



TRIFACTA

Databases Guide

Version: 9.2
Doc Build Date: 07/29/2022

Copyright © Trifacta Inc. 2022 - All Rights Reserved. CONFIDENTIAL

These materials (the “Documentation”) are the confidential and proprietary information of Trifacta Inc. and may not be reproduced, modified, or distributed without the prior written permission of Trifacta Inc.

EXCEPT AS OTHERWISE PROVIDED IN AN EXPRESS WRITTEN AGREEMENT, TRIFACTA INC. PROVIDES THIS DOCUMENTATION AS-IS AND WITHOUT WARRANTY AND TRIFACTA INC. DISCLAIMS ALL EXPRESS AND IMPLIED WARRANTIES TO THE EXTENT PERMITTED, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE AND UNDER NO CIRCUMSTANCES WILL TRIFACTA INC. BE LIABLE FOR ANY AMOUNT GREATER THAN ONE HUNDRED DOLLARS (\$100) BASED ON ANY USE OF THE DOCUMENTATION.

For third-party license information, please select **About Trifacta** from the Help menu.

1. <i>Install Databases</i>	4
1.1 <i>Install Database Client for PostgreSQL</i>	6
1.2 <i>Install Databases for PostgreSQL</i>	7
1.3 <i>Install Database Client for MySQL</i>	9
1.4 <i>Install Databases for MySQL</i>	10
1.5 <i>Configure the Databases</i>	16
1.6 <i>Create Databases and Users</i>	21
1.7 <i>Enable SSL for Databases</i>	22
1.8 <i>Upgrade Databases for PostgreSQL</i>	27
1.9 <i>Upgrade Databases for MySQL</i>	31
1.10 <i>Database Reference</i>	32
1.10.1 <i>Change Database Passwords for MySQL</i>	33
1.10.2 <i>Change Database Passwords for PostgreSQL</i>	35
1.10.3 <i>Change Database Port</i>	37
1.10.4 <i>Database Parameter Reference</i>	39
1.10.5 <i>Install Databases on Amazon RDS</i>	45
1.10.6 <i>Manual Database Install</i>	50

Install Databases

The Trifacta® platform uses multiple SQL databases to manage platform metadata. This section describes how to install and initialize these databases.

DB Installation Prerequisites

- You must install a supported database distribution. For more information on the supported database versions, see *System Requirements* in the Planning Guide.
 - You must also acquire the database dependencies associated with the operating system distribution where the database is to be installed. Please see the database vendor for more information.
- Please verify that the ports used by the database are opened on the Trifacta node.
 - For more information on default ports, see *System Ports* in the Planning Guide.
 - If you need to use different ports, additional configuration is required. More instructions are provided later.
- Installation and configuration of the database cannot be completed until the Trifacta software has been installed. You should install the software on the Trifacta node first.
- You must install the appropriate database client for your software distribution. Instructions are provided later.
- If you are installing the databases on an instance of Amazon RDS, additional setup is required first. See *Install Databases on Amazon RDS*.

Other prerequisites specific to the database distribution may be listed in the appropriate section below.

If you are concerned about durability and disaster recovery of your Trifacta metadata, your enterprise backup procedures should include the Trifacta databases. See *Backup and Recovery* in the Admin Guide.

List of Databases

The Trifacta® platform requires access to the following databases. Below, you can review database names, descriptions and release in which it was introduced:

Database Name	Description	First Release
Trifacta database (Main)	Storage of users and metadata about your datasets, including completed jobs.	Release 1.0
Jobs database	Storage of job tracking information. Jobs are purged upon completion or job timeout. Failed jobs are purged periodically.	Release 3.2
Scheduling database	Storage of schedules, including datasets to execute.	Release 4.1
Time-based Trigger database	Storage of triggering information.	Release 4.1
Configuration Service database	Storage of system-, tier-, and user-level configuration settings.	Release 6.0
Artifact Storage Service database	Storage for feature-specific usage data such value mappings.	Release 6.0
Job Metadata Service database	Storage of metadata on job execution.	Release 6.4
Authorization Service database	Storage of object permissions.	Release 7.1

Orchestration Service database	Storage of plan execution information. Information is purged upon completion or run timeout. Failed runs are purged periodically.	Release 7.1
Optimizer Service database	Storage of SQL queries to optimize logical and physical performance during job execution.	Release 7.6
Secure Token Service database	Storage of access tokens for managing integrations with external datasource such as OAuth2 connections.	Release 7.10
Connector Configuration Service database	Storage of metadata and overrides on connector types.	Release 8.1

Install Database Client for PostgreSQL

For install or upgrade operations on the Trifacta® databases, the client for your supported database distribution must be installed and available on the Trifacta node. Please complete the following steps for your database distribution.

NOTE: The following commands install the latest database client of the listed version. If you need a specific version, please download from the software provider.

NOTE: The installing user must have sudo privileges.

Install database client for PostgreSQL 12

Setup - CentOS/RHEL all versions:

```
curl -s https://packagecloud.io/install/repositories/trifacta/dependencies/script.rpm.sh | sudo bash
```

Install - CentOS/RHEL 8:

```
sudo yum install postgresql-client-12
```

Install - CentOS/RHEL 7:

```
sudo yum install postgresql-client-12
```

Setup - Ubuntu all versions:

```
curl -s https://packagecloud.io/install/repositories/trifacta/dependencies/script.deb.sh | sudo bash
```

Install - Ubuntu 18.04 (Bionic):

```
sudo apt-get install postgresql-client-12
```

Install - Ubuntu 16.04 (Xenial):

```
sudo apt-get install postgresql-client-12
```

Install Databases for PostgreSQL

Contents:

- *Limitations*
 - *Prerequisites*
 - *Select Configuration File*
 - *Database Install*
 - *PostgreSQL 12*
 - *Acquire Port Information*
 - *PostgreSQL 12*
 - *SSL*
-

This section describes how to install the PostgreSQL database server, after which you can create and initialize the databases and their users.

Limitations

- You must install a supported version of the database.

NOTE: Installation of the databases into any database schema other than `PUBLIC` is not supported.

- For more information on supported versions of this database type, see *System Requirements* in the Planning Guide.

Prerequisites

- The installing user must have write permissions to the directory from which the commands are executed.
- The installing user must have sudo privileges.
- The database client should already be installed. See *Install Database Client for PostgreSQL*.

Select Configuration File

By default, the Trifacta platform assumes that you are installing the databases in a PostgreSQL instance, which is reflected in the default platform configuration file. This configuration file is stored here:

```
/opt/trifacta/conf/trifacta-conf.json
```

No modifications to this file are required at this time.

