

TIDAL

Tidal Workload Automation Sqoop Adapter Guide

Version 6.3.3

First Published: January 2018

tidalautomation.com

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS. THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE PRODUCTS IN THIS MANUAL ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR STA GROUP REPRESENTATIVE FOR A COPY.

The implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. © 1981 Regents of the University of California. All rights reserved.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies are considered uncontrolled copies and the original online version should be referred to for latest version.

© 2018 STA Group LLC. All rights reserved.

Contents

Contents	3
Preface	5
Audience	5
Related Documentation	5
Obtaining Documentation and Submitting a Service Request	5
Document Change History	6
Introducing the Sqoop Adapter	7
Overview	7
Prerequisites	9
Software Requirements	9
Configuring the Sqoop Adapter	11
Overview	11
Installing the Hadoop Client Libraries	11
Installing Maven	11
Downloading the Hadoop Client Library	12
Configuring the Adapter	13
MapR Client Software Requirements	13
Licensing an Adapter	14
Securing the Adapter	15
Defining Runtime Users	16
Defining a Security Policy for Sqoop Adapter Jobs	17
Authorizing Schedulers to Work with Sqoop Adapter Jobs	17
Defining a Connection	18
Using the Sqoop Adapter	25
Overview	25
Defining Sqoop Jobs	25
Code Generation Task	27
Export Job Task	29
Import Job Task	31
Merge Job Task	36
Monitoring Sqoop Job Activity	36
Controlling Adapter and Agent Jobs	38
Holding a Job	39
Aborting a Job	39
Rerunning a Job	39
Making One Time Changes to an Adapter or Agent Job Instance	39
Deleting a Job Instance before It Has Run	40

Troubleshooting the Sqoop Adapter	41
Overview	41
Review Service Log files for More Information	41
Connection Failures	41
Setting Up Zookeeper on the Cluster	41
Job Failures	42
Adapter Is Out-of-Memory	42
Configuring service.props	43
About Configuring service.props	43
service.props Properties	43

Preface

This guide describes the installation, configuration, and usage of the Sqoop Adapter with Tidal Workload Automation (TWA).

Audience

This guide is for administrators who install and configure the Sqoop Adapter for use with TWA, and who troubleshoot TWA installation and requirements issues.

Related Documentation

For a list of all Tidal Workload Automation guides, see the *Tidal Workload Automation Documentation Overview* of your release on tidalautomation.com at:

<http://docs.tidalautomation.com/>

Note: We sometimes update the documentation after original publication. Therefore, you should also review the documentation on tidalautomation.com for any updates.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see *What's New in Tidal Product Documentation* at:

<http://docs.tidalautomation.com/rss>

Subscribe to *What's New in Tidal Product Documentation*, which lists all new and revised Tidal technical documentation, as an RSS feed and deliver content directly to your desktop using a reader application. The RSS feeds are a free service.

Document Change History

The table below provides the revision history for the *Tidal Workload Automation Sqoop Adapter Guide*.

Table 2-1

Version Number	Issue Date	Reason for Change
6.1.0	February 2013	New Cisco version.
6.2.1	June 2014	Available in online Help only.
6.2.1 SP2	June 2015	Configuration provided in the <i>TWA Installation Guide</i> ; usage provided in online Help only.
6.2.1 SP3	May 2016	Consolidated all Sqoop Adapter documentation into one document.
6.3 Beta	June 2016	Rebranded “Cisco Tidal Enterprise Scheduler (TES)” to “Cisco Workload Automation (CWA)”. Added the new Installing the Hadoop Client Libraries, page 11 section. Updates to the Configuring the Adapter, page 13 section. Updates to the Defining a Connection, page 18 section. Added the service.props configuration chapter. Updated and corrected the documentation for the 6.3 release.
6.3.3	January 2018	Rebranded “Cisco Workload Automation (CWA)” to “Tidal Workload Automation (TWA)”

1

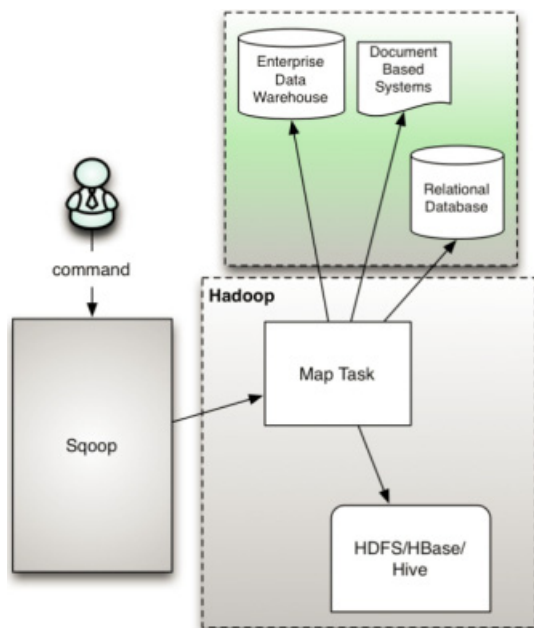
Introducing the Sqoop Adapter

This chapter provides an overview of the Sqoop Adapter and its requirements:

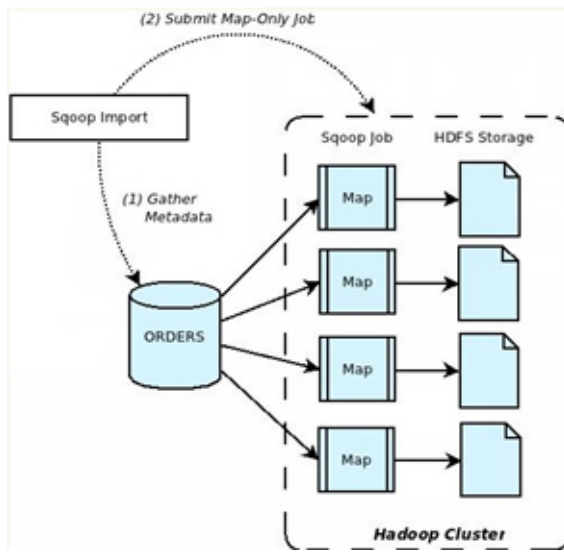
- [Overview, page 7](#)
- [Prerequisites, page 9](#)
- [Software Requirements, page 9](#)

Overview

The Tidal Workload Automation (TWA) Sqoop Adapter provides easy import and export of data from structured data stores such as relational databases and enterprise data warehouses. Sqoop is a tool designed to transfer data between Hadoop and relational databases. You can use Sqoop to import data from a relational database management system (RDBMS) into the Hadoop Distributed File System (HDFS), transform the data in Hadoop MapReduce, and then export the data back into an RDBMS. Sqoop Adapter allows users to automate the tasks carried out by Sqoop.

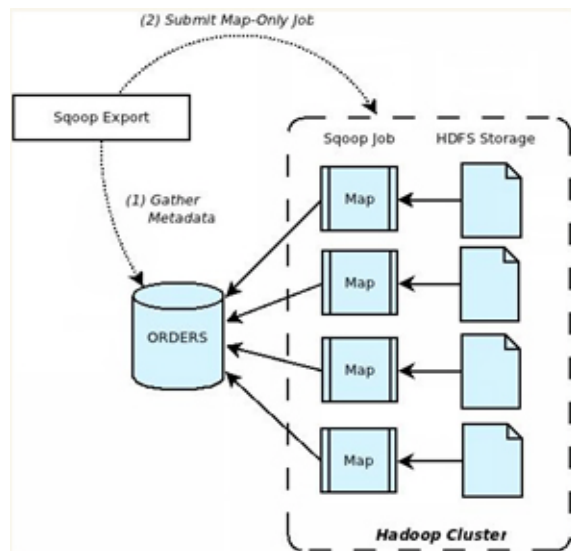


The import is performed in two steps as depicted in figure below. In the first Step Sqoop introspects the database to gather the necessary metadata for the data being imported. The second step is a map-only Hadoop job that Sqoop submits to the cluster. It is this job that does the actual data transfer using the metadata captured in the previous step.



The imported data is saved in a directory on HDFS based on the table being imported. As is the case with most aspects of Sqoop operation, the user can specify any alternative directory where the files should be populated.

Export is done in two steps as depicted in figure below. The first step is to introspect the database for metadata, followed by the second step of transferring the data. Sqoop divides the input dataset into splits and then uses individual map tasks to push the splits to the database. Each map task performs this transfer over many transactions in order to ensure optimal throughput and minimal resource utilization.



Prerequisites

Before you can run the Sqoop Adapter, the following prerequisites must be met:

- Linux is the only supported production platform for Apache Hadoop. However, the Sqoop Adapter can run on any platform supported by the TWA Master. See the *Tidal Workload Automation Compatibility Guide* for current version support.

Note: MapR configuration is supported only on Windows 2008 and Linux servers.

- JDK must be installed on TWA Master machine and TWA Master machine must have environmental variable "JAVA_HOME" pointing to the directory where JDK is installed and not to a JRE directory.
- Tidal Workload Automation Adapters require Java 8. (Refer to *Tidal Workload Automation Compatibility Guide* for further details).
- Apache Sqoop is installed on with Hadoop Cluster.
- Hadoop cluster and database are accessible to each other and the TWA Master over the network.
- TWA Master has the database drivers available on CLASSPATH.
- Hadoop cluster has the database drivers available on HADOOP_CLASSPATH. (This can be configured in `hadoop-env.conf`).

Software Requirements

The 6.3 Sqoop Adapter is installed with the TWA 6.3 master and client and cannot be used with an earlier TWA version.

Refer to your *Tidal Workload Automation Compatibility Guide* for a complete list of hardware and software requirements.

2

Configuring the Sqoop Adapter

Overview

The Sqoop Adapter software is installed as part of a standard installation of TWA. However, you must perform the following steps to license and configure the Adapter before you can schedule and run Sqoop jobs:

- [Installing the Hadoop Client Libraries](#) - Install the necessary Hadoop client libraries for Sqoop.
- [Configuring the Adapter](#) - Add configuration properties to the service.props file.
- [Licensing an Adapter](#) - Apply the license to the Adapter. You cannot define a Sqoop connection until you have applied the license.
- [Securing the Adapter](#) - Define Sqoop users that the Adapter can use to establish authenticated sessions with the Sqoop server and permit requests to be made on behalf of the authenticated account.
- [Defining a Connection](#) - Define a connection so the master can communicate with the Sqoop server.

