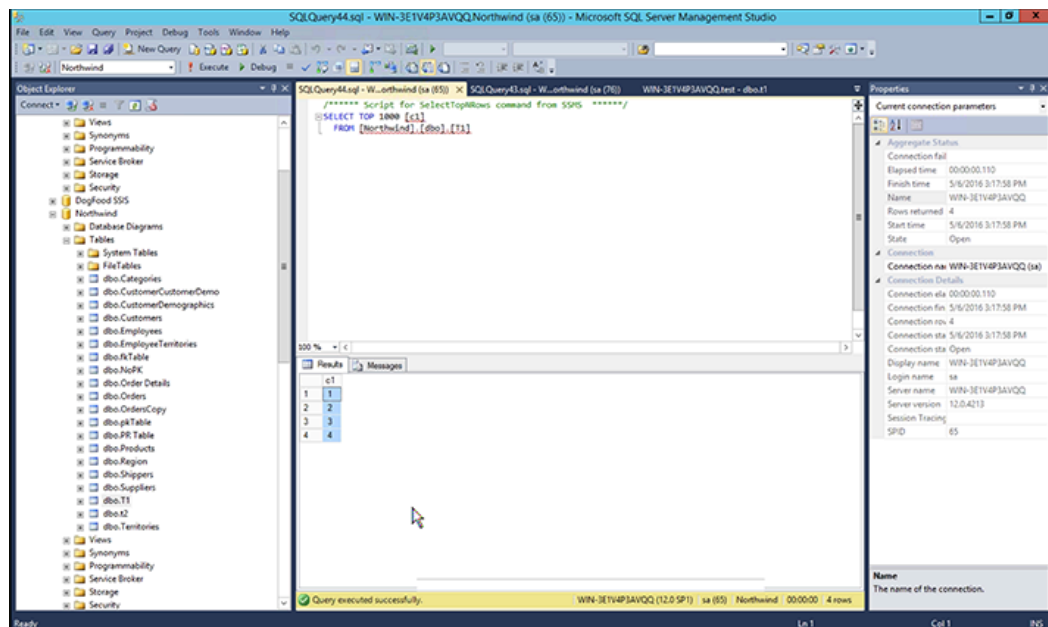
Configure the ODBC Data Source Administrator

The ODBC Data Source Administrator has to be configured to connect to ThoughtSpot and bring the table in.

1. Search for and open your **ODBC Data Sources (32-bit)** program.
2. Click the **System DSN** tab and select **ThoughtSpot_32**.

3. Click **Configure**.
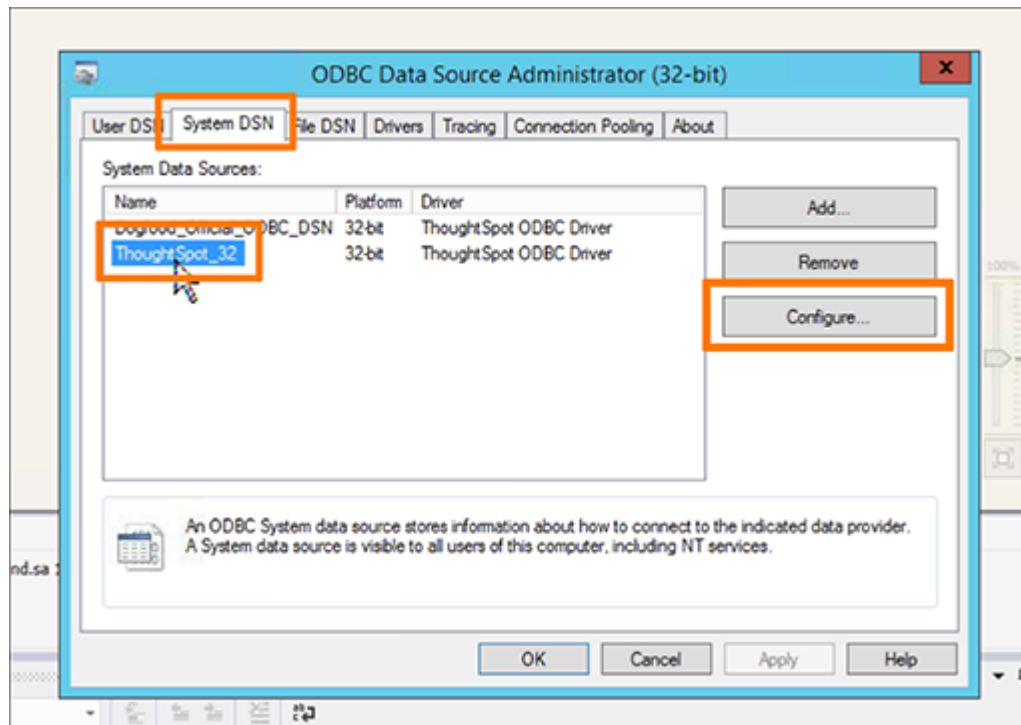
4. In the Client Configuration Dialog, enter the **Server IP** and **Server Port**.

   Any node IP that has Simba server running on it should work. You can provide multiple IPs (using the **Secondary Servers** dialog) so that it will find the one that the Simba server is running on.

5. Click **OK** twice to close the Client Configuration Dialog and the ODBC Data Source Administrator.
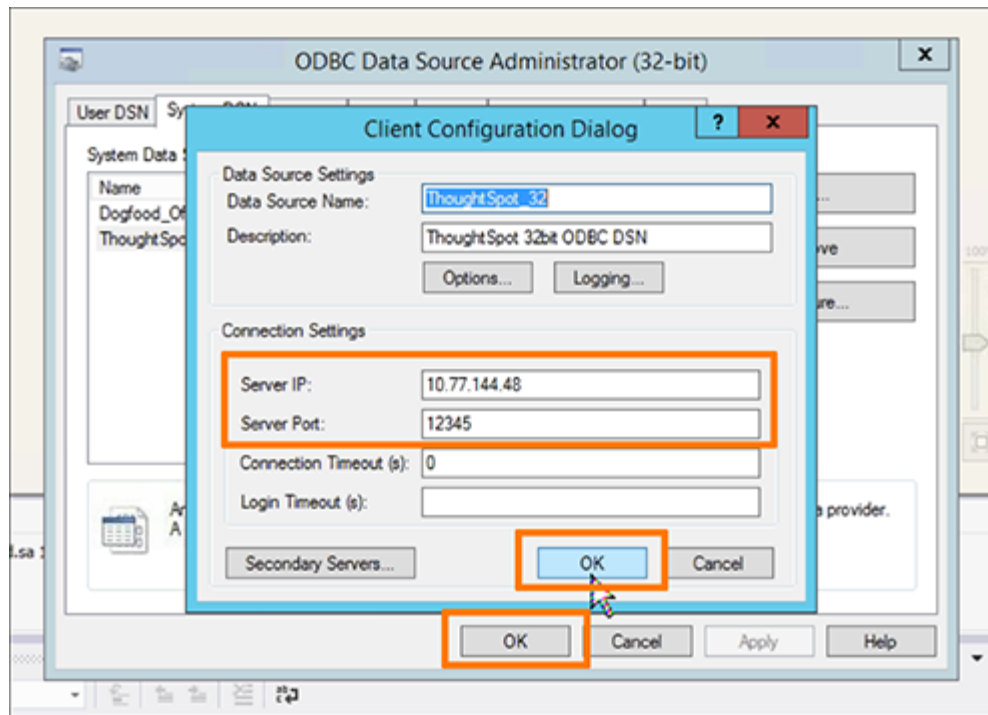
Create a file to take the feed

Now that you have set up your source, create the empty table in ThoughtSpot to take this feed. SSIS does not allow you to create the table in ThoughtSpot. You have to do this first in TQL. In Pentaho, it will create the table in ThoughtSpot, but not in SSIS.

Create the ODBC Destination. Use the one you created and named in the ODBC Data Source Administrator.

1. In the **SSIS Toolbox** tab, under **Other Destinations**, drag and drop **ODBC Destination** to the main window.
2. Drag the **blue arrow** to connect the OLE DB Source icon to the ODBC Destination icon.