Database Install

PostgreSQL 12

O/S Distribution	URL	Package Name
CentOS/RHEL 7	https://download.postgresql.org/pub/repos/yum/repopms/EL-7-x86_64/pgdg-redhat-repo-latest.noarch.rpm	postgresql12-server
CentOS/RHEL 8	https://download.postgresql.org/pub/repos/yum/repopms/EL-8-x86_64/pgdg-redhat-repo-latest.noarch.rpm	postgresql12-server

For CentOS/RHEL 7.x:

```
wget https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-x86_64/pgdg-redhat-repo-latest.noarch.rpm
sudo yum -y install pgdg-redhat-repo-latest.noarch.rpm
sudo yum -y install postgresql12-server
```

For CentOS/RHEL 8.x:

```
wget https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-x86_64/pgdg-redhat-repo-latest.noarch.rpm
sudo yum -y install pgdg-redhat-repo-latest.noarch.rpm
sudo yum -y install postgresql12-server
```

Acquire Port Information

After you have completed the installation, you must acquire the port information for each database from the following locations on the Trifacta node. These port numbers need to be applied inside the Trifacta platform.

NOTE: By default, PostgreSQL and the platform use port 5432 for communication. If that port is not available at install/upgrade time, the next available port is used, which is typically 5433. This change may occur if a previous version of PostgreSQL is on the same server. When a non-default port number is used, the platform must be configured to use it. For more information, see *Change Database Port*.

PostgreSQL 12

CentOS/RHEL:

```
/var/lib/pgsql/12/data/postgresql.conf
```

Ubuntu:

```
/etc/postgresql/12/main/postgresql.conf
```

SSL

Support for SSL requires additional configuration. For more information, see *Enable SSL for Databases*.

Install Database Client for MySQL

For install or upgrade operations on the Trifacta® databases, the client for your supported database distribution must be installed and available on the Trifacta node. The database client for MySQL can be installed from your MySQL distribution. For more information, please see the documentation provided with the distribution.

Install Databases for MySQL

Contents:

- *Limitations*
 - *Prerequisites*
 - *Acquire MySQL Java driver*
 - *MySQL log_bin_trust_function_creators is required for installation*
 - *Select Configuration File*
 - *Database Install*
 - *Acquire Port Information*
 - *Configure MySQL Timezone*
 - *Enable SQL Mode*
 - *Update MySQL Password Policy*
-

This section describes how to install the MySQL database server, after which you can create and initialize the databases and their users.

Limitations

NOTE: MySQL 5.7 Community is not supported on CentOS/RHEL 8.x.

- You must install a supported version of the database. For more information on supported versions of this database type, see *System Requirements* in the Planning Guide.

Prerequisites

- The installing user must have write permissions to the directory from which the commands are executed.
- The installing user must have sudo privileges.

NOTE: The database client should already be installed. See *Install Database Client for MySQL*.

Acquire MySQL Java driver

The MySQL Java driver is not packaged with the Trifacta installer. If you are installing the Trifacta databases into MySQL, please acquire the following driver files.

```
mysql-connector-java-8.0.20.jar
```

This file can be downloaded from the following locations:

- <https://dev.mysql.com/downloads/connector/j/>
- <https://mvnrepository.com/artifact/mysql/mysql-connector-java/8.0.20>
- This file needs to be installed in the following locations on the Trifacta node:

```

/opt/trifacta/services/artifact-storage-service/build/install/artifact-storage-service/lib/mysql-connector-
java-8.0.20.jar
/opt/trifacta/services/authorization-service/build/install/authorization-service/lib/mysql-connector-java-
8.0.20.jar
/opt/trifacta/services/configuration-service/build/install/configuration-service/lib/mysql-connector-java-
8.0.20.jar
/opt/trifacta/services/batch-job-runner/build/install/batch-job-runner/lib/mysql-connector-java-8.0.20.jar
/opt/trifacta/services/job-metadata-service/build/install/job-metadata-service/lib/mysql-connector-java-
8.0.20.jar
/opt/trifacta/services/orchestration-service/build/install/orchestration-service/lib/mysql-connector-java-
8.0.20.jar
/opt/trifacta/services/scheduling-service/server/build/install/scheduling-service/lib/mysql-connector-java-
8.0.20.jar
/opt/trifacta/services/time-based-trigger-service/server/build/install/time-based-trigger-service/lib/mysql-
connector-java-8.0.20.jar
/opt/trifacta/services/optimizer-service/build/install/optimizer-service/lib/mysql-connector-java-8.0.20.jar
/opt/trifacta/services/secure-token-service/server/build/install/secure-token-service/lib/mysql-connector-
java-8.0.20.jar
/opt/trifacta/services/secure-token-service/server/build/install/connector-configuration-service/lib/mysql-
connector-java-8.0.20.jar

```

MySQL log_bin_trust_function_creators is required for installation

When the Trifacta databases are hosted on MySQL, for the installation process, you must enable the `log_bin_trust_function_creators` flag. When enabled, this flag allows MySQL to trust the creators of stored functions to not write unsafe events to the binary log.

NOTE: This flag is a global flag in MySQL. It is only required during the installation process and can be disabled afterward.

For more information, see
https://dev.mysql.com/doc/refman/5.7/en/replication-options-binary-log.html#sysvar_log_bin_trust_function_creators

Select Configuration File

If you are installing the database in a MySQL instance, a separate base configuration has been provided. This configuration file is stored in the following location:

```

/opt/trifacta/conf/trifacta-conf.json.MYSQL_DB

```

To use MySQL, you should back up the default configuration file and then copy the MySQL version in its place:

If you have already applied configuration changes to `trifacta-conf.json` through the above file or the Admin Settings page, these changes are lost when the following steps are performed. You must manually migrate those changes over or apply the MySQL changes manually.

```

cp /opt/trifacta/conf/trifacta-conf.json /opt/trifacta/conf/trifacta-conf.json.POSTGRES_DB
cp /opt/trifacta/conf/trifacta-conf.json.MYSQL_DB /opt/trifacta/conf/trifacta-conf.json

```

Database Install

NOTE: The following distributions and commands are for MySQL Community Server 5.7.