See [Configuring service.props, page 43](#) for information about general and adapter-specific properties that can be set to control things like logging and connection properties.

Installing the Hadoop Client Libraries

Hadoop client libraries are required for processing the Hadoop-related DataMover, Hive, MapReduce, and Sqoop jobs. As of TWA 6.3, Hadoop libraries are not included with TWA. Instead, we provide a Maven script (POM.xml) to install the required libraries.

If you do not already have Maven, you must download and install it. Obtain the POM.xml file from the folder/directory named "Hadoop" in the CD and run the file script to download the required Hadoop client libraries. Instructions for obtaining Maven and downloading the Hadoop libraries are included in these sections:

- [Installing Maven, page 11](#)
- [Downloading the Hadoop Client Library, page 12](#)

Note: The instructions here are for Windows.

Installing Maven

If you do not have Maven installed, follow the instructions below.

Maven Prerequisites

- JDK must be installed.
- The JAVA_HOME environment variable must be set and point to your JDK.

To download and install Maven:

1. Download maven 3 or above from <https://maven.apache.org/download.cgi>.

2. Unzip apache-maven-<3 or above>-bin.zip.
3. Add the bin directory of the created directory (for example, apache-maven-3.3.9) to the PATH environment variable
4. Confirm a successful Maven installation by running the `mvn -v` command in a new shell. The result should look similar to this:

```
C:\Users\subrechan\Desktop>mvn -v
Apache Maven 3.3.9 (bb5280502b132cc0a5a3f4c09453c07478323dc5; 2015-11-10T22:11:47+05:30)
Maven home: C:\winoth\software\apache-maven-3.3.9-bin\apache-maven-3.3.9
Java version: 1.7.0_79, vendor: Oracle Corporation
Java home: C:\Program Files\Java\jdk1.7.0_79\jre
Default locale: en_US, platform encoding: Cp1252
OS name: "windows 7", version: "6.1", arch: "amd64", family: "windows"
```

Downloading the Hadoop Client Library

With Maven installed, you can now download the Hadoop client library. Maven scripts (POM.xml) are provided for the following distributions of Hadoop:

Table 2-1

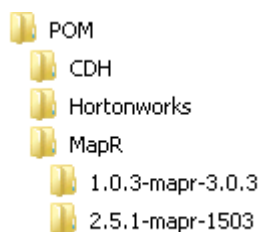
Hadoop Distribution Type	Versions
Cloudera	CDH5
Hortonworks	HDP 2.4.x
MapR	5.1.0

Note: The *Tidal Workload Automation Compatibility Guide* contains the most current version information.

To download and install the Hadoop client library

1. Download the POM.zip file. This file is provided in the /Hadoop directory in the TWA 6.3 distribution package.
2. Unzip the POM.zip.

The POM xml files needed by Maven are saved in the directory structure shown here:



3. Open a Windows command prompt and navigate to the directory for the Hadoop distribution in which you are interested. For example, navigate to the CDH directory if you want to download Hadoop client libraries for Cloudera.
4. Edit the POM.xml file to mention exact versions of MapR, Hadoop, Hive, and Sqoop that you are using. For example, for Cloudera the required properties could be edited as shown below:

```
<properties>
<Hadoop.version>2.6.0-cdh5.6.0</Hadoop.version>
<Hive.version>1.1.0-cdh5.7.0</Hive.version>
<Sqoop.version>1.4.6-cdh5.6.0</Sqoop.version>
</properties>
```

For MapR it is also necessary to mention the version of MapR used, as shown in the following example:

```
<properties>
<Hadoop.version>2.7.0-mapr-1602</Hadoop.version>
<Hive.version>1.2.0-mapr-1605</Hive.version>
<Sqoop.version>1.4.6-mapr-1601</Sqoop.version>
<Mapr.version>5.1.0-mapr</Mapr.version>
</properties>
```

5. From the directory containing the Hadoop distribution you want, execute this command:

```
mvn dependency:copy-dependencies -DoutputDirectory=<directory to which you want to download the jars>
```

For example, running the following command from the CDH directory:

```
mvn dependency:copy-dependencies -DoutputDirectory=C:\CDHlib
```

would insert the Cloudera Hadoop client libraries to the “C:\CDHlib” directory.

Configuring the Adapter

You must configure the Sqoop Adapter before you can schedule and run Sqoop jobs.

To configure the Sqoop Adapter:

1. Stop the Master.
2. In the {722A6A78-7C2C-4D8B-AA07-B0D9CED6C55A} directory, create a Config subdirectory.
3. Create the *service.props* file in the Config directory if it doesn't already exist. See [Configuring service.props, page 43](#).
4. (For the MapR Distribution only) Install the MapR client in the TWA Master machine, and add the following lines in the *service.props* file for your platform:

Windows:

```
JVMARGS=-Djava.library.path=C:\\opt\\maprv1\\hadoop\\hadoop-0.20.2\\lib\\native\\Windows_7-amd64-64
```

Linux:

```
JVMARGS=-Djava.library.path=/opt/mapr/hadoop/hadoop-0.20.2/lib/native/Linux-amd64-64
```

All paths above are derived from the MapR Client installation. If a filename does not exactly match, use the match closest to the filename. See [MapR Client Software Requirements, page 13](#)

5. (Optional) Add properties to *service.prop* to control the polling, output, and logging for the Sqoop Adapter. See [Configuring service.props, page 43](#).
6. Restart the Master.

Note: Make sure JDK is installed in the machine and set the JAVA_HOME, and add JAVA_HOME/bin to the system PATH. The path to the database drivers and the sqoop jars must be added to the HADOOP_CLASSPATH.

MapR Client Software Requirements

When using MapR:

- MapR Client software must be configured on the TWA master machine.

- MapR Client software must be configured appropriately using the link <http://www.mapr.com/doc/display/MapR/Setting+Up+the+Client>. The Adapter will not work unless there is confirmed communication between the client and cluster.
- Although MapR does not support user impersonation from Windows, spoofing is supported. Refer <http://doc.mapr.com/display/MapR/Setting+Up+the+Client#Configuring%20MapR%20Client%20User%20on%20Windows>.
- The Hadoop client libraries must be downloaded as described in [Installing the Hadoop Client Libraries, page 11](#).
- When using the MapR distribution, `service.props` must be modified for your platform. See [Configuring service.props, page 43](#).
- If the Java version used in the Hadoop environment is lower than Java 8, then install the same lower JDK version in the TWA Master and include the path to the JDK in the `HADOOP_JAVA_HOME` property in `service.props`. See [Configuring service.props](#). This is required to generate code that runs on a lower version of Java.

For example:

```
HADOOP_JAVA_HOME=C:\\Program Files\\Java\\jdk1.7.0_71
```

- If the MapR setup is configured to use Kerberos, open "mapr.login" file in the MapR client and edit the `hadoop_default_keytab` entry to have the following:

```
com.sun.security.auth.module.Krb5LoginModule required
    useKeyTab=true
    principal="<<user principle>>"
    keyTab="<<Path to the keytab file>>";
```

An example the `hadoop_default_keytab` entry could look as follows:

```
hadoop_default_keytab {

    com.sun.security.auth.module.Krb5LoginModule required
        useKeyTab=true
        doNotPrompt=false
        principal="mapr/TWA kerb.cluster.com@TIDALSOFT.LOCAL"
        keyTab="C:/opt/mapr/conf/mapr.keytab"
        debug=true
        useTicketCache=false;

    org.apache.hadoop.security.login.GenericOSLoginModule required;
    com.mapr.security.maprsasl.MaprSecurityLoginModule required
        checkUGI=false
        useServerKey=true;
    org.apache.hadoop.security.login.HadoopLoginModule required
        principalPriority=com.mapr.security.MapRPrincipal;

};
```

Licensing an Adapter

Each TWA Adapter must be separately licensed. You cannot use an Adapter until you apply the license file. If you purchase the Adapter after the original installation of TWA, you will receive a new license file authorizing the use of the Adapter.

You might have a Demo license which is good for 30 days, or you might have a Permanent license. The procedures to install these license files are described below.

To license an Adapter:**1. Stop the master:**

Windows:

- a. Click on **Start** and select **All Programs>Tidal Workload Automation >Scheduler>Service Control Manager**.
- b. Verify that the master is displayed in the **Service** list and click on the **Stop** button to stop the master.

UNIX:

Enter **tesm stop**

2. Create the license file:

- For a Permanent license, rename your Permanent license file to *master.lic*.
- For a Demo license, create a file called *demo.lic*, then type the demo code into the *demo.lic* file.

3. Place the file in the C:\Program Files\TIDAL\Scheduler\Master\config directory.**4. Restart the master:**

Windows:

Click **Start** in the Service Control Manager.

UNIX:

Enter **tesm start**

The master will read and apply the license when it starts.

5. To validate that the license was applied, select Registered License from Activities main menu.

Securing the Adapter

There are two types of users associated with the Sqoop Adapter, **Runtime Users** and **Schedulers**. You maintain definitions for both types of users from the **Users** pane.

■ Runtime Users

Runtime users in the context of Sqoop jobs represent those users and passwords required for authentication. Sqoop operations require authentication against a valid Sqoop user as defined by a Sqoop administrator. You can also use runtime users to override data source logons used by your reports.

■ Schedulers

Schedulers are those users who will define and/or manage Sqoop jobs. There are three aspects of a user profile that grant and/or limit access to scheduling jobs that affect Sqoop:

- Security policy that grants or denies add, edit, delete and view capabilities for Sqoop jobs.
- Authorized runtime user list that grants or denies access to specific authentication accounts for use with Sqoop jobs.
- Authorized agent list that grants or denies access to specific Sqoop Adapter connections for use when defining Sqoop jobs.

Defining Runtime Users

To define a Sqoop connection, a Hadoop and a database user must be specified. These users can act as sqoop user if they are configured with a Sqoop password.

To define a runtime user:

1. From the **Navigator** pane, expand the **Administration** node and select **Runtime Users** to display the defined users.
2. Right-click **Runtime Users** and select **Add Runtime User** from the context menu (*Insert mode*).

-or-

Click the **Add** button on the TWA menu bar. The **User Definition** dialog box displays.