3. Double click the **ODBC Destination** icon.
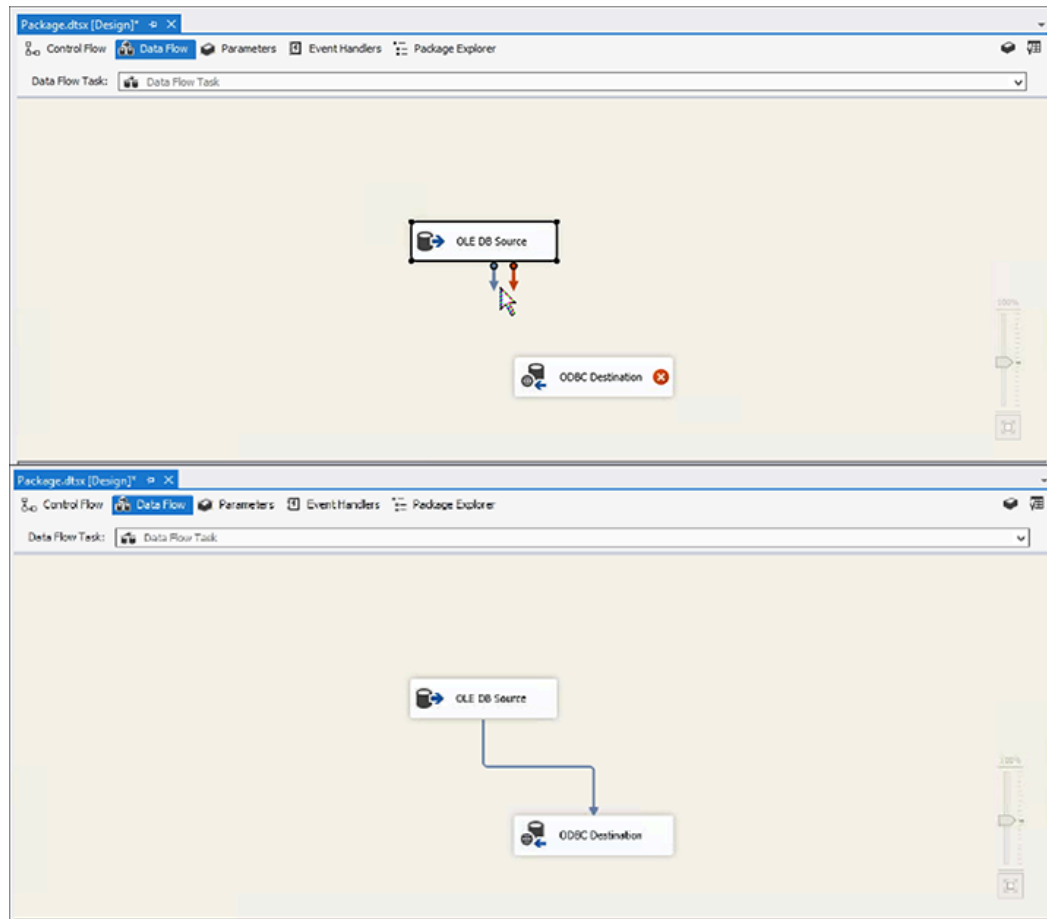
4. Use ODBC Destination to set the **Batch size** for the connection in the Connection Manager tab. You can set the size to be up to 10,000.
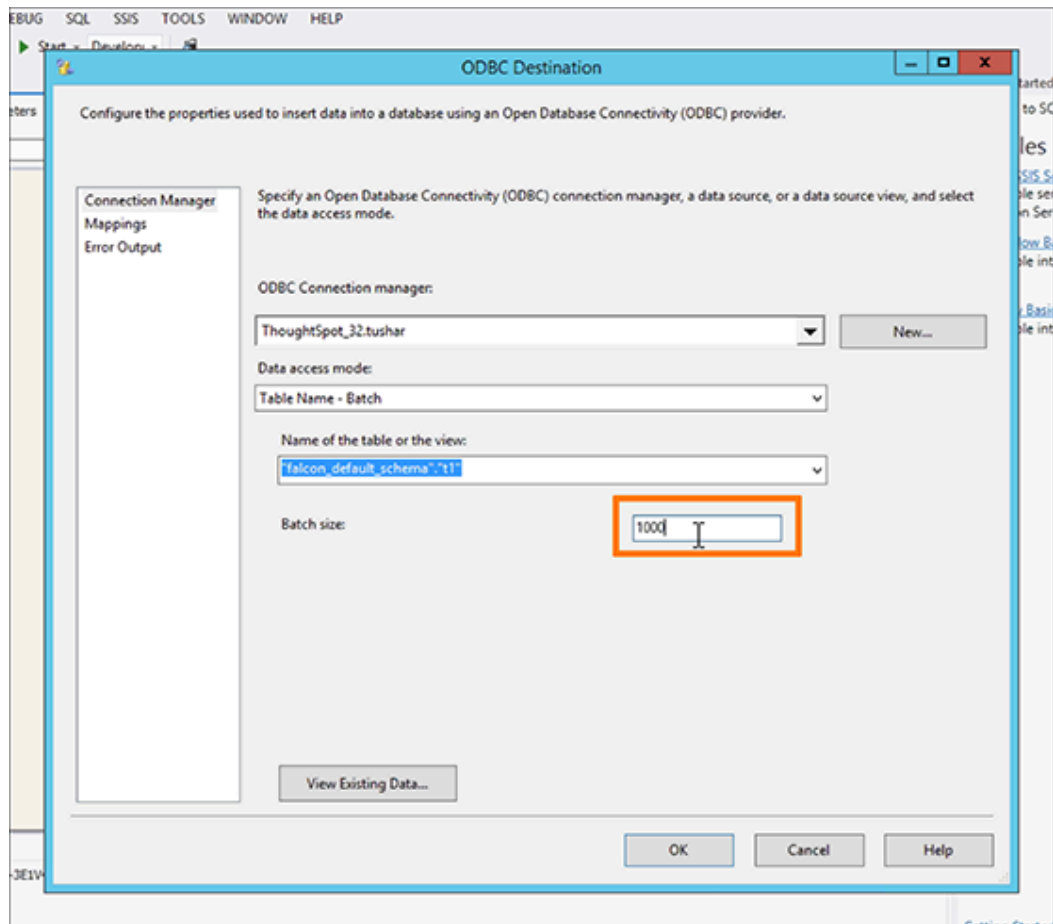
If the load fails, the entire batch will be lost, and you will have to start that load over again.

5. Set the **Transaction Size** to match the total number of rows that are expected to be loaded in the load cycle.

    Your transaction size can be quite large—even spanning a million rows. However, too many small batches can leave the cluster in a rough state. This is because each batch acts as a separate transaction and creates a separate commit. Too many of these will slow down our system since each transaction creates a "data version" in our system. In Pentaho, the transaction size setting is called Commit Size.

6. Set the **Transaction Option** attribute of the Data Flow Task to **Supported**.

7. In the **Mappings** tab, validate the mapping or change it.

    You can have different column names in each database if you map them. Of course, they must be of the same or compatible datatype.
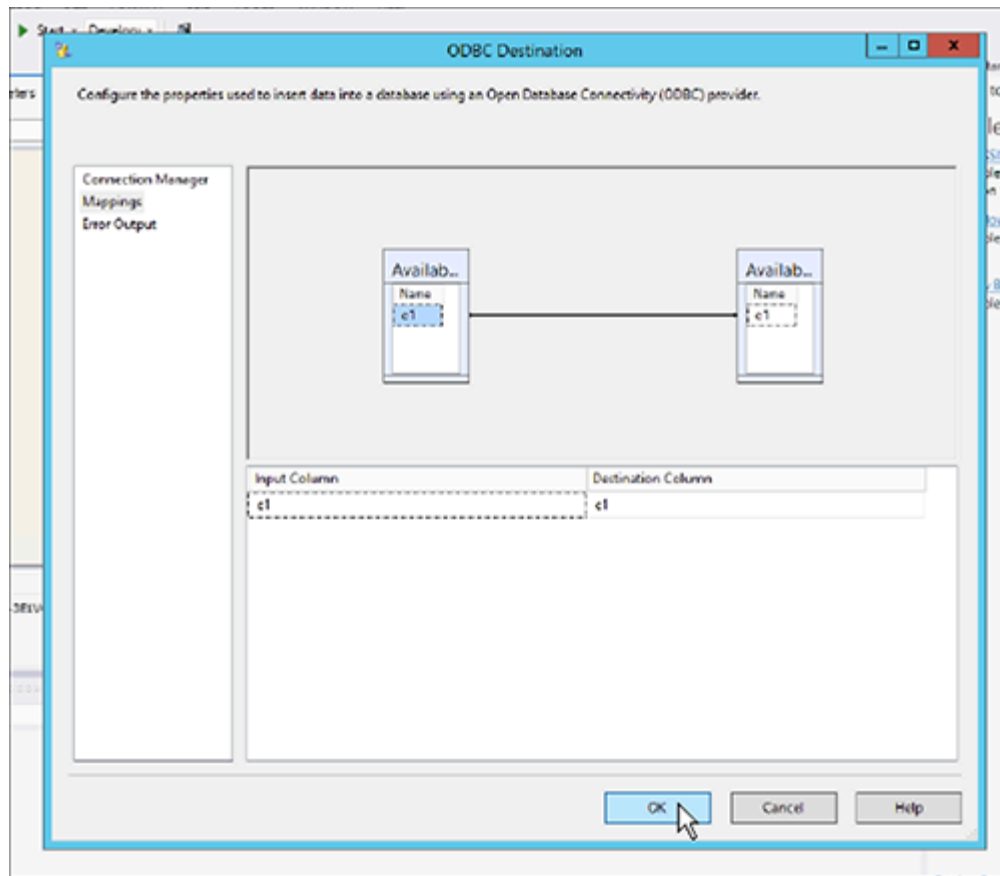
8. Start the import job by clicking the **Start** button.

You should see an animation indicating that the data is transferring over. When the import is complete, the number of successfully transferred rows is displayed.
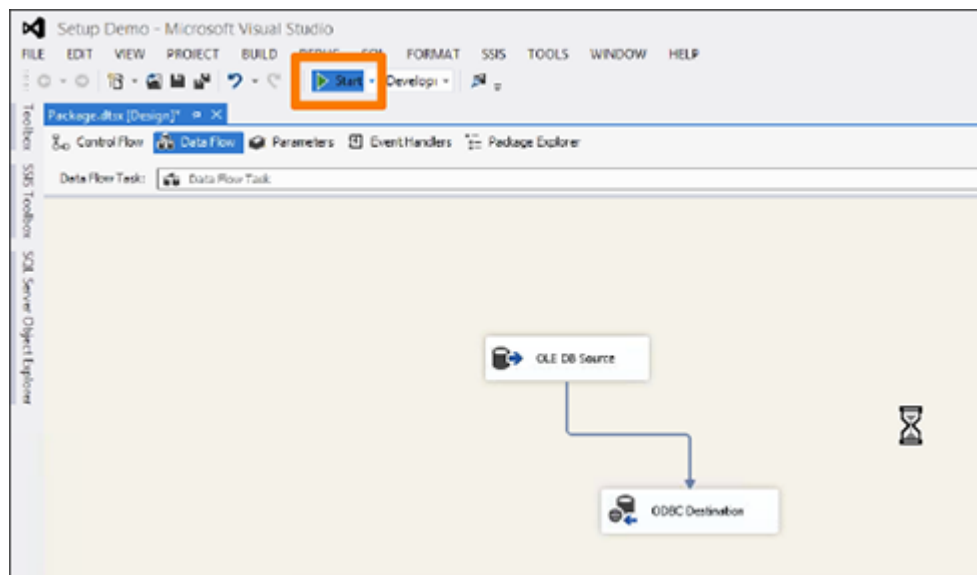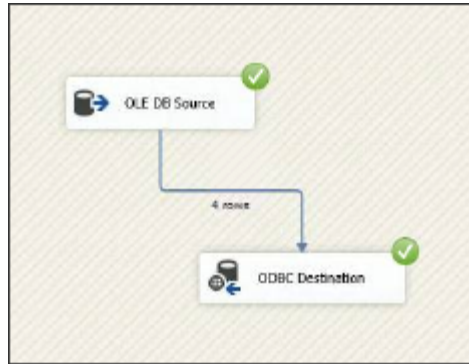
You can validate the import using TQL or in the **DATA** screen.