O/S Distribution	URL	Package Name
CentOS 7	https://dev.mysql.com/get/mysql80-community-release-el7-1.noarch.rpm	mysql-community-server
CentOS 8	https://dev.mysql.com/get/mysql80-community-release-el8-1.noarch.rpm <div> NOTE: MySQL 5.7 Community is not supported on CentOS/RHEL 8.x. </div>	mysql-community-server
RHEL 7	https://dev.mysql.com/get/mysql80-community-release-el7-1.noarch.rpm	mysql-community-server
RHEL 8	https://dev.mysql.com/get/mysql80-community-release-el8-1.noarch.rpm <div> NOTE: MySQL 5.7 Community is not supported on CentOS/RHEL 8.x. </div>	mysql-community-server

O/S Distribution	URL	Package Name
Ubuntu 16.04	https://dev.mysql.com/get/mysql-apt-config_0.8.10-1_all.deb	
Ubuntu 18.04	https://dev.mysql.com/get/mysql-apt-config_0.8.10-1_all.deb	

For CentOS 7.x:

```
# Install MySQL Repo List
sudo wget https://dev.mysql.com/get/mysql80-community-release-el7-1.noarch.rpm
sudo rpm -Uvh mysql80-community-release-el7-1.noarch.rpm
# Check list of available MySQL Repos; by default 8.0 is enabled, but we want 5.7
yum repolist all | grep mysql
# Disable 8.0 and enable 5.7
sudo yum-config-manager --disable mysql80-community
sudo yum-config-manager --enable mysql57-community
# Verify repo state
yum repolist all | grep mysql
# Install MySQL Server
sudo yum install mysql-community-server
# Start MySQL server
sudo systemctl start mysqld.service
# Verify status
sudo systemctl status mysqld.service
```

For CentOS 8.x:

NOTE: MySQL 5.7 Community is not supported on CentOS/RHEL 8.x.

```
# Install MySql Repo List
sudo wget https://dev.mysql.com/get/mysql80-community-release-el8-1.noarch.rpm
sudo rpm -Uvh mysql80-community-release-el7-1.noarch.rpm
# Check list of available MySQL Repos; by default 8.0 is enabled, but we want 5.7
yum repolist all | grep mysql
# Disable 8.0 and enable 5.7
sudo yum-config-manager --disable mysql80-community
sudo yum-config-manager --enable mysql57-community
# Verify repo state
yum repolist all | grep mysql
# Install Mysql Server
sudo yum install mysql-community-server
# Start mysql server
sudo systemctl start mysqld.service
# Verify status
sudo systemctl status mysqld.service
```

For Red Hat Enterprise Linux 7.x: See CentOS 7 above.

For Red Hat Enterprise Linux 8.x: See CentOS 8 above.

For Ubuntu 16.04:

```
# Install and configure repo config package
sudo apt-get update && sudo apt-get install lsb-release
wget https://dev.mysql.com/get/mysql-apt-config_0.8.10-1_all.deb
sudo debconf-set-selections <<< "mysql-apt-config mysql-apt-config/select-server select mysql-5.7"
sudo DEBIAN_FRONTEND=noninteractive dpkg -i mysql-apt-config_0.8.10-1_all.deb
sudo apt-get update
# Set Installer configs Admin password and install MySQL Server package
sudo debconf-set-selections <<< "mysql-community-server mysql-community-server/root-pass password
<MYSQL_ADMIN_PASSWORD>"
sudo debconf-set-selections <<< "mysql-community-server mysql-community-server/re-root-pass password
<MYSQL_ADMIN_PASSWORD>"
sudo DEBIAN_FRONTEND=noninteractive apt-get install mysql-community-server
sudo service mysql start
```

For Ubuntu 18.04:

```
# Install and configure repo config package
sudo apt-get update && sudo apt-get install lsb-release
wget https://dev.mysql.com/get/mysql-apt-config_0.8.10-1_all.deb
sudo debconf-set-selections <<< "mysql-apt-config mysql-apt-config/select-server select mysql-5.7"
sudo DEBIAN_FRONTEND=noninteractive dpkg -i mysql-apt-config_0.8.10-1_all.deb
sudo apt-get update
# Set Installer configs Admin password and install MySQL Server package
sudo debconf-set-selections <<< "mysql-community-server mysql-community-server/root-pass password
<MYSQL_ADMIN_PASSWORD>"
sudo debconf-set-selections <<< "mysql-community-server mysql-community-server/re-root-pass password
<MYSQL_ADMIN_PASSWORD>"
sudo DEBIAN_FRONTEND=noninteractive apt-get install mysql-community-server
sudo service mysql start
```

Acquire Port Information

After you have completed the installation, you must acquire the port information for each database from the following locations on the Trifacta node. These port numbers need to be applied inside the Trifacta platform.

CentOS/RHEL (MySQL 5.7):

The default port is 3306.

Ubuntu (MySQL 5.7) : Not supported.

Configure MySQL Timezone

If your MySQL databases are in a different timezone from the Trifacta node, you must configure the timezone value for each database, so that it can be inserted as part of the connection string.

NOTE: If the Trifacta node and your MySQL databases are co-located in the same timezone, you can skip this section.

In the Admin Settings page, these parameters are in the following form:

```
"*.database.mysqlServerTimezone": "",
```

For each database, insert the appropriate timezone value. For more information on supported values, see the documentation for your MySQL product.

See *Database Parameter Reference*.

Enable SQL Mode

When using MySQL, you must enable the following modes:

NOTE: These modes are enabled by default in deployments of upstream MySQL. They must be enabled manually if you are installing the databases in a hosted environment, such as Amazon RDS or Azure MySQL. Refer to your MySQL host's documentation.

- STRICT_TRANS_TABLES
- NO_ENGINE_SUBSTITUTION

To verify:

To verify if these modes are enabled, please execute the following query:

```
SELECT @@sql_mode;
```

The two required values should be listed in the query output.

To enable:

For more information on enabling these modes, see <https://dev.mysql.com/doc/refman/5.7/en/sql-mode.html>.

Update MySQL Password Policy

NOTE: This section only applies to CentOS and RHEL platforms only where MySQL is the installed database.

By default, MySQL enforces a stricter password policy on database passwords. If you prefer to set your own passwords outside of this policy, you must lower the password policy. Please complete the following steps:

```
# Get temporary root password from mysql log
sudo grep 'temporary password' /var/log/mysqld.log
# Connect to server as root
mysql -uroot -p
# Update password
ALTER USER 'root'@'localhost' IDENTIFIED BY '<my_new_password>';
# Unless you plan to update all the User passwords to be meet MySql Security requirements, you should set the
password policy to low
SET GLOBAL validate_password_policy=LOW;
```

Configure the Databases

Contents:

- *Initialize*
 - *PostgreSQL 12*
 - *MySQL*
 - *Set custom database parameters*
 - *Next Steps*
-

Initialize

Use the following steps to initialize the databases of the Trifacta® platform.

NOTE: These steps assume that the Trifacta node is the host of these databases. Please modify the following steps if you are connecting to databases on other nodes.

Prerequisites:

- The initializing user must have write permissions to the directory from which the commands are executed.
- The initializing user must have sudo privileges.

PostgreSQL 12

For CentOS/RHEL 7.x and 8.x:

```
sudo /usr/pgsql-12/bin/postgresql12-setup initdb
```

For Ubuntu 16.04 and 18.04:

```
pg_createcluster -d /var/lib/postgresql/12/main 12 main
```

MySQL

No additional steps are required to initialize the databases in MySQL.