The screenshot shows the 'User Definition' dialog box with the following fields and controls:

- User Name:** Sqoop Runtime User
- Full Name:** Sqoop Runtime User
- Domain:** dv
- Windows/FTP:** (empty field)
- Adapter:** Sqoop Password (selected from a dropdown menu)
- Password:** ***
- Buttons:** Add, Edit, Delete (next to the password table), OK, Cancel (top right)

3. Enter the new user name in the **User Name** field.
4. For documentation, enter the **Full Name** or description associated with this user.
5. In the **Domain** field, select a Windows domain associated with the user account required for authentication, if necessary.
6. To define this user as a runtime user for Sqoop Adapter jobs, click **Add** on the **Passwords** tab.

The **Change Password** dialog box displays.

7. Select **Sqoop** from the **Password Type** list.
8. Enter a password (along with confirmation) in the **Password/Confirm Password** fields.

Only those users with a password specified for Sqoop will be available for use with Sqoop jobs. The password might be the same as the one specified for Windows/FTP/DataMover jobs.

9. Click **OK** to return to the **User Definition** dialog box.

The new password record displays on the **Passwords** tab.

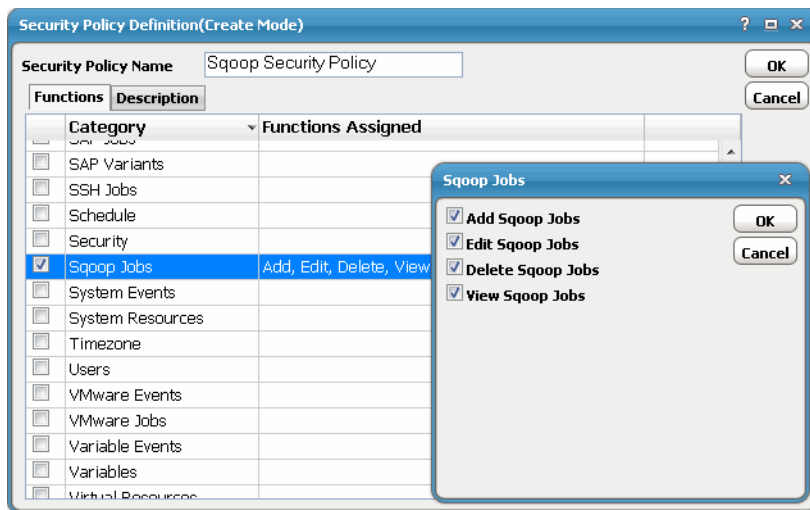
10. Click **OK** to add or save the user record in the TWA database.

For further information about the **User Definition** dialog box, see your *Tidal Workload Automation User Guide*.

Defining a Security Policy for Sqoop Adapter Jobs

To define a security policy:

1. From the **Navigator** pane, select **Administration > Security Policies** to display the **Security Policies** pane.
2. Right-click **Security Policies** and select **Add Security Policy** from the context menu. You can also right-click to select an existing security policy in the **Security Policies** pane and select **Edit Security Policy**.



3. In the **Security Policy Name** field, enter a name for the policy.
4. On the **Functions** page, scroll to the **Sqoop Jobs** category, click the ellipses on the right-hand side of the dialog box and select the check boxes next to the functions that are to be authorized under this policy (**Add**, **Edit**, **Delete** and **View Sqoop Jobs**).
5. Click **Close** on the **Function** drop-down list.
6. Click **OK** to save the policy.

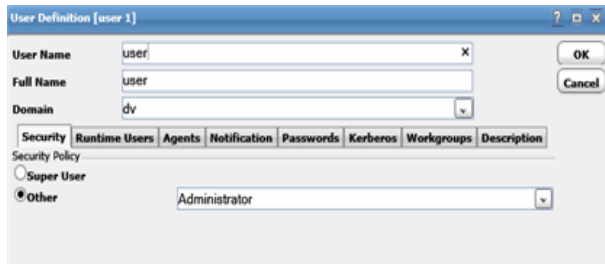
For further information about setting up security policies, see your *Tidal Workload Automation User Guide*.

Authorizing Schedulers to Work with Sqoop Adapter Jobs

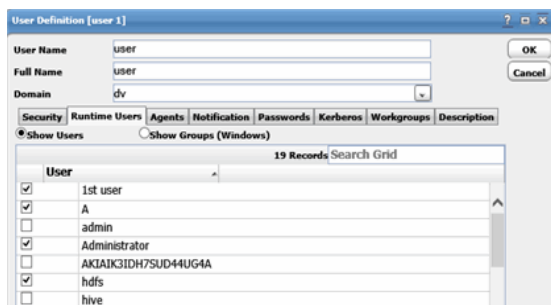
To define a Scheduler user to work with Sqoop Adapter jobs:

1. From the **Navigator** pane, expand the **Administrative** node and select **Interactive Users** to display the defined users.
2. Right-click **Interactive Users** and select **Add Interactive User** from the context menu (*Insert* mode). You can also right-click a user in the **Interactive Users** pane and select **Edit Interactive User** from the shortcut menu (*Edit* mode).

The **User Definition** dialog box displays.



3. If this is a new user definition, enter the new user name in the **User/Group Name** field.
4. For documentation, enter the **Full Name** or description associated with this user.
5. In the **Domain** field, select a Windows domain associated with the user account required for authentication, if necessary.
6. On the **Security** page, select the **Other** option and then select the security policy that includes authorization for Sqoop Adapter jobs.
7. Click the **Runtime Users** tab.



8. Select the Sqoop Adapter users that this scheduling user can use for Sqoop Adapter authentication from Sqoop Adapter jobs.
9. Click the **Agents** tab.
10. Select the check boxes for the Sqoop Adapter connections that this scheduling user can access when scheduling jobs.
11. Click **OK** to save the user definition.

The Adapter supports only Simple authentication.

Defining a Connection

You must create one or more Sqoop connections before TWA can run your Sqoop Adapter jobs. These connections also must be licensed before TWA can use them. A connection is created using the **Connection Definition** dialog box.

A Sqoop connection includes the following two parts:

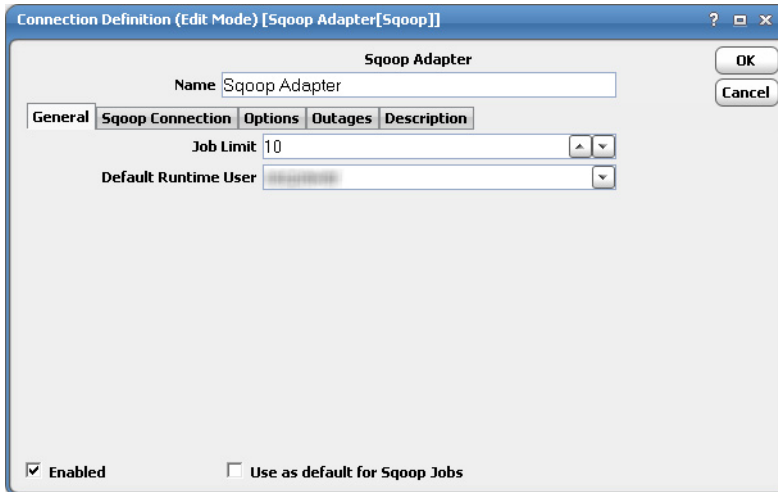
- **Sqoop DB Connection using JDBC** – This information is used by the TWA master to interact with the database using Sqoop tools and to pass on the JDBC connection information to a Sqoop Job.
- **Hadoop Connection** – This information is used by the TWA master to interact with Hadoop Cluster and to pass on the Hadoop cluster information to a Sqoop Job.

To define a Sqoop connection, a job tracker, namenode, and user must be specified. A Sqoop user is a user with a password.

To define a Sqoop connection:

1. From the **Navigator** pane, navigate to **Administration>Connections** to display the **Connections** pane.
2. Right-click **Connections** and select **Add Connection>Sqoop Adapter** from the context menu.

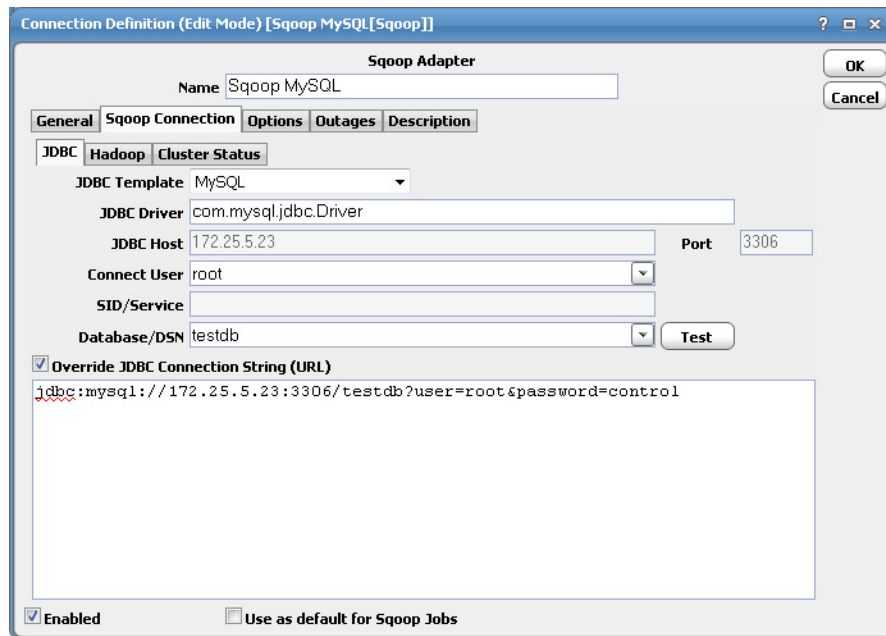
The **Sqoop Adapter Connection Definition** dialog box displays.



3. On the **General** page, enter a name for the new connection in the **Name** field.
4. In the **Job Limit** field, select the maximum number of concurrent active processes that TWA should submit to the server at one time.
5. From the **Default Runtime User** drop-down list, you have the option to select the name of a default user for Sqoop Adapter jobs. The runtime user is auto-selected when defining Sqoop Adapter jobs.