About Pentaho

You can use the Pentaho Data Integration (PDI) to create a JDBC connection. The Pentaho Data Integration (PDI) suite is a comprehensive data integration and business analytics platform. You can use it to create a JDBC connection to ThoughtSpot.

PDI consists of a core data integration (ETL) engine and GUI applications that allow you to define data integration jobs and transformations. Through Pentaho, we primarily use the JDBC driver to set up a connection. The process is not as complicated as with SSIS, and is much more lenient.

Community and enterprise editions of PDI are available. Using the community edition is sufficient, though you may use the enterprise edition, which is subscription based, and therefore contains extra features and provides technical support.

Set up the JDBC Driver for Pentaho

Summary: JDBC to connect to the Falcon Simba server from Pentaho.

Use JDBC to connect to the Falcon Simba server from Pentaho. The connection will be made between a new Falcon Table Input and Output objects.

Download the required files

Before starting the Pentaho Data Integration (PDI) client and creating the connection, ensure that the Simba JDBC client libraries are present in the Pentaho client/server machines. This will ensure that they can be picked up at runtime. Please copy the `SimbaJDBCClient4.jar` file or the `thoughtspot_jdbc4.jar` file to the following directories:

- `<Pentaho_install_dir>/server/data-integration-server/tomcat/webapps/pentaho-di/WED-INF/lib/`
- `<Pentaho_install_dir>/design-tools/data-integration/lib/`
- `<Pentaho_install_dir>/server/data-integration-server/tomcat/lib/`
- `<Pentaho_install_dir>/design-tools/data-integration/plugins/spoon/agile-bi/lib/`

You can download these files from the Help Center.

Set up the driver

In this example, we are using Spoon, the graphical transformation and job designer associated with the PDI suite. It is also known as the Kettle project. Therefore, the screenshots will reflect this client version.

To set up the JDBC driver using Pentaho:

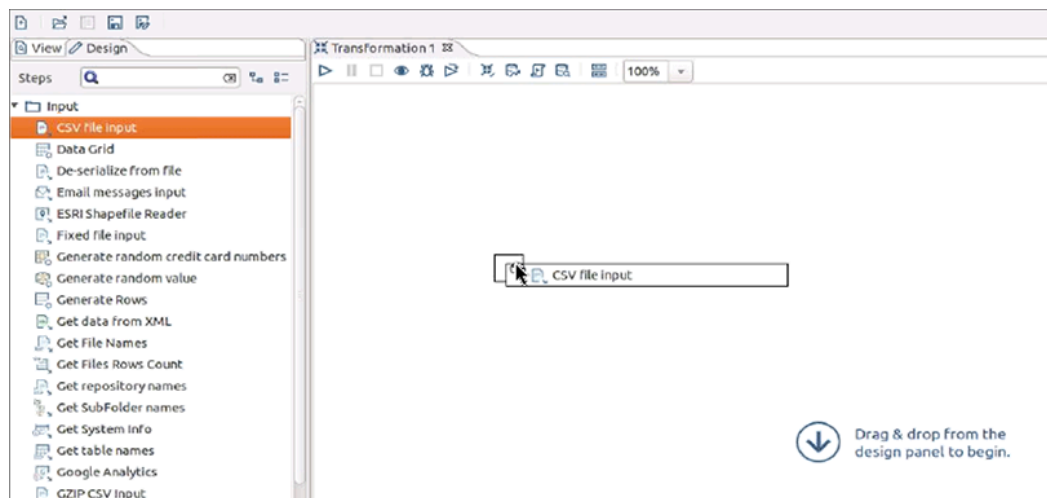1. Open the PDI client. You may use the command:

   ```
   ./spoon.sh &>/dev/null &
   ```

2. Right click **Transformations** in the left View tab, and click **New** to create a new transformation.
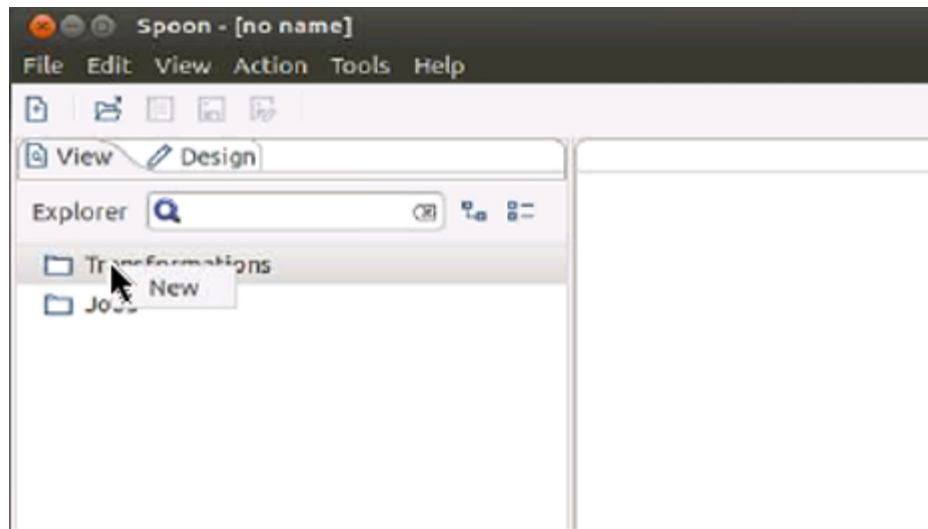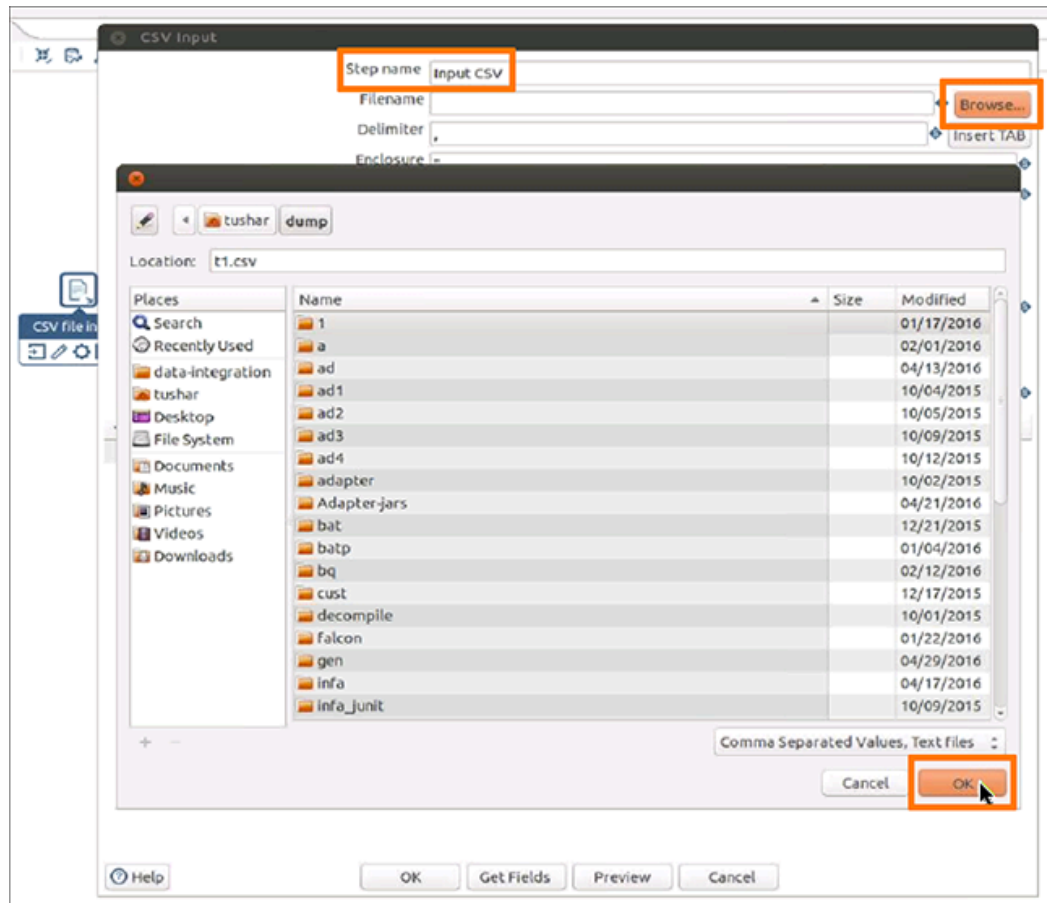
3. Click **Input** under the Design tab to expand it, and drag and drop **CSV File Input** to the Transformation window.

    This will bring in a new CSV file.

4. Double click the **CSV File Input** icon to open the CSV Input dialog box.
5. Name the Step.
6. Click **Browse** next to the Filename field to provide the file you want to read from.