Set custom database parameters

Use the following steps to set custom database names, usernames, and passwords in the Trifacta platform:

1. Edit `/opt/trifacta/conf/trifacta-conf.json`.
2. For each database, you can review the parameters in the listed area and make modifications as needed.

NOTE: For each database, you should change the default password. This change must also be applied on the database server. See *Change Database Passwords for PostgreSQL* . See *Change Database Passwords for MySQL*.

NOTE: The `type` is set to `POSTGRESQL` by default. Modify the value if you are installing the databases into a different database server.

Database	Parameter area
Main database	webapp.database.*
Jobs database	batch-job-runner.database.*
Scheduling database	scheduling-service.database.*
Time-Based Trigger database	time-based-trigger-service.database.*
Configuration Service database	configuration-service.database.*
Job Metadata Service database	job-metadata-service.database.*
Artifact Storage Service database	artifact-storage-service.database.*
Authorization Service database	authorization-service.database.*
Orchestration Service database	orchestration-service.database.*
Optimizer Service database	optimizer-service.database.*
Secure Token Service database	secure-token-service.database.*
Connector Configuration Service database	connector-configuration-service.database.*

For more information, see *Database Parameter Reference*.

3. Make changes in the file as needed and save.

Apply customizations on upgrade

If you have customized database properties, you must apply the edits from the new sample file to the existing configuration file after you have upgrade the Trifacta platform.

If you are using all defaults, you can just overwrite the existing file with the new version's sample file.

PostgreSQL:

1. Locate the sample Postgres configuration file:

```
/opt/trifacta/bin/setup-utils/db/pg_hba.conf.SAMPLE
```

2. If you are upgrading and have customizations in your existing version, you must apply the edits in the above to the following file. Otherwise, overwrite the following file with the above one based on your operating system:
 - a. CentOS/RHEL dir: /var/lib/pgsql/9.6/data/pg_hba.conf
 - b. Ubuntu dir: /etc/postgresql/9.6/main/pg_hba.conf
3. From the SAMPLE file, copy the following declarations and paste them into the production pg_hba.conf file **above any other declarations**:

NOTE: You can substitute different database usernames and groups for the ones listed below (trifacta and trifacta). These values may be needed for other configuration.

- a. Trifacta database:

local	trifacta	trifacta		md5
host	trifacta	trifacta	127.0.0.1/32	md5
host	trifacta	trifacta	::1/128	md5

b. Jobs database:

--	--	--	--	--

c. Scheduling database:

local	trifactaschedulingservice			
trifactaschedulingservice			md5	
host	trifactaschedulingservice	trifactaschedulingservice	127.0.0.1	
/32	md5			
host	trifactaschedulingservice	trifactaschedulingservice	::1	
/128	md5			

d. Time-based Trigger database:

local	trifactatimebasedtriggerservice			
trifactatimebasedtriggerservice			md5	
host	trifactatimebasedtriggerservice	trifactatimebasedtriggerservice		
127.0.0.1/32	md5			
host	trifactatimebasedtriggerservice	trifactatimebasedtriggerservice	::1	
/128	md5			

e. Configuration Service database:

local	trifactaconfigurationservice			
trifactaconfigurationservice			md5	
host	trifactaconfigurationservice	trifactaconfigurationservice	127.0.0.1	
/32	md5			
host	trifactaconfigurationservice	trifactaconfigurationservice	::1	
/128	md5			

f. Artifact Storage Service database:

local	trifactaartifactstorageservice			
trifactaartifactstorageservice			md5	
host	trifactaartifactstorageservice	trifactaartifactstorageservice	127.0.0.1	
/32	md5			
host	trifactaartifactstorageservice	trifactaartifactstorageservice	::1	
/128	md5			

g. Job Metadata Service database:

local	trifactajobmetadataservice			
trifactajobmetadataservice			md5	
host	trifactajobmetadataservice	trifactajobmetadataservice	127.0.0.1	
/32	md5			
host	trifactajobmetadataservice	trifactajobmetadataservice	::1	
/128	md5			

h. Authorization Service database:

```

local    trifactaauthorizationsservice
trifactaauthorizationsservice                md5
host     trifactaauthorizationsservice        trifactaauthorizationsservice    127.0.0.1
/32      md5
host     trifactaauthorizationsservice        trifactaauthorizationsservice    ::1
/128     md5

```

i. Orchestration Service database:

j. Optimizer Service database:

```

trifacloptimizerservice
trifacloptimizerservice    127.0.0.1/32
trifacloptimizerservice    ::1/128

```

k. Secure Token Service database:

```

local    trifactasecuretokenservice
trifactasecuretokenservice                md5
host     trifactasecuretokenservice        trifactasecuretokenservice    127.0.0.1
/32      md5
host     trifactasecuretokenservice        trifactasecuretokenservice    ::1
/128     md5

```

l. Connector Configuration Service database:

NOTE: The default values for this database do not follow the naming conventions for other databases.

```

local    trifactaconnectorconfigservice
trifactaconnectorconfigservice                md5
host     trifactaconnectorconfigservice        trifactaconnectorconfigservice    127.0.0.1
/32      md5
host     trifactaconnectorconfigservice        trifactaconnectorconfigservice    ::1
/128     md5

```

m. Save the file.

4. Restart the databases:

- a. If you have also restarted the operating system, please execute the following first, followed by the O/S-specific commands:

NOTE: This command is valid only if the Postgres DB is also hosted in the Trifacta node.

For PostgreSQL 12 on CentOS/RHEL 7:

```
chkconfig postgresql-12 on
```

b. CentOS/RHEL 7 (PostgreSQL 12):

```
sudo service postgresql-12 start
```

c. Ubuntu:

```
sudo service postgresql start
```

MySQL:

Upgrading MySQL versions is not supported in this release.

Next Steps

1. If the configuration files indicate that the databases are listening on a port other than the default, this port number must be applied within the Trifacta platform configuration. For more information, see *Change Database Port*.
2. If you are using non-default usernames and passwords, they must be applied within the Trifacta platform configuration. For more information, see *Change Database Passwords for PostgreSQL*.
3. When you have completed the above configuration, you can create the databases and their roles (users) and perform additional configuration. See *Create Databases and Users*.
4. You can enable use of TLS secure access for connections between the Trifacta application and the Trifacta databases. For more information, see *Enable SSL for Databases*.