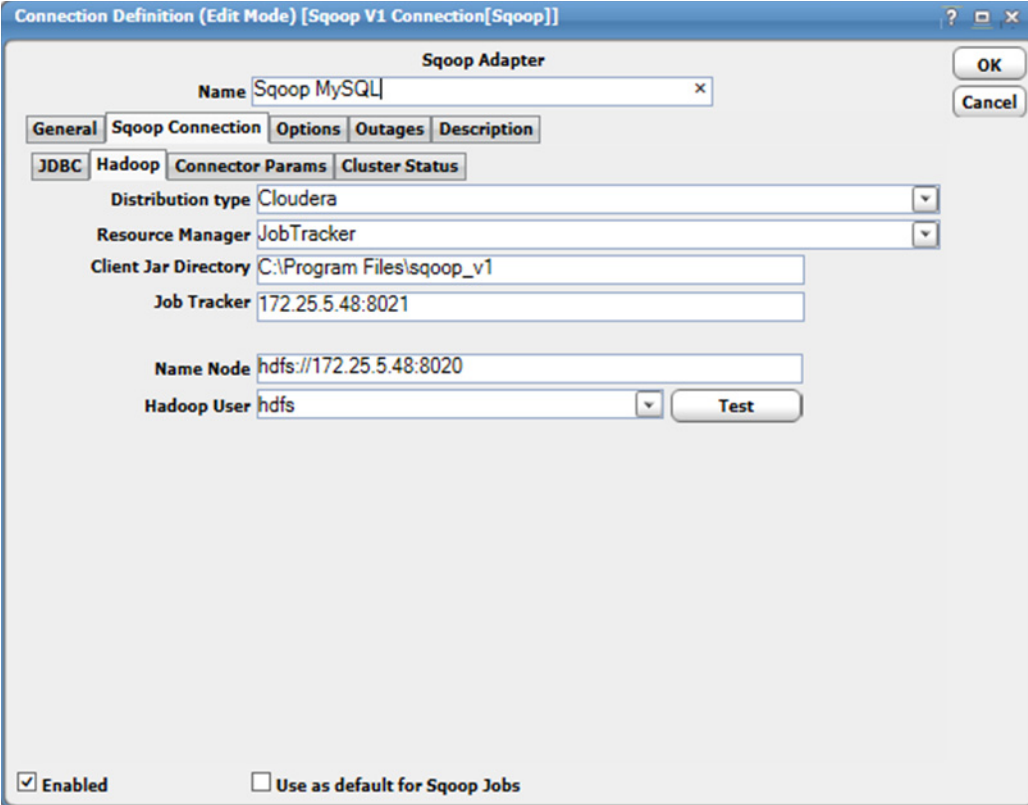
Only authorized users that have been defined with Sqoop passwords display in this list. The selected user is automatically supplied as the default runtime user in a new Sqoop Adapter job definition.

6. Click the **Sqoop Connection** tab.



7. On the JDBC page, select a template from the **JDBC Template** list.
8. The **JDBC Driver** field is automatically populated with the JDBC driver's fully qualified class name and the **Port** field is automatically populated with default value. This driver class must exist in the CLASSPATH on TWA Master and in HADOOP_CLASSPATH on the Hadoop Cluster.
9. In the **JDBC Host** field, enter the database host machine name.
10. In the **Port** field, enter the port on which the database is accepting JDBC connection.
11. From the **Connect User** list, select the user that will connect to the database.
12. In the **SID/Service** field, enter the SID or Service Name. This field will be enabled for Oracle databases.
13. From the **Database/DSN** list, select the database for this connection.
14. If you want to override the default JDBC Connection String, select the **JDBC Connection String option (URL)** option, and then enter the JDBC driver connection URL as specified in database driver's documentation.
15. On the **Sqoop Connection** tab, click the **Hadoop** tab. The tab contents are different depending on whether the connection is for Hadoop 1 or Hadoop 2 as shown in the two examples here:

Sqoop connection definition for Hadoop 1:



Connection Definition (Edit Mode) [Sqoop V1 Connection[Sqoop]]

Sqoop Adapter

Name: Sqoop MySQL

General | Sqoop Connection | Options | Outages | Description

JDBC | Hadoop | Connector Params | Cluster Status

Distribution type: Cloudera

Resource Manager: JobTracker

Client Jar Directory: C:\Program Files\sqoop_v1

Job Tracker: 172.25.5.48:8021

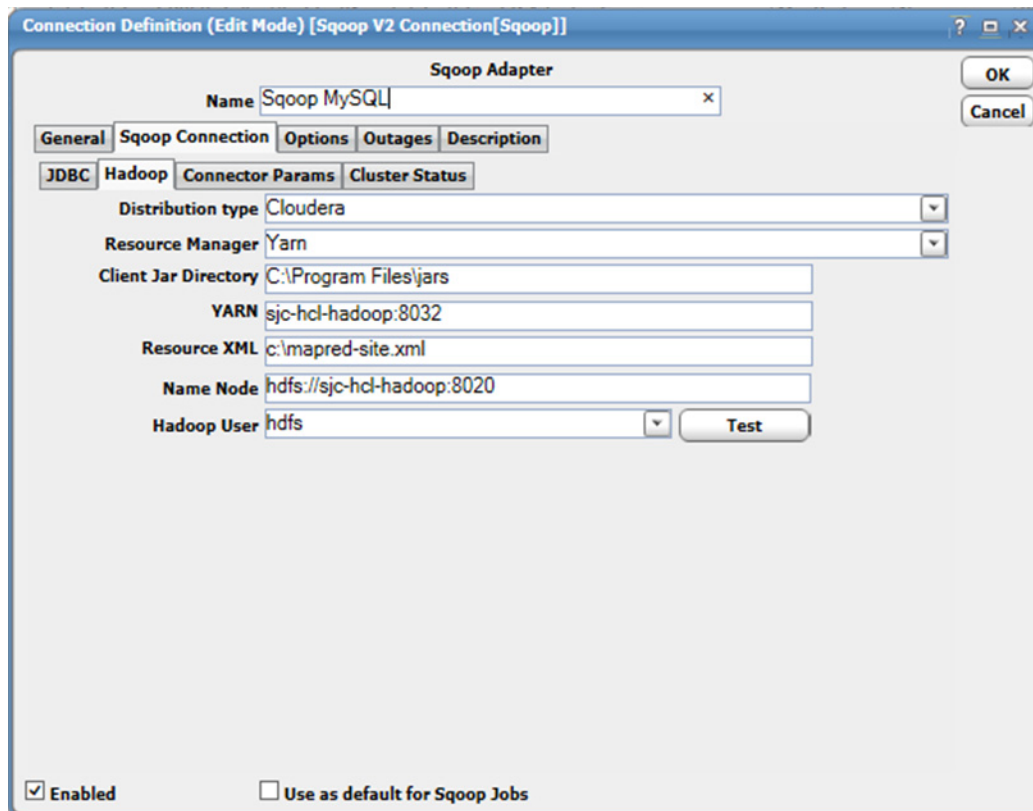
Name Node: hdfs://172.25.5.48:8020

Hadoop User: hdfs

Test

Enabled Use as default for Sqoop Jobs

Sqoop connection definition for Hadoop 2:



16. Enter the input fields depending on whether the connection is for Hadoop 1 or Hadoop 2. These fields are required:

- **Distribution Type**—Select the Hadoop distribution that you are using.
- **Resource Manager**—Select **YARN** for Hadoop 2 and **Job Tracker** for Hadoop 1.
- **Client Jar Directory**—Enter the path to the directory where all the Hadoop client libraries reside. See [Installing the Hadoop Client Libraries, page 11](#) for instructions on obtaining these client jar files.
- **YARN** (Hadoop 2 only)—Enter the hostname and port at which YARN is running.
- **Resource XML** (Hadoop 2 only)—Get the mapred-site.xml from your Hadoop installation, copy the file to the TWA Master, and provide the path here.
- **Job Tracker** (Hadoop 1 only)—Enter the location of your Job Tracker.
- **Name Node**—Enter the URI of the Name node.

Note: For MapR, **Job Tracker** and **Name Node** must be set to "maprfs:///".

Note: Ensure that the mapred-site.xml file defines the following properties:

```
mapreduce.application.classpath
yarn.application.classpath
mapreduce.jobhistory.address
mapreduce.jobhistory.intermediate-done-dir
yarn.app.mapreduce.am.staging-dir
mapreduce.app-submission.cross-platform
```

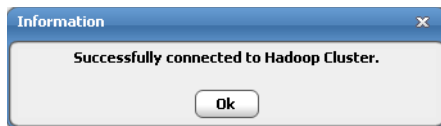
When the TWA Master is running on Windows, mapreduce.app-submission.cross-platform should be set to true.

17. From the **Hadoop User** list, select the associated Runtime User for Sqoop to be used to execute Sqoop tools.

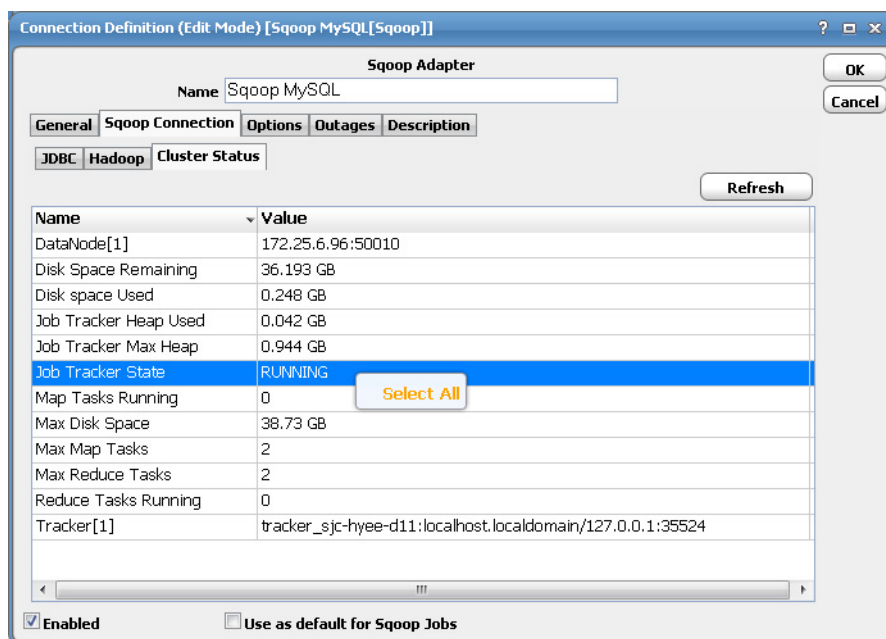
This is a persistent user connection that is only used for administration and monitoring and for jobs with a matching runtime user.

Note: It is recommended that the connection's Hadoop user be a Hadoop Super User and is a requirement to display Distributed File System statistics in the Cluster Status tab.