7. Once you have selected the file, click **OK**.

8. In the CSV Input dialog box, click **Get Fields**.

9. Enter the number of lines you would like to sample in the Sample size dialog box.

    The default setting is 100.
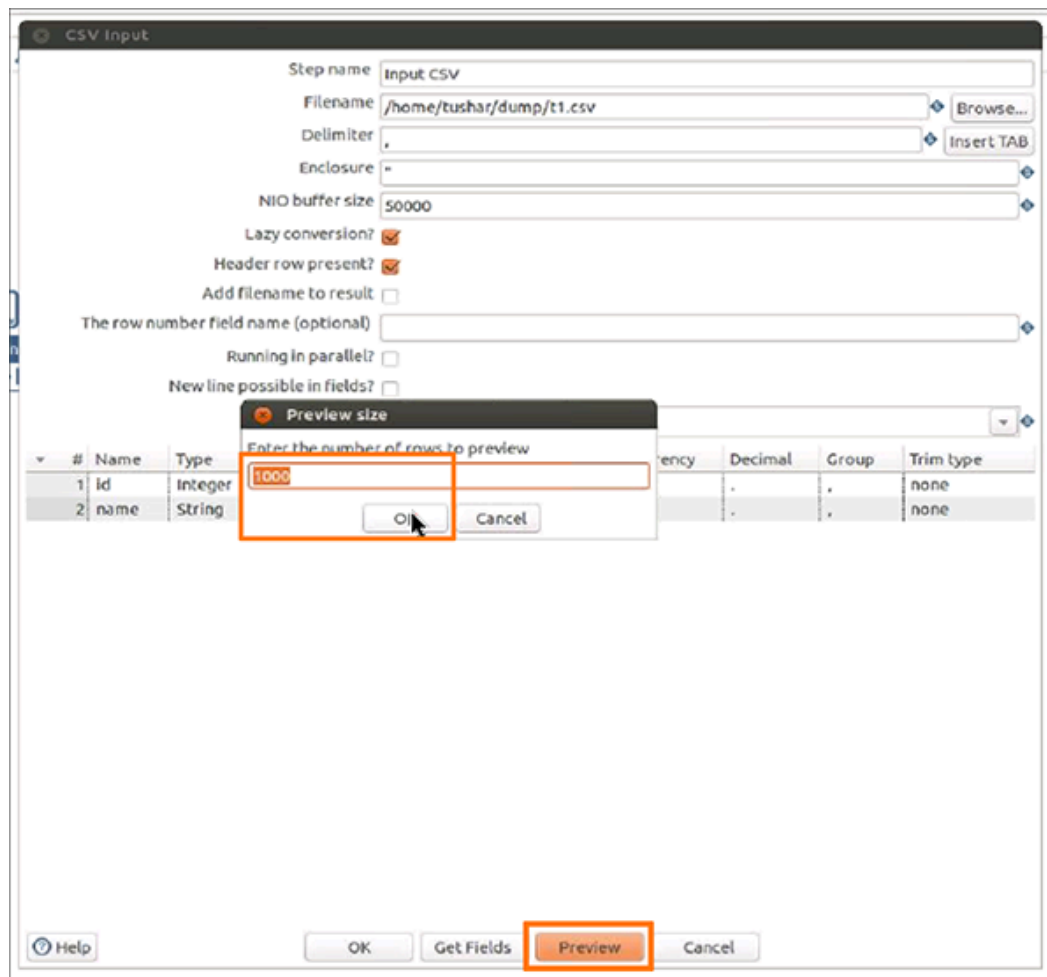
10. Click **OK** when you are ready.
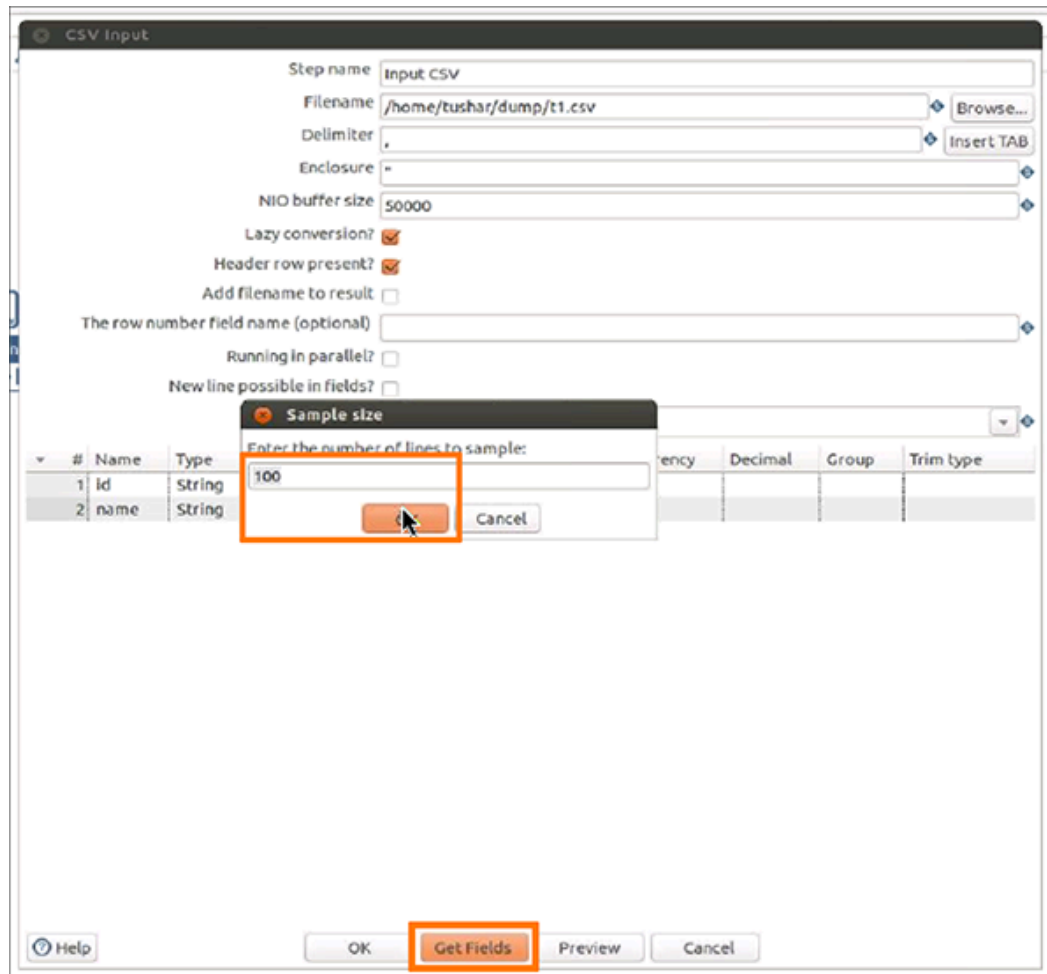
It will read the file and suggest the field name and type.
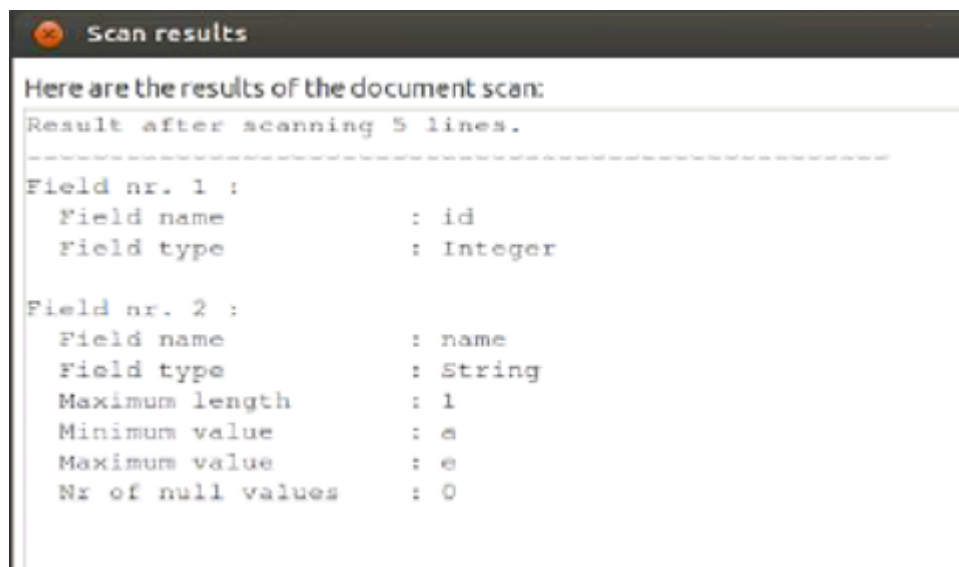
11. Click **Preview** to preview the data.

12. *Enter the number of rows to preview in the Preview size dialog box.*

The default setting is 1000. Click **OK** to start the transformation in preview.
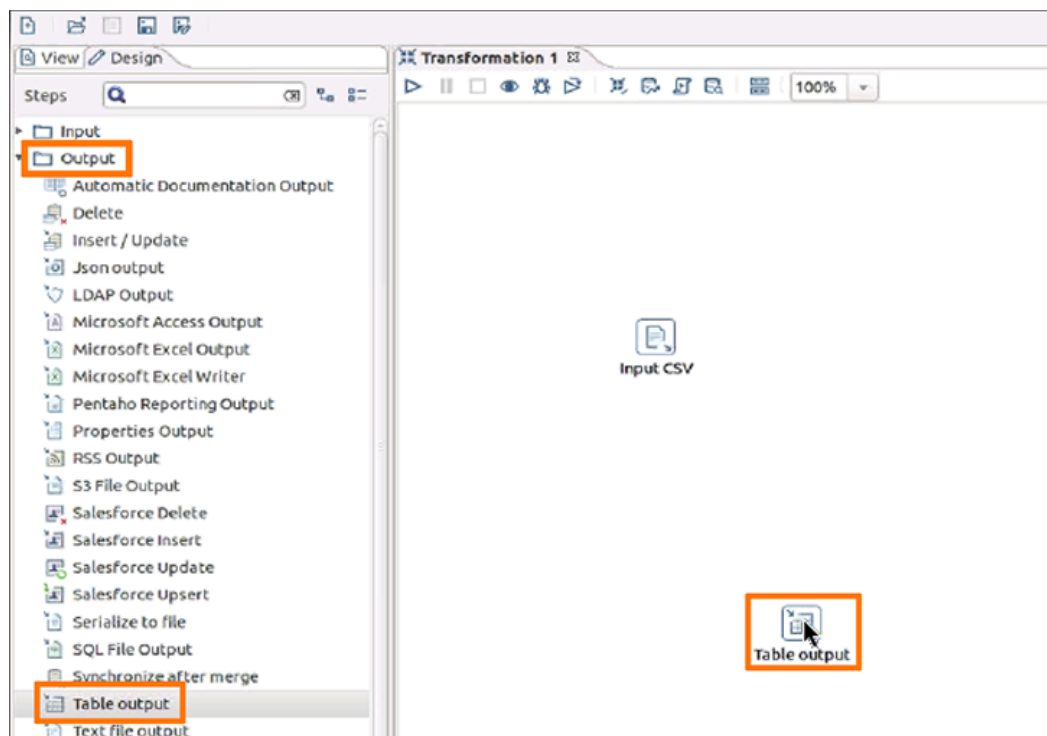