Create Databases and Users

Contents:

- *Create DBs and Users*
 - *PostgreSQL*
 - *MySQL*
 - *Backup*
 - *Configure Non-Default Database Connections*
-

Create DBs and Users

PostgreSQL

Run the following script, which builds the four databases and specifies the appropriate roles (users) for each database, based on the parameters you have specified in `trifacta-conf.json` and in the `pg_hba.conf`:

NOTE: This script must be run as the root user or via sudo superuser.

```
/opt/trifacta/bin/setup-utils/db/trifacta-create-postgres-roles-dbs.sh
```

MySQL

Run the following script, which builds the four databases and specifies the appropriate roles (users) for each database, based on the parameters you have specified in `trifacta-conf.json`:

NOTE: This script must be run as the root user or via sudo superuser.

```
/opt/trifacta/bin/setup-utils/db/trifacta-create-mysql-users-dbs.sh -u <MySQL_admin_username> -w  
<MySQL_admin_password>
```

where:

- `-u` = MySQL admin username, which is usually `root`.
- `-w` = MySQL admin password

Backup

For more information on backup recommendations and commands, see *Backup and Recovery* in the Admin Guide.

Configure Non-Default Database Connections

If you have used non-default values for the username, password, host, or port value for either database, you must update platform configuration. For more information, see *Database Parameter Reference*.

Enable SSL for Databases

Contents:

- *Install SSL Certificate*
 - *Enable*
 - *Configure for Certificate*
 - *Configure Databases for SSL*
 - *Configuration Example - Minimal SSL configuration*
 - *Configuration Example - SSL with Client Authentication*
 - *Configuration Example - SSL with a custom certificate*
 - *Use*
-

Optionally, you can enable Transport Layer Security (TLS), commonly known as SSL, access between the Trifacta® application, its services, and the Trifacta databases.

Tip: SSL can be applied to any supported database distribution.

NOTE: This configuration applies only to the databases that are used to store metadata for the Trifacta platform. For more information on enabling SSL for external JDBC connections, see *Configure Security for Relational Connections*.

Install SSL Certificate

Before you enable SSL for the Trifacta databases, you must deploy a security certificate on the Trifacta node. The certificate must be installed on the Trifacta node, whether the databases are installed locally or remotely.

NOTE: Please retain the location of the certificate on the server, as well as other information listed in the sections below.

NOTE: If you receive a `org.postgresql.util.PSQLException: Could not read SSL key file` error message when connecting via SSL to your PostgreSQL databases, you may need to convert your certificate to DER format and re-install. For more information, see <https://www.enterprisedb.com/postgres-tutorials/how-enable-ssl-authentication-edb-postgres-advanced-server>.

- If SSL is in use for access to the Trifacta application, you can use the same SSL certificate for the databases. For more information, see *Install SSL Certificate*.
- You can also use a separate certificate for the databases, if desired.

Enable

To enable use of SSL to connect to the platform databases, please complete the following.

Steps:

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

2. Locate the following setting, and set it to `true`:

3. `"webapp.database.ssl.enabled": true,`

4. Do not save your changes yet.

Configure for Certificate

After the SSL certificate has been deployed to the server, please complete the following steps to configure use of the certificate by the Trifacta application.

Steps:

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Locate the following settings and set them accordingly:

Setting	Description
<code>rejectUnauthorized</code>	(optional) Set this value to <code>true</code> to reject access by any client that is presenting an invalid server certificate.
<code>serverCertificateAuthorityFile</code>	(optional) Path on the Trifacta node to the certificate authority verification file, which is used to verify the presented server certificate.
<code>clientKeyFile</code>	(optional) Path on the Trifacta node to the client key file, which is used for client authentication.
<code>clientCertificateFile</code>	(optional) Path on the Trifacta node to the SSL certificate to use for client authentication.

3. Save your changes and restart the platform.

Configure Databases for SSL

After you have enabled the use of SSL in the platform, you must configure each Trifacta database to use secure access.

Steps:

To enable SSL on individual databases, you must apply the appropriate configuration settings as `additionalConnectionProperties` for the database.

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.

Tip: Although you can apply these changes through `trifacta-conf.json`, it may be easier to apply through the Admin Settings page in the Trifacta application if it is available.

2. Search for the following string:

```
database.additionalConnectionProperties
```

- For each of the above settings, you must add the following text string(s) containing key-value pairs to the `additionalConnectionProperties`, based on your database distribution, for each listed service database:

NOTE: Key-value pairs must be separated by an ampersand (&). See Configuration Examples below.

Trifacta platform setting and value	PostgreSQL	MySQL
"webapp.database.ssl.enabled": true,	ssl=true	requireSSL=true
"webapp.database.ssl.rejectUnauthorized": true,	sslmode=require	verifyServerCertificate=true
"webapp.database.ssl.serverCertificateAuthorityFile": "/path/to/caFile",	sslrootcert=/path/to/caFile	trustCertificateKeyStoreUrl=file:/path/to/truststore&trustCertificateKeyStorePassword=<password>
"webapp.database.ssl.clientKeyFile": "/path/to/keyFile",	sslkey=/path/to/keyFile	clientCertificateKeyStoreUrl=file:/path/to/truststore&clientCertificateKeyStorePassword=<password>
"webapp.database.ssl.clientCertificateFile": "/path/to/certFile",	sslcert=/path/to/certFile	

- Apply the values based on your configuration example below.

Configuration Example - Minimal SSL configuration

For minimal SSL configuration, the configuration that you performed above look like the following:

PostgreSQL:

```
"<service>.database.additionalConnectionProperties": "ssl=true&sslmode=require",
```

MySQL:

```
"<service>.database.additionalConnectionProperties": "requireSSL=true&verifyServerCertificate=true",
```

Configuration Example - SSL with Client Authentication

If you have deployed a client key and certificate for authentication, your configuration may look like the following:

PostgreSQL:

```
"<service>.database.additionalConnectionProperties": "ssl=true&sslmode=require&sslkey=/path/to/keyFile&sslcert=/path/to/certFile",
```

MySQL:

```
"<service>.database.additionalConnectionProperties":  
"requireSSL=true&verifyServerCertificate=true&clientCertificateKeyStoreUrl=file:/path/to/truststore&clientCertificateKeyStorePassword=<password>",
```

Configuration Example - SSL with a custom certificate

If you have deployed a custom SSL certificate on the Trifacta node, your configuration may look like the following. For more information, see *Install SSL Certificate*.

PostgreSQL:

```
"<service>.database.additionalConnectionProperties": "ssl=true&sslmode=require&sslrootcert=/path/to/caFile",
```

MySQL:

```
"<service>.database.additionalConnectionProperties":  
"requireSSL=true&verifyServerCertificate=true&trustCertificateKeyStoreUrl=file:/path/to/truststore&trustCertificateKeyStorePassword=<password>",
```

Use

When SSL is enabled and configured, users of the Trifacta platform automatically connect to the database using SSL.

NOTE: There may be a small performance cost to using SSL.