18. Click the **Test** button to verify connectivity. If successful, the following message displays:



19. Click the **Cluster Status** tab to display current cluster's status in real time. This is for the display of Distributed File System (DFS) info and requires a Hadoop Super User.



Click the **Refresh** button to refresh the status information.

20. Click the **Options** tab to specify Global Job Parameters that are applicable to all jobs using the connection.

Connection Definition (Edit Mode) [Sqoop MySQL[Sqoop]]

Sqoop Adapter

Name: Sqoop MySQL

OK Cancel

General Sqoop Connection Options Outages Description

Polling Interval (in seconds)

Connection Poll: 5

Global Job Parameters

Name	Value
dfs.data.dir	/home/sacpawar/hadoop_storage/data
dfs.name.dir	/home/sacpawar/hadoop_storage/name
hadoop.tmp.dir	/home/sacpawar/hadoop_storage/tmp

Add Edit Delete

Enabled Use as default for Sqoop Jobs

21. To add a parameter, click **Add** to display the **Parameter Definition** dialog box.

22. Click **OK** to save the new Sqoop connection. The configured connection displays in the **Connections** pane.

The status light next to the connection indicates whether the TWA Master is connected to the Database using JDBC and to Hadoop Cluster. If the light is green, the TWA Master is connected to both Database and Hadoop cluster.

A red light indicates that the master cannot connect to the Database, Hadoop Cluster, or both. Sqoop jobs will not be submitted without a working Sqoop connection. You can only define jobs from the Client if the connection light is green.

3

Using the Sqoop Adapter

Overview

This chapter describes how to use the Sqoop Adapter:

- [Defining Sqoop Jobs](#) - Describes how to define code generation, export, import, and merge jobs.
- [Monitoring Sqoop Job Activity](#) - Describes how to monitor Sqoop jobs.
- [Controlling Adapter and Agent Jobs](#) - Describes how to perform typical tasks to control Sqoop jobs.

Defining Sqoop Jobs

This section provides instructions for defining a Sqoop job in TWA and descriptions of the various options that can be included in the jobs. The Sqoop Adapter allows for the definition of the following job tasks:

- **Code Generation** - This task generates Java classes which encapsulate and interpret imported records. The Java definition of a record is instantiated as part of the import process, but can also be performed separately. If Java source is lost, it can be recreated using this task. New versions of a class can be created which use different delimiters between fields or different package name.
- **Export** - The export task exports a set of files from HDFS back to an RDBMS. The target table must already exist in the database. The input files are read and parsed into a set of records according to the user-specified delimiters. The default operation is to transform these into a set of INSERT statements that inject the records into the database. In "update mode," Sqoop will generate UPDATE statements that replace existing records in the database.
- **Import** - The import tool imports structured data from an RDBMS to HDFS. Each row from a table is represented as a separate record in HDFS. Records can be stored as text files (one record per line), or in binary representation such as Avro or SequenceFiles.
- **Merge** - The merge tool allows you to combine two datasets where entries in one dataset will overwrite entries of an older dataset. For example, an incremental import run in last-modified mode will generate multiple datasets in HDFS where successively newer data appears in each dataset. The merge tool will "flatten" two datasets into one, taking the newest available records for each primary key.

This can be used with both SequenceFile-, Avro- and text-based incremental imports. The file types of the newer and older datasets must be the same. The merge tool is typically run after an incremental import with the date-last-modified mode.

To define a Sqoop job:

1. In the **Navigator** pane, select **Definitions>Jobs** to display the **Jobs** pane.
2. Right-click **Jobs** and select **Add Job>Sqoop Job** from the context menu.

The **Sqoop Job Definition** dialog box displays.

The **Run** tab is selected by default. You must first specify a name for the job, the Sqoop Adapter connection that will be used for the job and a valid runtime user who has the appropriate Sqoop authority for the report being scheduled.

3. In the upper portion of the dialog box, specify the following information to describe the job:
 - **Job Name** – Enter a name that describes the job.
 - **Job Class** – If you want to assign a defined job class to this job, select it from the drop-down list. This field is optional.
 - **Owner** – Select the Sqoop owner of the selected job. The user must have the appropriate Sqoop authority for the operation.
 - **Parent Group** – If this job exists under a parent group, select the name of the parent group from the drop-down list. All properties in the **Agent Information** section are inherited from its parent job group.
4. Specify the following connection information in the **Agent/Adapter Information** section:
 - **Agent/Adapter Name** – Select the Sqoop Adapter connection to be used for this job from the drop-down list.
 - or–
 - Agent List Name** – Select a list for broadcasting the job to multiple servers.
 - **Runtime User** – Select a valid runtime user with the appropriate Sqoop authority for the job from the drop-down list.
5. Specify the appropriate Tracking and Duration information for the job. Refer to the *Tidal Workload Automation User Guide* for information on these options.
6. Click the **Sqoop** tab, to specify the job configuration.
7. From the **Sqoop Task** list, select a job task type and enter the task details as described in these sections:
 - [Code Generation Task](#)
 - [Export Job Task](#)
 - [Import Job Task](#)
 - [Merge Job Task, page 36](#)
8. Click **OK** to save the job.

Code Generation Task

The screenshot shows the 'Sqoop Job Definition' dialog box. At the top, the 'Sqoop Job Name' is 'Sqoop.Code Gen Job', 'Job Class' is empty, 'Parent Group' is empty, and 'Owner' is 'Schedulers'. Below this is a tabbed interface with 'Sqoop' selected. Under 'Sqoop Task', 'Code Generation' is selected. The 'Details' tab is active, showing fields for 'Source Table' (foldermst), 'Binary Output Directory' (tes\sqoop\output\binary), 'Class Name' (Class Name), 'Source Output Directory' (tes\sqoop\output\source), and 'Package Name' (codegen.pkg). There is a 'Variables' button and an 'Enabled' checkbox checked at the bottom.

Details Tab

- **Source Table** - Select the Source table for the task.
- **Binary Output Directory** - Output directory for compiled objects
- **Class Name** - Sets the generated class name. This overrides package-name. When combined with jar-file, sets the input class.
- **Source Output directory** - Output directory for generated code
- **Package Name** - Put auto-generated classes in this package

Other Tab

Click this tab to define output formatting and input parsing elements and code generation related parameters.

Sqoop Job Definition

Sqoop Job Name: Sqoop_Code Gen Job [OK]

Job Class: [] Owner: Schedulers [Cancel]

Parent Group: []

Sqoop Schedule Run Dependencies Resources Job Events Options Run Book Notes Images

Sqoop Task: Code Generation

Details Other

Output Formatting

Use Default (MySQL) Delimiters
(columns: , lines: \n escaped-by: \ enclosed-by: ')

Column Delimiters: [] Line Delimiters: []

Enclosed By: [] Escaped By: []

Optionally Enclosed By: []

Input Parsing

Column Delimiters: [] Line Delimiters: []

Enclosed By: [] Escaped By: []

Optionally Enclosed By: []

Variables []

Enabled

In the **Output formatting** section, enter the following:

- **Use default delimiters** - Select this option to use MySQL's default delimiter set. (fields: , lines: \n escaped-by: \ optionally-enclosed-by: ')
- **Column delimiters** - Sets the field separator character
- **Line delimiters** - Sets the end-of-line character
- **Enclosed by** - Sets a required field enclosing character
- **Escaped by** - Sets the escape character
- **Optionally enclosed by** - Sets a field enclosing character

In the **Input Parsing** section, enter the following:

- **Column delimiters** - Sets the input field separator
- **Line Delimiters** - Sets the input end-of-line character
- **Enclosed by** - Sets a required field encloser
- **Escaped by** - Sets the input escape character
- **Optionally enclosed by** - Sets a field enclosing character

Export Job Task

The screenshot shows the 'Sqoop Job Definition' dialog box with the following fields and options:

- Sqoop Job Name:** Export data to db using existing jar file
- Job Class:** (empty dropdown)
- Owner:** gatest
- Parent Group:** (empty dropdown)
- Task Type:** Sqoop Task: Export
- Source Path:** /user/sacpawar/bizunit_1
- Destination Table:** bizunit_1
- Existing JAR File Path:** /tmp/sqoop-sacpawar/compile/ad2a3481086a649cf
- Allow Updates:** (updateonly dropdown) (Insert Only if unchecked)
- Key Columns for Updates:** (empty text box)
- Use Batch Mode:**
- Use High Performance direct Export:**
- Number of Mappers:** (empty dropdown)
- Staging Table:** (empty text box) Clear Staging Table
- Variables:** (button)
- Enabled:**
- Last Modified:** 08/28/2012 18:27:44

Details Tab

Note: You can click the Variables button to insert a predefined variable into a selected field on this tab.

- **Source Path** - HDFS source path for the export that contains the source data.
 - **Destination Table** - Select the table you want to use to populate in the database.
 - **Existing Jar File** - Enter the name of the jar to export the record class from.
 - **Allow Updates** - Depending on the target database, you can update rows if they exist in the database already or insert rows if they do not exist yet. By default, the export is carried out using INSERT statements. By checking the **Allow Updates** option, you can choose to use UPDATE statements or INSERT and UPDATE statements together. The list box next to the **Allow Updates** option allows you to select the desired update mode.
 - **Key Columns for Updates** - Anchor columns to use for updates. Use a comma separated list of columns if there is more than one column. This is a mandatory field if **Allow Updates** is selected.
 - **Use Batch Mode** - Use batch mode for underlying statement execution.
 - **Use High Performance Direct Export** - Select to use the direct export fast path.
- MySQL provides a direct mode for exports as well, using the **mysqlimport** tool. This may be higher-performance than the standard JDBC codepath. When using export in direct mode with MySQL, the MySQL bulk utility **mysqlimport** must be available in the shell path of the task process.
- **Number of Mappers** - Select the number of map tasks to export in parallel.

- **Staging Table** - The table in which data will be staged before being inserted into the destination table. Support for staging data prior to pushing it into the destination table is not available for high performance direct exports. It is also not available when export is invoked using key columns for updating existing data.
- **Clear Staging Table** - Indicates that any data present in the staging table can be deleted.

Other Tab

Click this tab to define output formatting and input parsing elements and code generation related parameters.

Note: You can click the Variables button to insert a predefined variable into a selected field on this tab.