13. Examine the preview data, then click **Close**.

You may want to verify that you are able to read the data using the SQL query from Falcon.
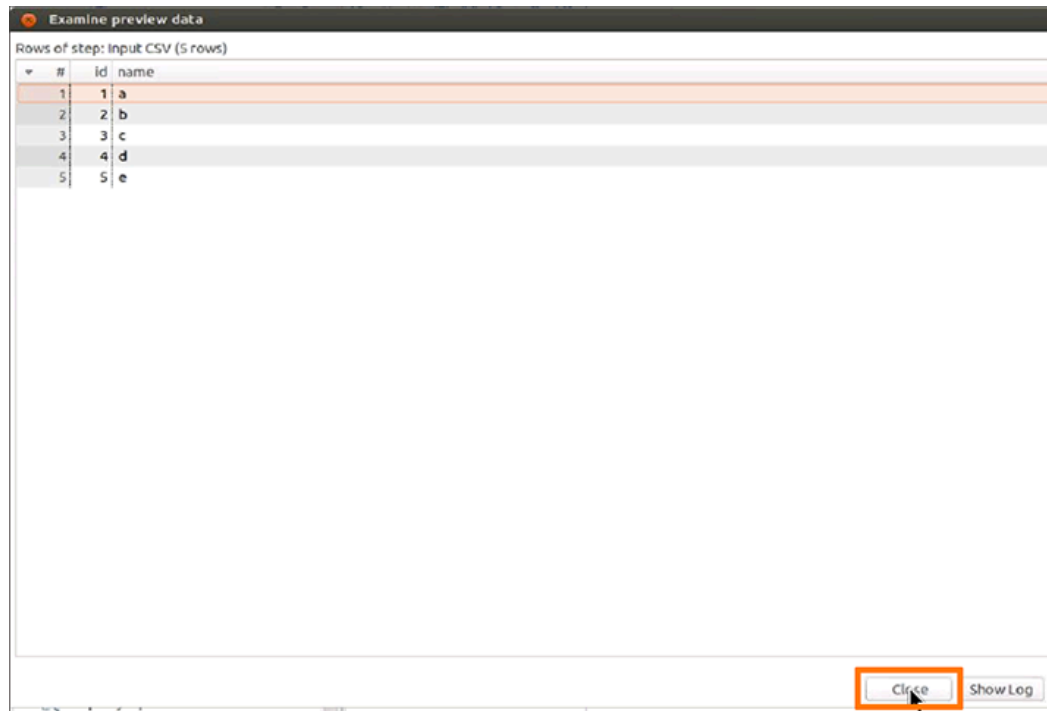
14. Click **OK** in the CSV Input dialog to confirm your CSV input settings.

15. Click **Output** under the Design tab to expand it, and drag and drop **Table output** to the Transformation window.


16. Double click the **Table output** icon to open the Table output dialog box.

17. Name the step. Then click **New** to create a new connection.

18. Enter or select the following information in the Database Connection dialog box:

- Connection Name
- Connection Type: Generic database
- Access: Native (JDBC)

- Custom Connection URL:
  `jdbc:simba://<server_ip>:12345;Database=<database_name or schema_name>`

  `<server_ip>` is the IP of your Falcon cluster. `<database_name>` is the name of the database you want to connect to. Use TQL to create a database name if needed.

- Custom Driver Class Name: `com.simba.client.core.jdbc4.JDBC4Driver` Please ensure that there are no leading or trailing spaces in the Custom Connection URL and the Custom Driver Class Name fields. JDBC will get confused if there are any such spaces, and as a result, will not be able to establish a connection.

- User Name and Password

  The User Name and Password are your ThoughtSpot credentials, but you can elect to keep these fields empty.
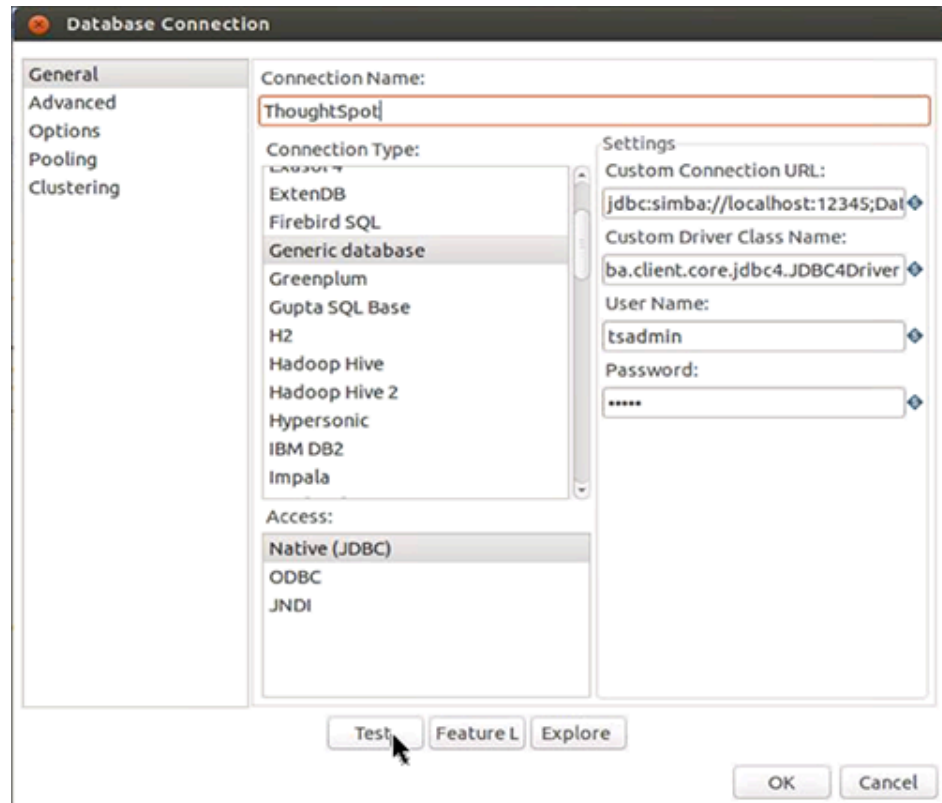
19.   Click **Test** to test your database connection.

20.   If you are able to make a successful connection to the Falcon Simba Server, click **OK**.
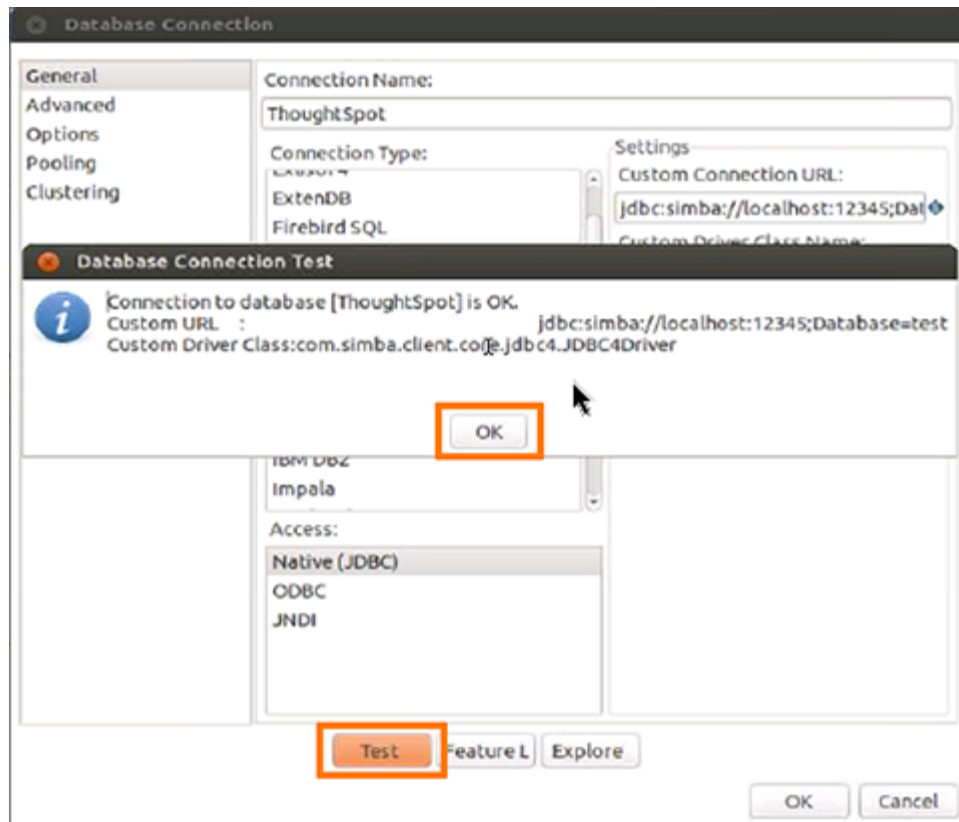
21. Click **OK** in the Database Connection dialog box to create the new connection.

Import data

1. In the Table output dialog box, select the connection you just created.
2. Click **Browse** next to the Target schema field, the select your **Target schema**.
3. Click **OK** when you are done.
4. Connect the Input CSV icon to the Table output icon by clicking and dragging an arrow.