Upgrade Databases for PostgreSQL

Contents:

- *Prerequisites*
 - *Backup*
 - *Acquire Distribution and Port Information*
 - *Database Prep*
 - *Before you begin*
 - *Using non-default ports*
 - *Upgrade PostgreSQL 9.6 to 12*
 - *PostgreSQL path references*
 - *Upgrade for CentOS/RHEL - PostgreSQL 9.6 to 12*
 - *Upgrade for Ubuntu - PostgreSQL 9.6 to 12*
 - *Upgrade on Amazon RDS - PostgreSQL 9.6 to 12*
 - *Transfer Settings*
 - *Verify and Cleanup*
-

This section describes the requirements and pre- and post-upgrade steps for upgrading the PostgreSQL database server of the Trifacta® databases for a working deployment of the Trifacta platform.

When you upgrade to a new release of the platform, the underlying databases and their structures are automatically migrated to the new format. However, the database releases are not touched.

Prerequisites

NOTE: Responsibility for performing the actual upgrade of the database server is the responsibility of the customer. Please review this section before you perform the database server upgrade.

- The installing user must have write permissions to the directory from which the commands are executed.
- The installing user must have sudo privileges.
- Verify that you know the host and port number for each database.

Backup

Before you begin, please verify that you have a valid backup of each Trifacta database. See *Backup and Recovery* in the Admin Guide.

Acquire Distribution and Port Information

Before you begin, you must:

- Acquire the port information for the current database. Typically, this value is 5432.

NOTE: By default, PostgreSQL and the platform use port 5432 for communication. If that port is not available at install/upgrade time, the next available port is used, which is typically 5433. This change may occur if a previous version of PostgreSQL is on the same server. When a non-default port number is used, the platform must be configured to use it. For more information, see *Change Database Port*.

- Acquire the latest distribution for the database software.

For more information, see *Install Databases for PostgreSQL*.

For more information, see *Install Databases for MySQL*.

Database Prep

This procedure describes the process for upgrading the PostgreSQL version in use by the Trifacta databases. This procedure assumes the following:

- All current databases are co-located with the software on the Trifacta node.
- Some downtime of the databases during the upgrade process is ok.

Before you begin

- If you haven't already, please back up each database and all PostgreSQL configuration files. See *Backup and Recovery* in the Admin Guide.
- Login to the node where the databases are hosted.

Using non-default ports

The default port for these commands is port 5432.

If you are using a non-standard port for either the pre- or post-upgrade versions of PostgreSQL, you can use the following parameters with the `pg_upgrade` command:

Parameter	Description
<code>--old-port=<pre-upgrade_port></code>	The port number of the pre-upgrade instance of PostgreSQL
<code>--new-port=<post-upgrade_port></code>	The port number to use for the post-upgrade instance of PostgreSQL

For more information, see <https://www.postgresql.org/docs/12/pgupgrade.html>.

Upgrade PostgreSQL 9.6 to 12

Support for PostgreSQL 9.6 is deprecated. You must upgrade to PostgreSQL 12.

The following sections cover upgrading the versions listed below:

- Source version: PostgreSQL 9.6
- Upgrade version: PostgreSQL 12

NOTE: The database version upgrade must be performed after you have upgraded the software.

NOTE: The database client is installed as part of this upgrade. No further action is required to enable use of the database client.

PostgreSQL path references

In PostgreSQL 10 and later, directory references to versions of PostgreSQL paths have changed.

PostgreSQL 9.6 example path:

```
/var/lib/pgsql/9.6/data/pg_hba.conf
```

PostgreSQL 12 example path:

```
/var/lib/pgsql/12/data/pg_hba.conf
```

For more information, see https://wiki.postgresql.org/wiki/YUM_Installation.

Upgrade for CentOS/RHEL - PostgreSQL 9.6 to 12

For more information, see <https://www.postgresql.org/docs/>.

Upgrade for Ubuntu - PostgreSQL 9.6 to 12

For more information, see <https://www.postgresql.org/docs/>.

Upgrade on Amazon RDS - PostgreSQL 9.6 to 12

NOTE: After upgrading to PostgreSQL on Amazon RDS, you may encounter an I/O error when importing data in which the storage location has the wrong hash version. The solution is to reindex tables. For more information, see <https://aws.amazon.com/blogs/database/postgresql-12-a-deep-dive-into-some-new-functionality/>.

For more information, see https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_UpgradeDBInstance.PostgreSQL.html.

Transfer Settings

For all operating systems, you must transfer the settings from your old version of PostgreSQL to the new one.

1. Update the port number and any other settings in the platform configuration. To apply this configuration change, login as an administrator to the Trifacta node. Then, edit `trifacta-conf.json`. Some of these settings may not be available through the *Admin Settings Page*. For more information, see *Platform Configuration Methods*.
 - a. Replace all instances of the PostgreSQL port number for the earlier version and replace with the desired port number for the upgrade version.

NOTE: By default, PostgreSQL and the platform use port 5432 for communication. If that port is not available at install/upgrade time, the next available port is used, which is typically 5433. This change may occur if a previous version of PostgreSQL is on the same server. When a non-default port number is used, the platform must be configured to use it. For more information, see *Change Database Port*.

- b. Make the above changes and save the file.
2. Transfer settings from the configuration files for your old database version to the new one. Please review the old and new versions of these files:

NOTE: It is risky to perform a straight copy of these configuration files. Settings may change. New ones may be introduced. Setting values specific to the installation may be overwritten in a copy. Please retain a backup of both versions of each file before migrating settings.

Path to PostgreSQL 9.6 file	Path to PostgreSQL 12 file
/var/lib/pgsql/9.6/data/pg_hba.conf	/var/lib/pgsql/12/data/pg_hba.conf
/var/lib/pgsql/9.6/data/postgresql.conf	/var/lib/pgsql/12/data/postgresql.conf

3. Start the service:

a. CentOS/RHEL:

```
sudo service postgresql-9.6 start
```

b. Ubuntu:

```
sudo service postgresql start
```

c. When the service restarts, you can check the cluster status using the following script:

```
./analyze_new_cluster.sh
```

4. After you have completed the database installation, you must review the port number of the newly installed database, which may have changed between versions of the database software. That new port number must be applied through the Trifacta software. For more information, see *Change Database Port*.
5. If all is well, restart the platform. See *Start and Stop the Platform* in the Install Guide.

Verify and Cleanup

1. Verify operations on all databases:
 - a. Login to the application.
 - b. Load a dataset from Flow View.
 - c. Run a job.
 - d. Schedule a job and execute it.
 - e. See *Verify Operations* in the Admin Guide.
2. If all of the above tests pass, you can use the following script to delete the old PostgreSQL version and its data directory:

```
./delete_old_cluster.sh
```

3. Restart the Trifacta platform. See *Start and Stop the Platform* in the Install Guide.

Upgrade Databases for MySQL

Upgrading the databases for MySQL is not supported in this release. Please start with a clean install of the databases on MySQL.

Database Reference

After installation, the following database topics may apply to your environment.

Change Database Passwords for MySQL

You should change the passwords for the Trifacta® databases immediately after installing the software.