Sqoop Job Definition [Export Sqoop Job Regression]

Sqoop Job Name: Export Sqoop Job Regression

Job Class: [Dropdown]

Owner: qatest

Parent Group: [Dropdown]

Sqoop Task: Export

Output Formatting

Use Default (MySQL) Delimiters
(columns: , lines: \n escaped-by: \ enclosed-by: ')

Column Delimiters: [Text Box]

Line Delimiters: [Text Box]

Enclosed By: [Text Box]

Escaped By: [Text Box]

Optionally Enclosed By: [Text Box]

Input Parsing

Column Delimiters: [Text Box]

Line Delimiters: [Text Box]

Enclosed By: [Text Box]

Escaped By: [Text Box]

Optionally Enclosed By: [Text Box]

CodeGen

Binary Output Directory: /tmp/sqoop-.../compile/...

Class Name: bizunit_1

Source Output Directory: /tmp/sqoop-.../compile/...

Package Name: [Text Box]

Variables

Enabled

Last Modified : 08/28/2012 18:50:45

In the **Output formatting** section, enter the following:

- **Use default delimiters** - Select this option to use MySQL's default delimiter set. (fields: , lines: \n escaped-by: \ optionally-enclosed-by: ')
- **Column delimiters** - Sets the field separator character
- **Line delimiters** - Sets the end-of-line character
- **Enclosed by** - Sets a required field enclosing character
- **Escaped by** - Sets the escape character
- **Optionally enclosed by** - Sets a field enclosing character

In the **Input Parsing** section, enter the following:

- **Column delimiters** - Sets the input field separator

- **Line Delimiters** - Sets the input end-of-line character
- **Enclosed by** - Sets a required field enclosure
- **Escaped by** - Sets the input escape character
- **Optionally enclosed by** - Sets a field enclosing character

In the **CodeGen** section, enter the following:

- **Binary Output Directory** - Output directory for compiled objects
- **Class Name** - Sets the generated class name. This overrides package-name.
- **Source Output Directory** - Output directory for generated code
- **Package Name** - Put auto-generated classes in this package

Import Job Task

The screenshot shows the 'Sqoop Job Definition' dialog box with the following configuration:

- Sqoop Job Name:** import a table with bin and src dirs
- Job Class:** (empty dropdown)
- Owner:** qatest
- Parent Group:** (empty dropdown)
- Sqoop Task:** Import
- Source:** Options, Other
- Import Source:**
 - All Tables
 - Table/View: bizunit_1
- Columns (List of columns separated by ","):** (empty text box)
- Where:** bizunit_id < 3
- QUERY
- Import Destination:**
 - Append
 - File Format:**
 - Plain Text
 - SequenceFile
 - Avro Data File
 - Destination:**
 - Target Directory
 - Warehouse Directory
- Variables:** (button)
- Enabled
- Last Modified:** 08/27/2012 19:17:08

Source Tab

Note: You can click the Variables button to insert a predefined variable into a selected field on this tab.

- **All Tables** - This tool imports a set of tables from an RDBMS to HDFS. Data from each table is stored in a separate directory in HDFS.

For the import-all-tables tool to be useful, the following conditions must be met:

- Each table must have a single-column primary key.
- You must intend to import all columns of each table.
- You must not intend to use non-default splitting column, nor impose any conditions via a WHERE clause.
- **Table/View** – Sqoop typically imports data in a table-centric fashion. This field can also identify a view or other table-like entity in a database. By default, all columns within a table are selected for import. You can select a subset of columns and control their ordering by using the selected **Column** grid you can control which rows are imported by adding a SQL WHERE clause to the import statement.
- **Query** – Sqoop can also import the result set of an arbitrary SQL query.

When importing a free-form query, you must specify a destination directory on the **Destination** tab.

The facility of using free-form query in the current version of Sqoop is limited to simple queries where there are no ambiguous projections and no OR conditions in the WHERE clause. Use of complex queries such as queries that have sub-queries or joins leading to ambiguous projections can lead to unexpected results.

- **Append** – By default, imports go to a new target location. If the destination directory already exists in HDFS, Sqoop will refuse to import and overwrite that directory's contents. If you check this check box, Sqoop will import data to a temporary directory and then rename the files into the normal target directory in a manner that does not conflict with existing filenames in that directory.
- **File Formats** – Sqoop supports the following file formats:
 - **Plain Text** – Delimited text is the default import format. This argument will write string-based representations of each record to the output files, with delimiter characters between individual columns and rows. Delimited text is appropriate for most non-binary data types. It also readily supports further manipulation by other tools, such as Hive.
 - **Sequence File** – This is binary format that store individual records in custom record-specific data types. These data types are manifested as Java classes. Sqoop will automatically generate these data types for you. This format supports exact storage of all data in binary representations, and is appropriate for storing binary data. Reading from SequenceFiles is higher-performance than reading from text files, as records do not need to be parsed.
 - **Avro File** – Avro data files are a compact, efficient binary format that provides interoperability with applications written in other programming languages. Avro also supports versioning, so that when, e.g., columns are added or removed from a table, previously imported data files can be processed along with new ones.
- **Destination** directory can be specified in 2 ways
 - **Target Directory** – HDFS destination directory. This field is mandatory when importing a free-form query OR
 - **Warehouse Directory** – You can adjust the parent directory of the import with this location set. By default, Sqoop will import a table named foo to a directory named foo inside your home directory in HDFS. For example, if your username is *someuser*, then the import tool will write to **/user/someuser/foo/(files)**. You can adjust the parent directory of the import by using this field.

Options Tab

Note: You can click the Variables button to insert a predefined variable into a selected field on this tab.

- **Split By Column** – Column of the table used to split work units. If you want to import the results of a query in parallel, then each map task will need to execute a copy of the query, with results partitioned by bounding conditions inferred by Sqoop. Your query must include the token \$CONDITIONS which each Sqoop process will replace with a unique condition expression. You must also select a splitting column with this field.

Since result of a free form query does not have a primary key it is mandatory to provide a split by column while importing the data using free form query.

This field is disabled for Import of all tables from database.

If the actual values for the primary key are not uniformly distributed across its range, then this can result in unbalanced tasks. You should explicitly choose a different column using this field. Sqoop cannot currently split on multi-column indices. If your table has no index column, or has a multi-column key, then you must also manually choose a splitting column.

- **Compress** –Enable compression. By default, data is not compressed. You can compress your data by using the deflate (gzip) algorithm with this check box checked.
- **Compression Codec** – User can specify any Hadoop compression codec using field. This applies all file formats supported by sqoop
- **Substitute Null with** – the value you want to substitute for null.
- **For String** –The string to be written for a null value for string columns
- **For Non-String** –The string to be written for a null value for non-string columns. If not specified, then the string " null" will be used.
- **Number of Mappers** –Number of Map tasks to import in parallel. Sqoop imports data in parallel from most database sources. You can specify the number of map tasks (parallel processes) to use to perform the import by using this field. It takes an integer value which corresponds to the degree of parallelism to employ. By default, four tasks are used. Some databases may see improved performance by increasing this value to 8 or 16.

Do not increase the degree of parallelism greater than that available within your MapReduce cluster; tasks will run serially and will likely increase the amount of time required to perform the import. Likewise, do not increase the degree of parallelism higher than that which your database can reasonably support. Connecting 100 concurrent clients to your database may increase the load on the database server to a point where performance suffers as a result.

- **Inline LOB Limit** – Sqoop handles large objects (BLOB and CLOB columns) in particular ways. If this data is truly large, then these columns should not be fully materialized in memory for manipulation, as most columns are. Instead, their data is handled in a streaming fashion. Large objects can be stored inline with the rest of the data, in which case they are fully materialized in memory on every access, or they can be stored in a secondary storage file linked to the primary data storage. By default, large objects less than 16 MB in size are stored inline with the rest of the data. At a larger size, they are stored in files in the `_lobs` subdirectory of the import target directory. These files are stored in a separate format optimized for large record storage, which can accommodate records of up to 2^{63} bytes each. The size at which lob spill into separate files is controlled by the `--inline-lob-limit` argument, which takes a parameter specifying the largest lob size to keep inline, in bytes. If you set the inline LOB limit to 0, all large objects will be placed in external storage. For high performance direct imports this field will be disabled and will not be passed over to sqoop tool.
- **Use High Performance Direct Import channel** – Uses direct import fast path if selected. By default, the import process will use JDBC which provides a reasonable cross-vendor import channel. Some databases can perform imports in a more high-performance fashion by using database-specific data movement tools. For example, MySQL provides the `mysqldump` tool which can export data from MySQL to other systems very quickly. You are specifying that Sqoop should attempt the direct import channel. This channel may be higher performance than using JDBC. Currently, direct mode does not support imports of large object columns.

Sqoop's direct mode does not support imports of BLOB, CLOB, or LONGVARBINARY columns. Use JDBC-based imports for these columns. It is user's responsibility to not select the direct mode of import if these kinds of columns are involved in import.

- **Split size for direct Import** – Split the input stream every `n` bytes when importing in direct mode. When importing from PostgreSQL in conjunction with direct mode, you can split the import into separate files after individual files reach a certain size. This size limit is controlled with this field.
- **Incremental Import** – Sqoop provides an incremental import mode which can be used to retrieve only rows newer than some previously-imported set of rows. There are two modes
 - **Append** – You should specify append mode when importing a table where new rows are continually being added with increasing row id values.
 - **Last modified** – You should use this when rows of the source table may be updated, and each such update will set the value of a last-modified column to the current timestamp. Rows where the check column holds a timestamp more recent than the timestamp specified with last value field are imported.
- **Check column** – Specifies the column to be examined when determining which rows to import during incremental import.
- **Last Value** – Specifies the maximum value of the check column from the previous import during incremental import.
- **Use boundary Query** – Boundary query to use for creating splits. By default sqoop will use query `select min (<split-by>), max (<split-by>) from <table name>` to find out boundaries for creating splits. In some cases this query is not the most optimal so you can specify any arbitrary query returning two numeric columns using `boundary-query`.

Other Tab

Note: You can click the Variables button to insert a predefined variable into a selected field on this tab.