5. When prompted, choose **Main output of step**.
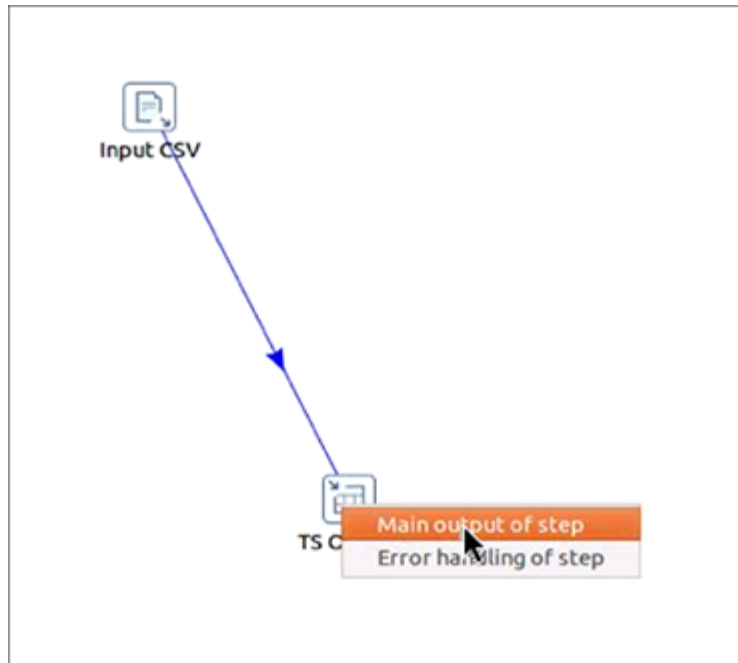
6. Double click the **Table output** icon to reopen the Table output dialog box.
7. Enter a **Target table name**.

8. Click **SQL**.

9. In the Simple SQL editor dialog box, click **Execute** to see the results of the SQL statements.
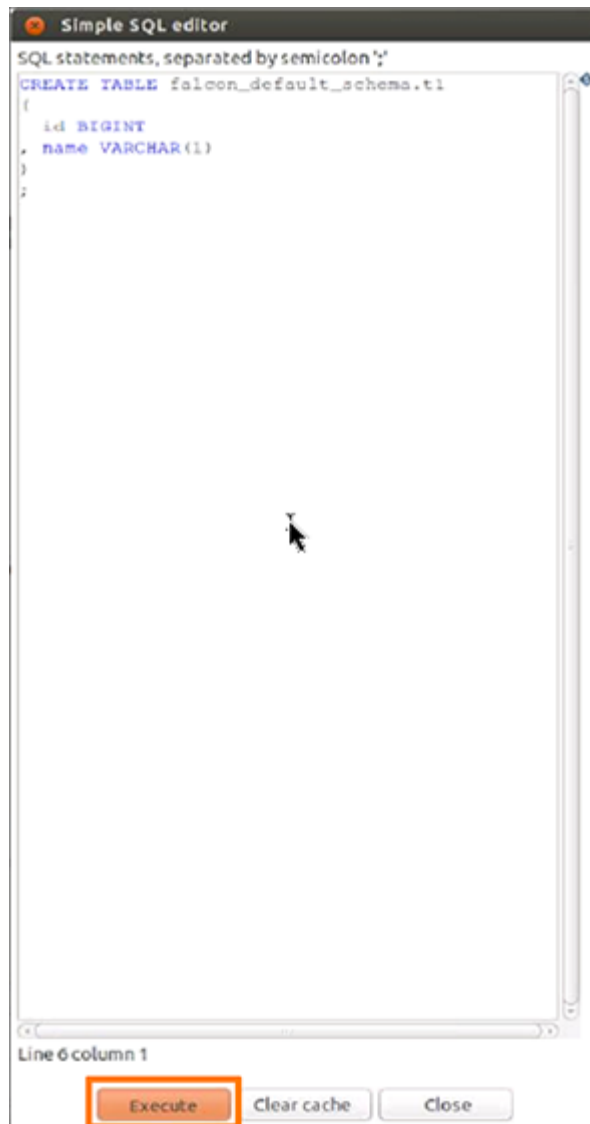
10. Close all open dialog boxes.

11. Click the **Play** button at the top of the Transformation window to execute the transformation.
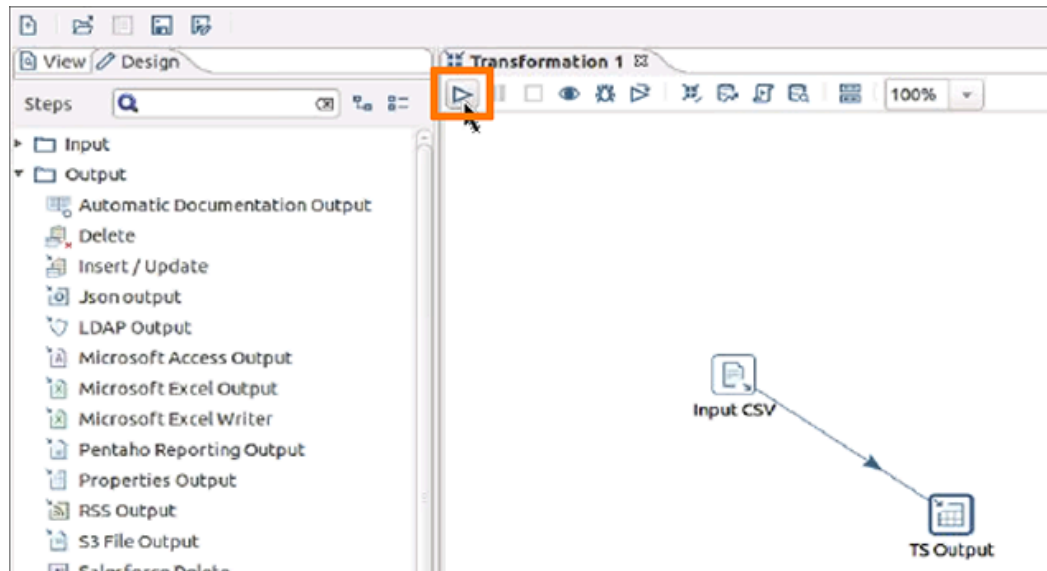
12. Click **Launch** in the Execute a transformation dialog box.

13. You will be asked to save it if you have not already.

14. View the Execution Results.

Troubleshooting Data Integrations

Summary: Learn how to fix connection issues.

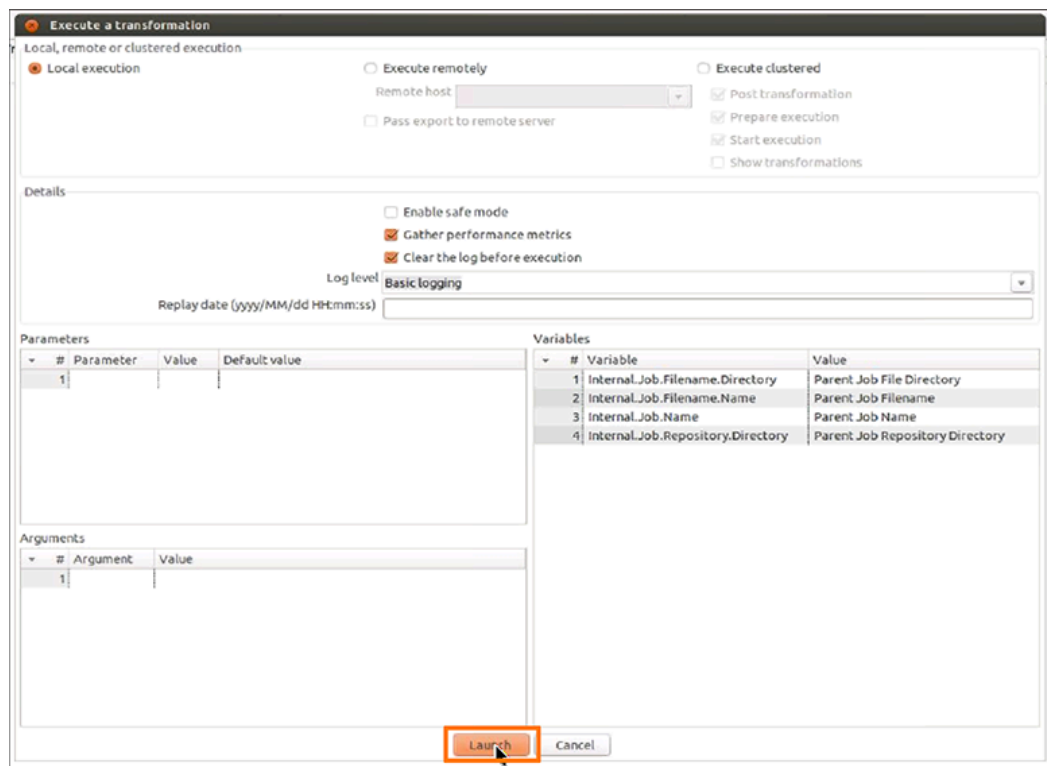This section can help if you're having trouble creating a connection or need to find out more information about what is going on with ODBC or JDBC.

The information contained here is very basic, and mostly about how to enable logs on the client side. If you need more detailed troubleshooting information or help, please contact ThoughtSpot Support.

- **Enabling ODBC Logs on Windows**
  If you need more information in order to troubleshoot ODBC connections, you can enable logging for ODBC. To do this on Windows, follow these instructions.
- **Enabling ODBC Logs on Linux or Solaris**
  If you need more information in order to troubleshoot ODBC connections, you can enable logging for ODBC. To do this on Linux or Solaris, follow these instructions.
- **Enabling JDBC Logs**
  To enable logging for JDBC, add the logging parameters to the connect string. Logs are stored on ThoughtSpot.
- **Schema not found error**
  When connecting with ODBC, you need to specify both the database and schema to connect to. If no schema is supplied, you will get an error indicating that the schema could not be found.
- **How to improve throughput of the load**
  The transaction/commit size value can improve the throughput of the load when setting up the ODBC Driver.

Enabling ODBC Logs on Windows

Summary: Using logs to aid in troulbeshooting.

If you need more information in order to troubleshoot ODBC connections, you can enable logging for ODBC. To do this on Windows, follow these instructions.