Please complete the following steps to change the passwords for the Trifacta platform users of the Trifacta databases.

Change in Database Server

1. If you have not done so already, switch to the MySQL user and launch:

```
sudo su - mysql
mysql
```

2. For each database, change the default password. Default usernames are listed for each database. Please modify the command if you are using different roles and insert a more secure password for <new_pwd>:

NOTE: In MySQL, the dash is a forbidden character in usernames. Please use underscores instead.

```
ALTER USER 'trifacta'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactaactiviti'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactaschedulingservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactatimebasedtriggerservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactaconfigurationsservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactasartifactstorageservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactajobmetadataservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactaauthorizationservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactaorchestrationservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactaoptimizerservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactasecuretokenservice'@'localhost' IDENTIFIED BY '<new_pwd>';
ALTER USER 'trifactaconnectorconfigservice'@'localhost' IDENTIFIED BY '<new_pwd>';
```

3. Exit:

```
\q
exit
```

Change in Trifacta configuration

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Modify the user and password properties for any of the following databases:

Database Name	Parameter area
Trifacta database	webapp.database.*
Jobs database	batch-job-runner.database.*
Scheduling database	scheduling-service.database.*

Time-based Trigger database	time-based-trigger-service.database.*
Configuration Service database	configuration-service.database.*
Artifact Storage Service database	artifact-storage-service.database.*
Job Metadata Service database	job-metadata-service.database.*
Authorization Service database	authorization-service.database.*
Orchestration Service database	orchestration-service.database.*
Optimizer Service database	optimizer-service.database.*
Secure Token Service database	secure-token-service.database.*
Connector Configuration Service database	connector-configuration-service.database.*

For more information, see *Database Parameter Reference*.

3. Save your changes and restart the platform.

Change Database Passwords for PostgreSQL

You should change the passwords for the Trifacta® databases immediately after installing the software.

Please complete the following steps to change the passwords for the Trifacta platform users of the Trifacta databases.

Change in Database Server

1. If you have not done so already, switch to the Postgres user and launch psql:

```
sudo su - postgres
psql
```

2. For each database, change the default password. Default usernames are listed for each database. Please modify the command if you are using different roles and insert a more secure password for <new_pwd>:

```
ALTER ROLE trifacta WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactaactiviti WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactatimebasedtriggerservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactaschedulingservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactaconfigurationsservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactaartifactstorageservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactajobmetadataservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactaauthorizationservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactaorchestrationservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactaoptimizerservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactasecuretokenservice WITH PASSWORD '<new_pwd>';
ALTER ROLE trifactaconnectorconfigservice WITH PASSWORD '<new_pwd>';
```

3. Exit psql:

```
\q
exit
```

Change in Trifacta configuration

1. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
2. Modify the user and password properties for any of the following databases:

Database Name	Parameter area
Trifacta database	webapp.database.*
Jobs database	batch-job-runner.database.*
Scheduling database	scheduling-service.database.*
Time-based Trigger database	time-based-trigger-service.database.*
Configuration Service database	configuration-service.database.*
Artifact Storage Service database	artifact-storage-service.database.*

Job Metadata Service database	job-metadata-service.database.*
Authorization Service database	authorization-service.database.*
Orchestration Service database	orchestration-service.database.*
Optimizer Service database	optimizer-service.database.*
Secure Token Service database	secure-token-service.database.*
Connector Configuration Service database	connector-configuration-service.database.*

For more information, see *Database Parameter Reference*.

3. Save your changes and restart the platform.

Change Database Port

If needed, you can change the default port used by the database instance for the Trifacta® platform. Please complete the following commands to avoid conflicts between versions.

If you are changing the port within the database, those changes must be applied through your database system by an administrator. To limit downtime, port number changes should be done at the same time in the platform and the database. For more information, please see the documentation included with your database product.

NOTE: By default, PostgreSQL and the platform use port 5432 for communication. If that port is not available at install/upgrade time, the next available port is used, which is typically 5433. This change may occur if a previous version of PostgreSQL is on the same server. When a non-default port number is used, the platform must be configured to use it. For more information, see *Change Database Port*.

NOTE: By default, the ports for all databases are set to the default database port for PostgreSQL. For more information on the default database ports, see *Database Parameter Reference*.

Steps:

1. Stop the Trifacta services. For more information, see *Start and Stop the Platform* in the Install Guide.
2. Stop the database server.
3. Change the port number in the database administration console. For more information, see the documentation provided with your database distribution.
4. Start the database server. Verify that it is listening on the new port number.
5. Restart the platform.
6. Login to the application.
7. You can apply this change through the *Admin Settings Page* (recommended) or `trifacta-conf.json`. For more information, see *Platform Configuration Methods*.
8. Modify the `port` properties for any of the following databases:

Database Name	Parameter area
Trifacta database	<code>webapp.database.*</code>
Jobs database	<code>batch-job-runner.database.*</code>
Scheduling database	<code>scheduling-service.database.*</code>
Time-based Trigger database	<code>time-based-trigger-service.database.*</code>
Configuration Service database	<code>configuration-service.database.*</code>
Artifact Storage Service database	<code>artifact-storage-service.database.*</code>
Job Metadata Service database	<code>job-metadata-service.database.*</code>
Authorization Service database	<code>authorization-service.database.*</code>
Orchestration Service database	<code>orchestration-service.database.*</code>
Optimizer Service database	<code>optimizer-service.database.*</code>
Secure Token Service database	<code>secure-token-service.database.*</code>

Connector Configuration Service database	connector-configuration-service.database.*
--	--

For more information, see *Database Parameter Reference*.

9. Save your changes and restart the platform.
10. Verify that you can login and run a simple job. See *Verify Operations* in the Admin Guide.

Database Parameter Reference

Contents:

- *Trifacta database*
 - *Jobs database*
 - *Scheduling Service database*
 - *Time-based Trigger Service database*
 - *Configuration Service database*
 - *Artifact Storage Service database*
 - *Job Metadata Service database*
 - *Authorization Service database*
 - *Orchestration Service database*
 - *Optimizer Service database*
 - *Secure Token Service database*
 - *Connector Configuration Service database*
 - *Parameter Reference*
-

This section contains reference information on the relevant Trifacta platform parameters that can be modified for each database.

NOTE: Defaults are listed below. The default database server is PostgreSQL. If you have installed databases on a different server, updates are required.

NOTE: Parameters that you specify through the application must match the values that were applied to the database at the time of creation or that have been set through the database administration console.

Tip: You should modify the default password for each database.

Tip: The `maxPoolSize` for each database should be greater than or equal to the thread pool size for the related service. Otherwise, under heavy loads, database connection timeouts can be expected.