In the **Output Formatting** section, enter the following:

- **Use Default Delimiters** - Select this option to use MySQL's default delimiter set. (fields: , lines: \n escaped-by: \ optionally-enclosed-by: ')
- **Column Delimiters** - Sets the field separator character
- **Line Delimiters** - Sets the end-of-line character
- **Enclosed By** - Sets a required field enclosing character
- **Escaped By** - Sets the escape character
- **Optionally Enclosed By** - Sets a field enclosing character

In the **Input Parsing** section, enter the following:

- **Column Delimiters** - Sets the input field separator
- **Line Delimiters** - Sets the input end-of-line character
- **Enclosed By** - Sets a required field encloser
- **Escaped By** - Sets the input escape character
- **Optionally Enclosed By** - Sets a field enclosing character

In the **CodeGen** section, enter the following:

- **Binary Output Directory** - Output directory for compiled objects

- **Class Name** - Sets the generated class name. This overrides package-name.
- **Source Output Directory** - Output directory for generated code
- **Existing Jar File Path** - Enter the path of the jar to reuse for import. This disables code generation.
- **Package Name** - Put auto-generated classes in this package

Merge Job Task

Sqoop Job Definition [Merge job]

Sqoop Job Name: Merge job OK

Job Class: ▼ Owner: qatest Cancel

Parent Group: ▼

Sqoop Schedule Run Dependencies Resources Job Events Options Run Book Notes History Images

Sqoop Task: Merge ▼

Details

Class Name: bizunit_new

Existing JAR File Path: /home/sacpawar/tmp/new/bizunit_new.jar

Primary Key Column: bizunit_id

Newer Dataset Path: /user/james/test_2

Older Dataset Path: /user/sacpawar/test

Output Path: /user/james/bizunit_test_merged

Variables ▼

Enabled Last Modified : 08/27/2012 19:07:44

Details Tab

Note: You can click the Variables button to insert a predefined variable into a selected field on this tab.

- **Class Name** - the name of the record-specific class to use during the merge job
- **Existing JAR File Path** - the name of the jar to load the record class from.
- **Primary Key Column** - The name of a column to use as the merge key. When merging the datasets, it is assumed that there is a unique primary key value in each record.
- **Newer Dataset Path** - the path of the newer dataset.
- **Older Dataset Path** - the path of the older dataset.
- **Output Path** - the target path for the output of the merge job.

Monitoring Sqoop Job Activity

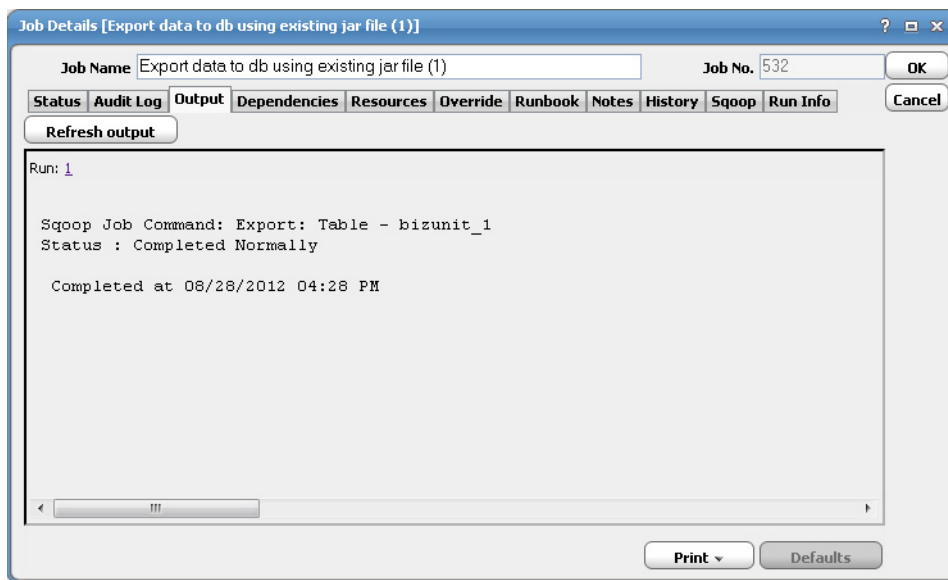
As Sqoop tasks run as pre-scheduled or event-based jobs, you can monitor the jobs as you would any other type of job in TWA using the **Job Details** dialog box. You can also use Business Views to monitor job activity and view when the jobs are active (see the *Tidal Workload Automation User Guide* for instructions on using Business Views).

To monitor job activity:

1. In the **Navigator** pane, select **Operations>Job Activity** to display the **Job Activity** pane.
2. Right-click to select a job and choose **Details** from the context menu.

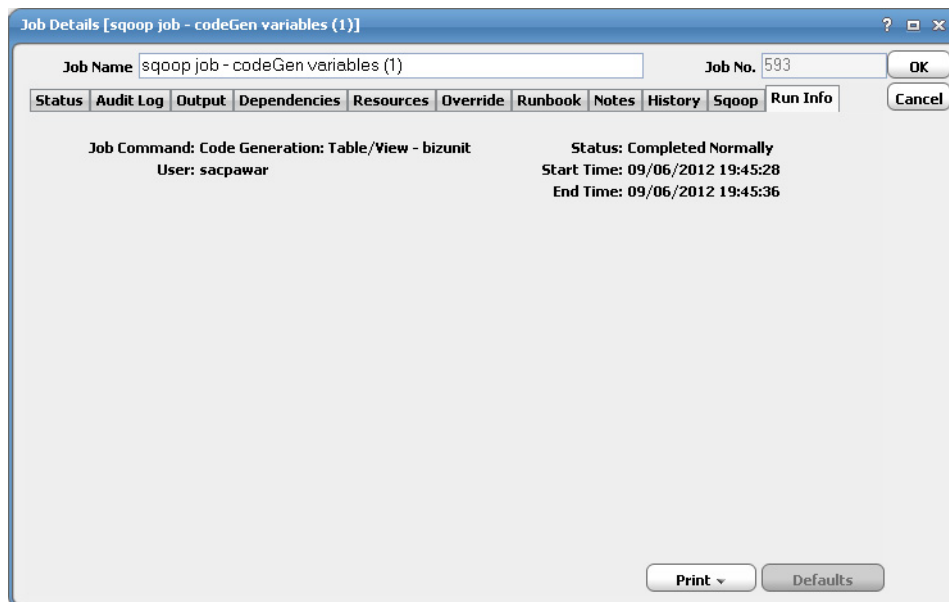
The **Job Details** dialog box displays. The **Status** page displays by default. You can view the status of the job, the start and end time, how long it ran, and how it was scheduled. The external ID is the Sqoop job number.

3. Click the **Output** tab to view a task summary after the job completes.

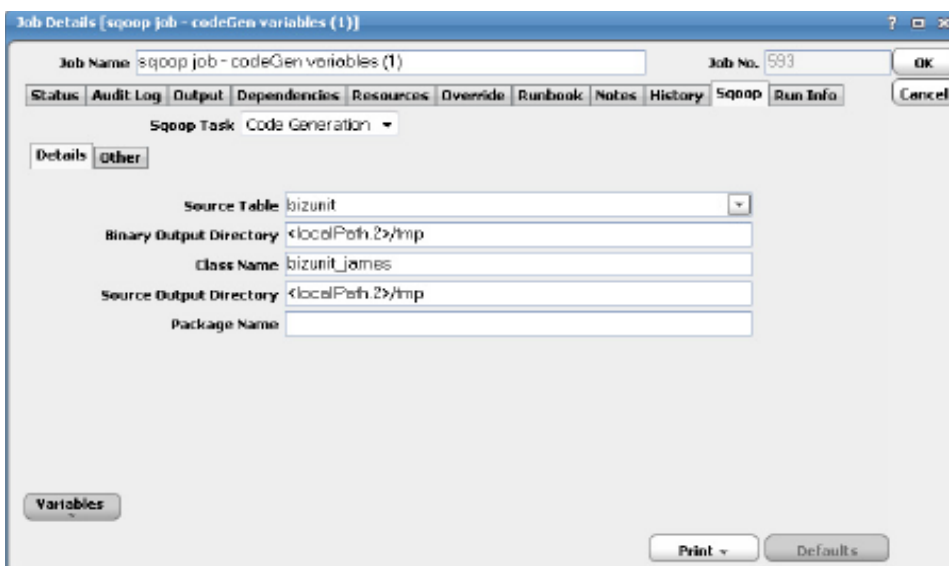


4. Click the **Run Info** tab to view additional details about the job. You can also view this tab to view information about the runtime status while the job is running, including any messages.

For example, the following figures depict the **Run Info** tab for a Code Generation task.



5. Click the **Sqoop** tab to view the job definition details and the variables that were used when the job was submitted.



While the job is running, the fields are disabled; however, prior to running or rerunning the job, you can override any value on this screen. Your changes here only apply to this instance of the job (the original job definition is not affected).

6. When you have finished viewing the job activity details, click **OK** to close the dialog box.

Controlling Adapter and Agent Jobs

Scheduler provides the following job control capabilities for either the process currently running or the job as a whole:

- **Holding a Job**—Hold a job waiting to run.

- [Aborting a Job](#)—Abort an active job.
- [Rerunning a Job](#)—Rerun a job that completed.
- [Making One Time Changes to an Adapter or Agent Job Instance](#)—Make last minute changes to a job.
- [Deleting a Job Instance before It Has Run](#)—Delete a job instance before it has run.

Holding a Job

Adapter/agent jobs are held in the same way as any other Scheduler jobs.

Adapter/agent jobs can only be held before they are launched. Once a job reaches the Adapter/Agent system, it cannot be held or suspended.

To hold a job:

1. From the **Job Activity** pane, right-click on the job.
2. Select **Job Control>Hold/Stop**.

Aborting a Job

Adapter/agent jobs are aborted in the same way as any other Scheduler jobs.

To abort a job:

1. From the **Job Activity** pane, right-click on the job.
2. Select **Job Control>Cancel/Abort**.

Rerunning a Job

On occasion, you may need to rerun an Adapter/Agent job. You can override parameter values first, if necessary, from the Adapter/Agent tab.

To rerun a job:

1. From the **Job Activity** pane, right-click the Adapter/Agent job you need to rerun.
2. Select **Job Control>Rerun** option from the context menu.

Making One Time Changes to an Adapter or Agent Job Instance

Prior to a run or rerun, you can edit data on the specific **Adapter/Agent** tab. To ensure that there is an opportunity to edit the job prior to its run, you can set the **Require operator release** option on the **Options** tab in the Adapter **Job Definition** dialog. Use this function to make changes to an Adapter job after it enters Waiting on Operator status as described in the following procedure.