To enable ODBC logs on Windows:

1. Open the ODBC Data Source Administrator and select the **System DSN** tab.

2. Select your ThoughtSpot data source and click **Configure**.
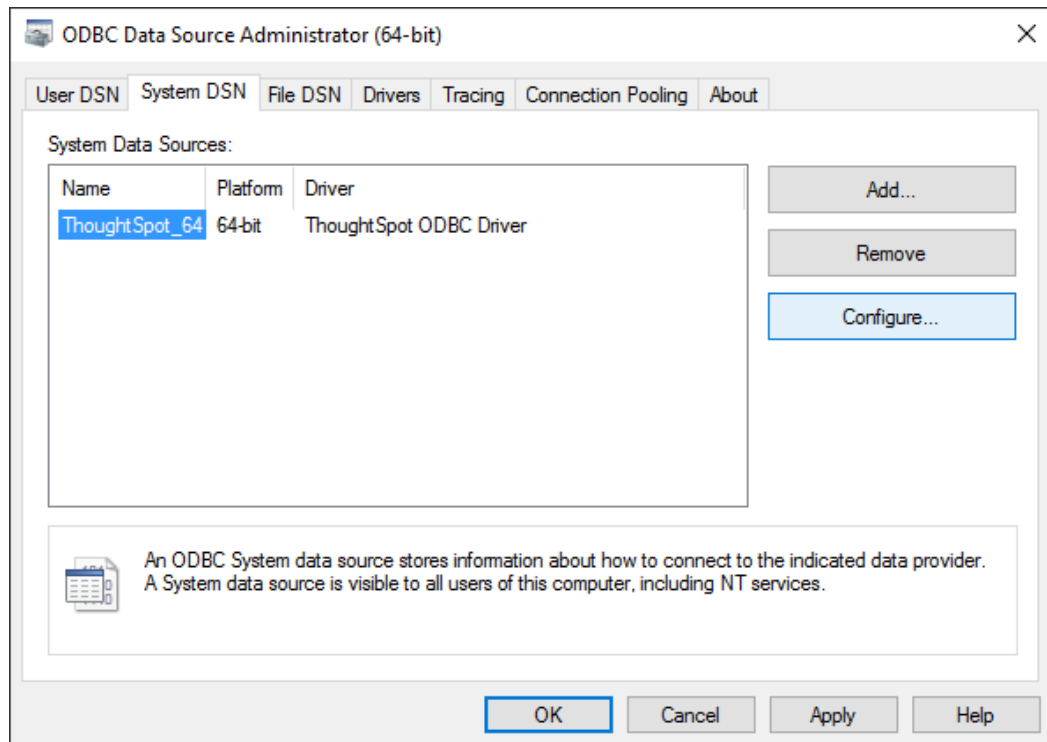
3. In the Client Configuration Dialog, click **Logging**.

4.   Choose a **Log Level**, depending on what level of verbosity you want to show in the logs.
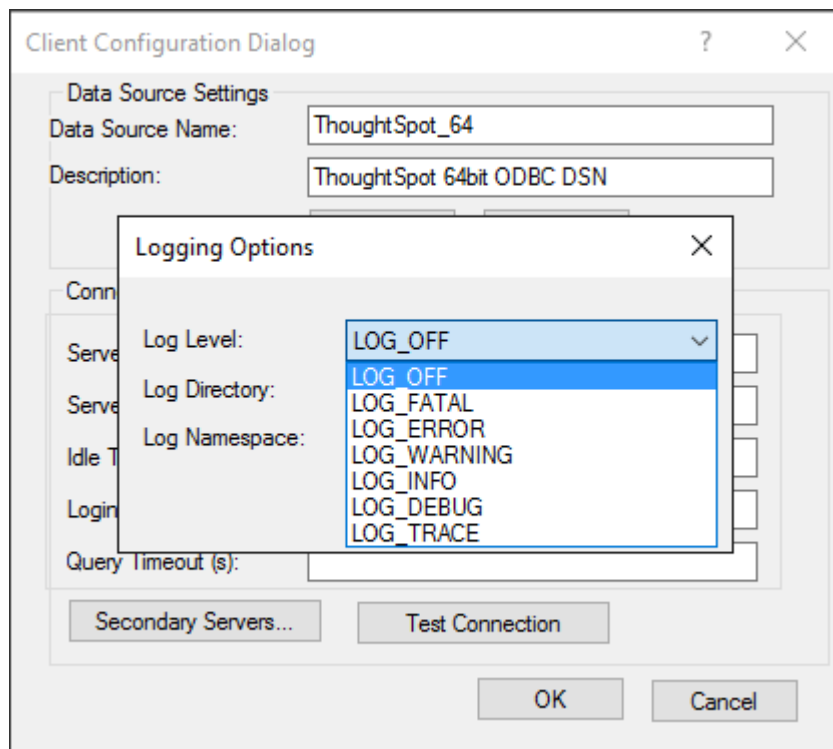
5.   For **Log Directory:**, type in the fully qualified path where you want the logs to be saved.
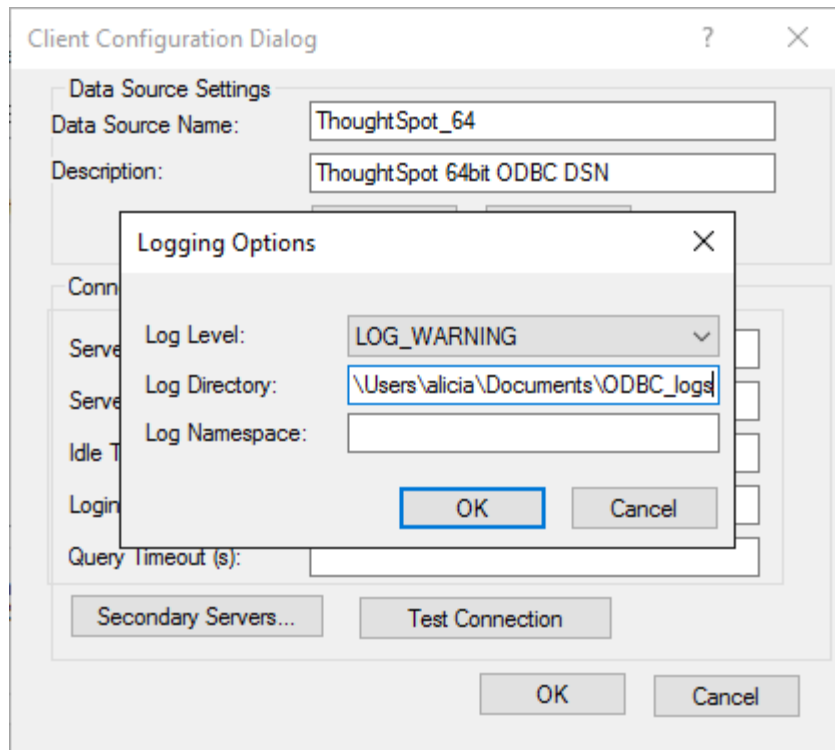
6. Click **OK** to save your settings, and **OK** again, to dismiss the ODBC Data Source Administrator.
7. Run the ODBC load.
8. Locate the log file that was generated, and send it to ThoughtSpot Support with a description of the problem.

Enabling ODBC Logs on Linux or Solaris

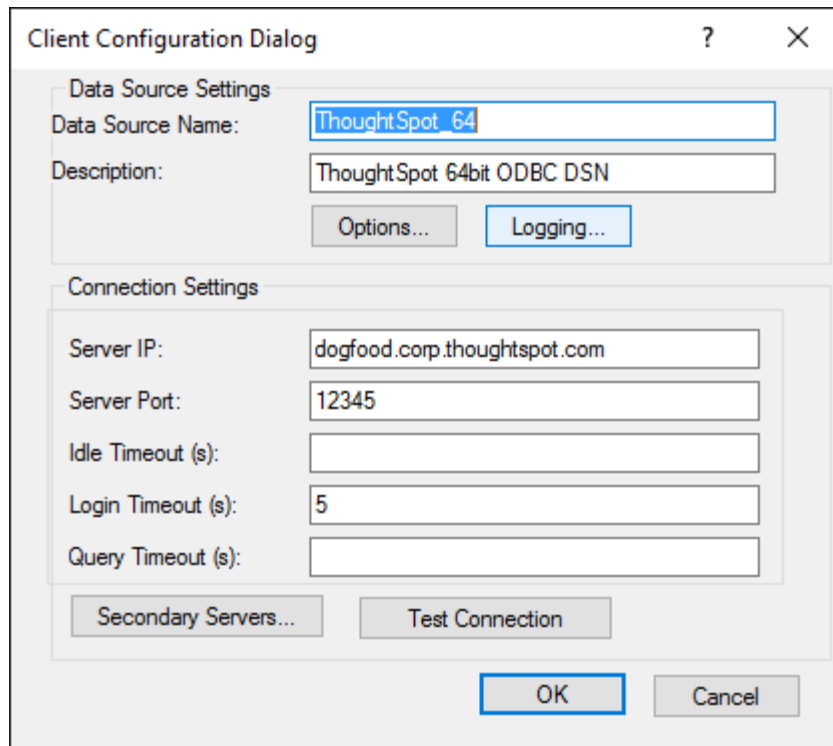**Summary:** Learn how to troubleshoot ODBC connections.

If you need more information in order to troubleshoot ODBC connections, you can enable logging for ODBC. To do this on Linux or Solaris, follow these instructions.

To enable ODBC logs:

1. Navigate to the directory where you installed ODBC.
2. Open the file `odbc.ini` in a text editor.
3. Find the settings `LogLevel` and `LogPath` in the file, and uncomment them by removing the "#" at the beginning of each line.

4. Edit the value for each of the logging properties:

   Example for Linux 64-bit:

   ```
   [ThoughtSpot_x64]
   Description        = ThoughtSpot 64-bit ODBC Driver
   Driver             = ThoughtSpot(x64)
   ServerList         = 127.0.0.1 12345
   Locale             = en-US
   ErrorMessagesPath  = /usr/local/scaligent/toolchain/local/simba/odbc/linux/
   ErrorMessages
   UseSsl             = 0
   #SSLCertFile       =   # Set the SSL certificate file path. The certificate
   file can be obtained by extracting the SDK tarball
   **\#LogLevel         = 3 \# Set log level to enable debug logging
   \#LogPath           = /usr/local/scaligent/toolchain/local/simba/odbc/linux/
   Logs  \# Set the debug log files path**
   ```

   For `LogLevel`, enter a number from 1 to 6 (with 6 being the most verbose). For `LogPath`, enter the fully qualified path where you want the log to be written.