To make last minute changes:

1. From the **Job Activity** pane, double-click the Adapter/Agent job to display the **Job Details** dialog.
2. Click the Adapter tab.
3. Make the desired changes to the job and click **OK** to close the **Job Details** dialog.
4. If this job is Waiting on Operator, perform one of the following tasks:

- To release the job, select **Job Control->Release**.
- To rerun the job with changes, select **Job Control->Rerun**.

Deleting a Job Instance before It Has Run

Adapter/Agent job instances are deleted in the same way as any other Scheduler job.

Deleting a job from the **Job Activity** pane removes the job from the Scheduler job activity only. The original definition is left in tact.

To delete a job instance:

1. From the **Job Activity** pane, right-click the Adapter/Agent job to be deleted.
2. Select **Remove Job(s) From Schedule**.

4

Troubleshooting the Sqoop Adapter

Overview

This chapter describes how to troubleshoot some issues you might run into:

- [Review Service Log files for More Information](#)
- [Connection Failures](#)
- [Job Failures](#)
- [Adapter Is Out-of-Memory](#)

Review Service Log files for More Information

Refer to the log files for further information regarding an issue.

Connection Failures

- Make sure that on the TWA Master machine JAVA_HOME is pointing to JDK directory and not to a JRE directory.
- If you are using MapR, verify that the Hadoop instance is running using Java 8. Also verify that the MapR client is configured correctly.
- If running TWA on Windows with the JDK set up correctly, the following error may display:

CommonsLogger: CompilationManager

Suggested fix:

Install a JDK and set \$JAVA_HOME to use it.

This is seen for JDK 1.6 update 34 onwards. The installation of JDK adds C:\windows\system32\java.exe file. Rename/Remove this file and restart the TWA master.

- java.net.ConnectException: Connection refused. This error can be seen in the log SqoopService.out file after "Initiating client connection, ConnectString=127.0.0.1:5181 OR localhost:5181"

This error is caused by the Zookeeper setting on the cluster; Zookeeper is set up on the cluster with either localhost or 127.0.0.1. The solution is to set up Zookeeper with IP address or fully qualified hostname as described in [Setting Up Zookeeper on the Cluster](#).

Setting Up Zookeeper on the Cluster

To set up Zookeeper on the cluster:

1. Edit /opt/mapr/conf/mapr-clusters.conf. Change Zookeeper server setting to use IP address or fully qualified hostname.

2. Edit `/opt/mapr/conf/cldb.conf`. Change Zookeeper server setting to use IP address or fully qualified hostname.
3. Edit `/opt/mapr/conf/warden.conf`. Change Zookeeper server setting to use IP address or fully qualified hostname.
4. Restart Zookeeper.
5. Optionally restart the warden (`sudo /etc/init.d/mapr-warden restart`).

Job Failures

- Verify Job is configured correctly.
- Verify the job can be run via the Sqoop CLI before running thru TWA.
- Check adapter logs and verify how it ran from the Hadoop Admin Console.
- The file paths and names are case sensitive and they must exist on the HDFS.
- Sqoop job failure on MapR Distribution of Hadoop due to Permission Denied error

CommonsLogger: ImportTool - Encountered IOException running import job: java.io.IOException: Error: 13:Permission denied(13), file: /var/mapr/cluster/mapred/jobTracker/staging/<username>/staging at com.mapr.fs.MapRFileSystem.makeDir(MapRFileSystem.java:448)

Suggested Fix:

If the TWA Master is running on Linux, verify it is running using the same user used to set up Sqoop on the cluster.

If the TWA Master is running on Windows, verify the MapR Client is set up using the "Spoofed user" that is same as the one used to setup Sqoop on the cluster.

Log on to Hadoop cluster and delete `/var/mapr/cluster/mapred/jobTracker/staging/<username>/` directory from the HDFS and try executing the job again.

Adapter Is Out-of-Memory

- Adapter memory sizes are verified on a 10-node cluster and can be increased in the regular way within the Adapter *service.props*.
- Output files cannot be viewed.

5

Configuring service.props

About Configuring service.props

The **service.props** file is used to configure adapter behavior. **service.props** is located in the \config directory located under the Adapter's GUID directory, You can create both the directory and file if it does not yet exist. Properties that can be specified in service.props control things like logging and connection configuration. Many of the properties are specific to certain adapters; others are common across all adapters.

service.props Properties

The table below lists many of the parameters that can be specified in service.props. Some properties apply to all adapters (shaded in the table) and some properties are adapter-specific as indicated by the **Applicable Adapter(s)** column. The properties are listed in alphabetical order.

Table 1

Property	Applicable Adapter(s)	Default	What It Controls
BYPASS_SEC_VALIDATION	Oracle Apps	N	If set to Y, the secondary user validation is bypassed. If not, secondary user validation is performed.
CLASSPATH	All	<none>	(Optional) - The path to the JDBC driver. If the default CLASSPATH used when the Adapter process is started does not include an appropriate JDBC driver jar required to connect to the PowerCenter Repository Database, you will need to specify this <i>service.props</i> configuration
CONN_SYNC	Informatica, Oracle Apps, SAP	N	Setting this flag to Y allows synchronous connections without overloading the RDOOnly Thread. If set to N, the adapter might stop trying to reconnect after an outage or downtime.
DISCONN_ON_LOSTCONN	Informatica	N	Setting this flag to Y avoids an unnecessary logout call to the Informatica server when the connection is lost. This logout call usually hangs.
EnableDynamicPollingInterval	All	N	Use to avoid frequent polling on long-running jobs. When set to Y in service.props of a particular adapter, these properties are enabled: MinDynamicPollInterval—Minimum value should be 5 seconds. MaxDynamicPollIntervalInMin—Maximum value should be 5 minutes. PercentOfEstDuration—Default value is 5.
HADOOP_JAVA_HOME	Sqoop	<none>	If the Java version used in the Hadoop environment is lower than Java 8, then install the same lower JDK version in the in the Master and include the path of the JDK in this property.

Table 1

Property	Applicable Adapter(s)	Default	What It Controls
IGNORE_CODES	Informatica	<none>	This parameter can be set in service.props, job configuration and connection configuration parameters. The order of precedence is service.props (applicable for all jobs running in all connections), job level (only for that particular job), and connection (applicable for all jobs in the connection). This parameter is used to specify Informatica-specific error codes, separated by commas (,), that you want to ignore while running a job.
IGNORESUBREQ	Oracle Apps	N	Y or N. Setting this flag to Y stops huge job xml file transfers back and forth between the adapter and the AdapterHost during polls when a single request set has multiple sub-requests of more than 100. The default value is N or empty.
kerbkdc	MapReduce	<none>	If the Hadoop cluster is Kerberos secured, use this value to specify the KDC Server. For example, kerbkdc=172.25.6.112
kerbrealm	MapReduce	<none>	If the Hadoop cluster is Kerberos secured, use this value to specify the Kerberos Realm. For example, kerbrealm=TIDALSOFT.LOCAL
Keystore	BusinessObjects, BusinessObjects BI, BusinessObjects DS, Cognos, JD Edwards, Oracle Applications, UCS Manager, VMware, Web Service	<none>	Specify Keystore=c:\\<adapter_certificate_directory>\\<your_trusted_keystore>.keystore when importing certificates into a Java keystore.
LAUNCH_DELAY (in milliseconds)	Informatica	<none>	This parameter can be set in service.props, job configuration and connection configuration parameters. The order of precedence is service.props (applicable for all jobs running in all connections), job level (only for that particular job), and connection (applicable for all jobs in the connection). If a non-zero value is set for this parameter, then the jobs are delayed for the specified number of milliseconds before being submitted to Informatica.
LoginConfig	BusinessObjects BI Platform, BusinessObjects Data Services	<none>	Specifies the location of the login configuration if using WinAD or LDAP authentication. For example: LoginConfig=c:\\windows\\bscLogin.conf where "c:\\windows\\bscLogin.conf" is the location of the login configuration information. Note the use of \\ if this is a Windows location.

Table 1

Property	Applicable Adapter(s)	Default	What It Controls
MaxLogFiles	Informatica, JDBC, PeopleSoft	50	(Optional) - Number of logs to retain.
OUTPUT_ASYNC_LOGOUT	Informatica	N	Setting this flag to Y avoids jobs getting stuck in Gathering Output status.
OUTPUT_SYNC	All	Y	Enables concurrent output gathering on a connection. To enable this feature, set the value to N.
POLL_SYNC	All	Y	Enables concurrent polling on connections of the same type. This is helpful when there is a heavily load on one connection of an adapter. The heavily loaded connection will not affect the other adapter connection. To enable this feature, set the value to N.
QUERY_TIMEOUT	Oracle Apps	N	Y or N. If set to Y, the timeout value defined using the parameter QUERY_TIMEOUT_VALUE is applied to the SQL queries. Default value is N or empty.
QUERY_TIMEOUT_VALUE	Oracle Apps	unset	The time period in seconds that SQL queries wait before timeout. If 0 or not set, there is no timeout.
READPCHAINLOG	SAP	Y	Used to control the log gathering in SAP Process Chain jobs. This property depends on the Summary Only check box of the job definition Options tab.
SCANFOR_SESSIONSTATS	Informatica	Y	Y or N - Set this parameter to N to turn off the default behavior of Informatica jobs collecting the session statistics during the job run.
SCANFOR_SESSIONSTATS_AFTER_WF_ENDS	Informatica	N	Y or N - Set this parameter to Y to turn off the gathering of session statistics during each poll for the status of Informatica jobs.
TDLINFA_LOCALE	Informatica	<none>	Points to the Load Manager Library locale directory. See "Configuring the Informatica Adapter" in the <i>Informatica Adapter Guide</i> for how to set this for Windows and Unix environments.
TDLINFA_REQUESTTIMEOUT	Informatica	<none>	(Optional) - The number of seconds before an API request times out. The default is 120 seconds, if not specified.
TDLJDBC_LIBPATH	JDBC	<none>	(Windows only, optional) An alternate path to the JDBC library files. The library file path should have been configured given system environment variables. This option is available in case you wish to use an alternate set of libraries and may be helpful for trouble-shooting purposes.
TDLJDBC_LOCALE	JDBC	<none>	The path to the JDBC locale files.
TRANSACTION_LOG_BATCH_SIZE	MS SQL	5000	Set this parameter if more than 5000 lines need to be read from the transaction table.
version_pre898	JD Edwards	N	If running on a JD Edwards server version that is less than 8.9.8, set version_pre898=Y.