5. Run the ODBC load, and send the log file to ThoughtSpot Support.

Enabling JDBC Logs

Summary: Configure logging parameter strings.

To enable logging for JDBC, add the logging parameters to the connect string. Logs are stored on ThoughtSpot. Before enabling JDBC logging, you will need:

- The level of logging you want to capture.
- The path on the ThoughtSpot server where the logs will be written. Make sure the directory has the correct permissions so that the "admin" Linux user can write logs to it.

To enable JDBC logging:

1. When forming the connect string for JDBC, add these two parameter, separated by "&":

   For example:

   ```
   jdbc:simba://192.168.2.248:12345;SERVERS=192.168.2.249:12345,

   192.168.2.247:12345;Database=test;Schema=falcon_default_schema;**LogLevel=3;Log
   Path=/usr/local/scaligent/logs**
   ```

   The `LogLevel` is the level of logging to capture (0-6). The `LogPath` is the fully qualified path where logs will be written on ThoughtSpot.

2. Run the JDBC code that uses the connection you modified.
3. Check the `LogPath` directory for logs generated by JDBC.

Schema not found error

Summary: Correct schema not found errors.

When connecting with ODBC, you need to specify both the database and schema to connect to. If no schema is supplied, you will get an error indicating that the schema could not be found.

When connecting with ODBC, remember to specify the schema. Note that you can add a default schema to use by supplying the parameter "SCHEMA" (Linux and Solaris) or the key "SCHEMA" (Windows).

Even if you do not use schema names in ThoughtSpot, you still have to specify a schema when connecting with ODBC. The default schema name in ThoughtSpot is "falcon_default_schema". This default schema is always assumed when you don't specify a schema name. However, with ODBC, you will need to specify it explicitly. To do this:

On Windows, change your ODBC configuration by adding a custom property with the key "SCHEMA" and the value "falcon_default_schema".
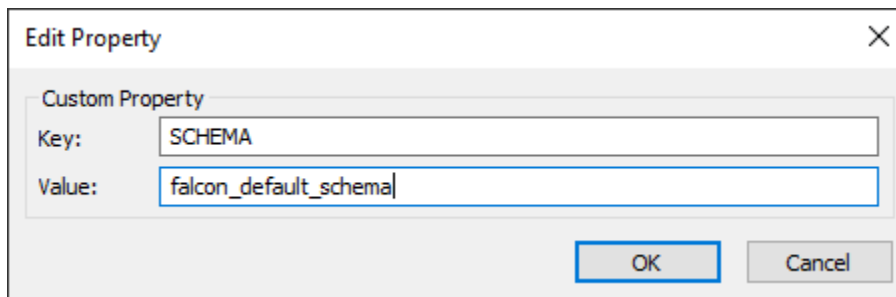
On Linux or Solaris, you can modify the custom properties related to database and schema, if you want to provide defaults. For a list of those properties and how to change them, see ODBC and JDBC configuration properties.

How to improve throughput of the load

Summary: Adjusting the transaction size may correct poor performance and low throughput.

The transaction/commit size value can improve the throughput of the load when setting up the ODBC Driver.

Adjusting the transaction size may correct poor performance and low throughput issues. The transaction size should be set to match the total number of rows that are expected to be loaded in the load cycle. However, increasing this value even higher should help improve throughput of the load.

❶ **Warning:** A high transaction size may slow down the ThoughtSpot system.
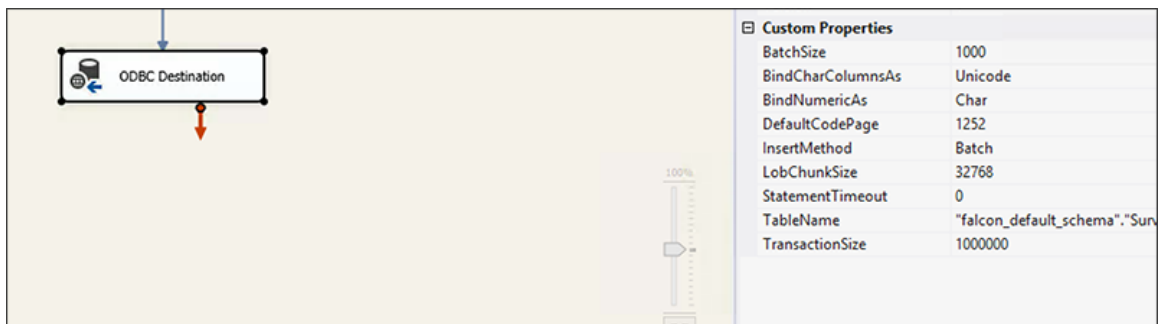
This is where the transaction size field exists for SSIS. Clicking on the ODBC destination reveals the properties on the right hand side, where the **Transaction Size** can be found.

See Set up the ODBC Driver for SSIS for more details on setting the transaction size.

ODBC supported SQL commands

Summary: ODBC driver supports a limited set of SQL commands.

The ODBC driver supports a limited set of SQL commands. When developing software that uses the ThoughtSpot ODBC driver, use this reference of supported commands. This reference is intended for developers using other tools (ETL, etc.) to connect to ThoughtSpot via the ODBC driver.

These SQL commands are supported for ODBC:

- CREATE TABLE

  Creates a table with the specified column definitions and constraints. The table is replicated on each node.

  ```
  CREATE TABLE country_dim (id_number int, country varchar, CONSTRAINT PRIMARY
  KEY (id_number));
  ```

- INSERT

  Creates placeholders in the table to receive the data.

  ```
  INSERT INTO TABLE country_dim (?, ?);
  ```

- DELETE FROM <table>

  Deletes ALL rows from the specified table. Does not support the WHERE clause.

  ```
  DELETE FROM country_dim;
  ```

- SELECT <cols_or_expression> FROM <table_list> [WHERE <predicates>] [GROUP BY <expressions>][ORDER BY <expressions>]

  Fetches the specified set of table data.

  ```
  SELECT id_number, country FROM country_dim WHERE id_number > 200;
  ```

ODBC and JDBC configuration properties

Summary: Lists the properties you can set for ODBC or JDBC connections

This section lists the properties you can set for ODBC or JDBC connections.

Setting Properties for ODBC

The properties information here comes mostly from the document Configuring SimbaClient for ODBC, published by Simba Technologies. You can access it directly <u>here</u> . Not all the parameters Simba accepts are supported by the ThoughtSpot ODBC clients, and ThoughtSpot has added some properties, which are listed separately here. All configuration properties use the type String (text).

You can set these properties on Windows by using the [ODBC Administrator](#).

For Linux and Solaris, the properties are located in three files, depending on their type:

Property Type	Location
DSN	`odbc.ini` file
Driver	`odbsinst.ini` file
SimbaSetting Reader	`simbaclient.ini` file

Setting Properties for JDBC

For JDBC, these properties are passed as key value pairs in the connect string. For more information, see [Use the JDBC Driver](#).

Properties Reference

The following tables summarize the configuration properties.

Property	Type	Description
`DATABASE`	DSN or Driver	The default database to connect to.
`SCHEMA`	DSN or Driver	The default schema to connect to.
`Description`	DSN	A brief, human-readable description of the DSN. This describes the DSN to users who are deciding which DSN to use.
`Driver`	DSN or Driver	In the driver configuration location, Driver should contain the path to the driver binary. In the DSN configuration location, Driver could contain the path to the driver binary, or it could contain the driver entry in the registry.

Property	Type	Description
Idle Timeout	DSN	The time to wait for a response from the server, in seconds. This property is optional, but SimbaClient will wait indefinitely for SimbaServer to respond to a request made to the server unless you specify a timeout period. IdleTimeout specifies how many seconds that SimbaClient will wait before aborting the attempt and returning to the application with an error. This timeout corresponds to ODBC's CONNECTION_TIMEOUT property and is only used when more specific timeouts, such as QUERY_TIMEOUT or LOGIN_TIMEOUT aren't applicable.
Locale	DSN	The connection locale. If this value is set, it overrides the driver-wide locale. For example, the driver-wide locale could be en-US. If the client would prefer fr-CA, it can set the connection locale to fr-CA. Values are composed of a 2-letter language code (in lower case), and an optional 2-letter country code (in upper case). If the country code is specified, it must be separated from the language code by a hyphen (-).
LoginTimeout	DSN	The timeout, in seconds, to wait for a response from the server when attempting to log in. A value of 0 means no timeout. The default value is 60.
Query Timeout	DSN	The timeout, in seconds, to wait for a response from the server during Prepare, Execute, and ExecuteDirect. A value of 0 means no timeout. The default value is 60.
ServerList	DSN	A comma separated list of all servers (IP address and port number) to connect to. SimbaClient must be able to find SimbaServer on the network. This property enables server discovery. SimbaClient will try to make a network connection to the servers in the order specified until a connection is made.
LogLevel	SimbaSetting Reader	Controls the granularity of the messages and events that are logged. With this keyword, you can control the amount of log output by controlling the kinds of events that are logged. Possible values (case sensitive): • 0 or LOG_OFF: no logging occurs • 1 or LOG_FATAL: only log fatal errors • 2 or LOG_ERROR: log all errors • 3 or LOG_WARNING: log all errors and warnings • 4 or LOG_INFO: log all errors, warnings, and informational messages • 5 or LOG_DEBUG: log method entry and exit points and parameter values for debugging • 6 or LOG_TRACE: log all method entry points
LogPath	SimbaSetting Reader	Specifies the directory where the log files are created. For example: `LogPath=C:\Simba Technologies\Temp` If this value is not set, the log files are written to the current working directory of the SimbaClient.
LogFileSize	SimbaSetting	The size of each log file, in bytes. The default values is 20971520 bytes.

Property	Type	Description
	Reader	When the maximum size of the file is reached, a new file is created.
LogFileCount	SimbaSetting Reader	The number of log files to create. When the maximum number of log files has been created, the oldest file will be deleted and a new one created. The default value is 50.