Parameters can be modified through:

- **Admin Settings Page:** If the application is working, then you are able to connect to the Trifacta database. Parameters can be modified by an administrator from within the application. See *Admin Settings Page*.

Tip: This method is recommended.

- `trifacta-conf.json`: If the application is not working, then an administrator must modify through the following file on the Trifacta node:
`/opt/trifacta/conf/trifacta-conf.json`
- For more information, see *Platform Configuration Methods* in the Configuration Guide.

Trifacta database

```
"webapp.database.username": "trifacta",
"webapp.database.logging": false,
"webapp.database.name": "trifacta",
"webapp.database.host": "localhost",
"webapp.database.password": "<pwd_trifactaDB>",
"webapp.database.type": "postgresql",
"webapp.database.port": 5432,
"webapp.database.mySqlServerTimezone": "",
"webapp.database.pool.maxIdleTimeInMillis": 30000,
"webapp.database.pool.maxConnections": 10,
```

The following parameters apply to the Trifacta database only:

Parameter	Description
logging	Set this value to <code>true</code> to enable logging on the Trifacta database.
pool.maxIdleTimeInMillis	Specifies the maximum permitted idle time for a database connection before it is automatically closed.
pool.maxConnections	Defines the maximum permitted database connections for the Trifacta database.

Jobs database

Modify the `batch-job-runner.database` settings:

```
"batch-job-runner.database.username": "trifactaactiviti",
"batch-job-runner.database.name": "trifacta-activiti",
"batch-job-runner.database.type": "postgresql",
"batch-job-runner.database.driver": "org.postgresql.Driver",
"batch-job-runner.database.host": "localhost",
"batch-job-runner.database.password": "<pwd_trifactaactivitiDB>",
"batch-job-runner.database.port": 5432,
"batch-job-runner.database.mySqlServerTimezone": "",
"batch-job-runner.database.verifyServerCertificate": false,
"batch-job-runner.database.poolMaxSize": 32,
"batch-job-runner.database.additionalConnectionProperties": "",
```

Jobs database thread pool size

The database thread pool size should be configured in conjunction with the other pool sizes for Batch Job Runner. For more information, see *Configure Batch Job Runner*.

Scheduling Service database

```
"scheduling-service.database.type": "POSTGRESQL",
"scheduling-service.database.host": "localhost",
"scheduling-service.database.port": "5432",
"scheduling-service.database.name": "trifactascheduling-service",
"scheduling-service.database.username": "trifactascheduling-service",
"scheduling-service.database.password": "<pwd_scheduling-serviceDB>",
"scheduling-service.database.mySqlServerTimezone": "",
"scheduling-service.database.verifyServerCertificate": false,
"scheduling-service.database.poolMaxSize": 20,
"scheduling-service.database.additionalConnectionProperties": "",
```


Time-based Trigger Service database

```
"time-based-trigger-service.database.type": "POSTGRESQL",
"time-based-trigger-service.database.host": "localhost",
"time-based-trigger-service.database.port": "5432",
"time-based-trigger-service.database.name": "trifactatimebasedtriggerservice",
"time-based-trigger-service.database.username": "trifactatimebasedtriggerservice",
"time-based-trigger-service.database.password": "<pwd_triggerserviceDB>",
"time-based-trigger-service.database.mySqlServerTimezone": "",
"time-based-trigger-service.database.verifyServerCertificate": false,
"time-based-trigger-service.database.poolMaxSize": 20,
"time-based-trigger-service.database.additionalConnectionProperties": "",
```

Configuration Service database

```
"configuration-service.database.type": "POSTGRESQL",
"configuration-service.database.host": "localhost",
"configuration-service.database.port": "5432",
"configuration-service.database.name": "trifactaconfigurationservice",
"configuration-service.database.username": "trifactaconfigurationservice",
"configuration-service.database.password": "<pwd_trifactaconfigurationserviceDB>",
"configuration-service.database.mySqlServerTimezone": "",
"configuration-service.database.verifyServerCertificate": false,
"configuration-service.database.poolMaxSize": 20,
"configuration-service.database.additionalConnectionProperties": "",
```

Artifact Storage Service database

```
"artifact-storage-service.database.type": "POSTGRESQL",
"artifact-storage-service.database.host": "localhost",
"artifact-storage-service.database.port": "5432",
"artifact-storage-service.database.name": "trifactaartifactstorageservice",
"artifact-storage-service.database.username": "trifactaartifactstorageservice",
"artifact-storage-service.database.password": "<pwd_trifactaartifactstorageserviceDB>",
"artifact-storage-service.database.mySqlServerTimezone": "",
"artifact-storage-service.database.verifyServerCertificate": false,
"artifact-storage-service.database.poolMaxSize": 20,
"artifact-storage-service.database.additionalConnectionProperties": "",
```

Job Metadata Service database

```
"job-metadata-service.database.type": "POSTGRESQL",
"job-metadata-service.database.host": "localhost",
"job-metadata-service.database.port": "5432",
"job-metadata-service.database.name": "trifactajobmetadataservice",
"job-metadata-service.database.username": "trifactajobmetadataservice",
"job-metadata-service.database.password": "<pwd_trifactajobmetadataserviceDB>",
"job-metadata-service.database.mySqlServerTimezone": "",
"job-metadata-service.database.verifyServerCertificate": false,
"job-metadata-service.database.poolMaxSize": 20,
"job-metadata-service.database.additionalConnectionProperties": "",
```

Authorization Service database

```
"authorization-service.database.type": "POSTGRESQL",
"authorization-service.database.host": "localhost",
"authorization-service.database.port": "5432",
"authorization-service.database.name": "trifactaauthorizationservice",
"authorization-service.database.username": "trifactaauthorizationservice",
"authorization-service.database.password": "<pwd_trifactaauthorizationserviceDB>",
"authorization-service.database.mySqlServerTimezone": "",
"authorization-service.database.verifyServerCertificate": false,
"authorization-service.database.poolMaxSize": 20,
"authorization-service.database.additionalConnectionProperties": "",
```

Orchestration Service database

```
"orchestration-service.database.type": "POSTGRESQL",
"orchestration-service.database.host": "localhost",
"orchestration-service.database.port": "5432",
"orchestration-service.database.name": "trifactaorchestrationservice",
"orchestration-service.database.username": "trifactaorchestrationservice",
"orchestration-service.database.password": "<pwd_trifactaorchestrationserviceDB>",
"orchestration-service.database.mySqlServerTimezone": "",
"orchestration-service.database.verifyServerCertificate": false,
"orchestration-service.database.poolMaxSize": 20,
"orchestration-service.database.additionalConnectionProperties": "",
```

Optimizer Service database

```
"optimizer-service.database.type": "POSTGRESQL",
"optimizer-service.database.host": "localhost",
"optimizer-service.database.port": "5432",
"optimizer-service.database.name": "trifactaoptimizerservice",
"optimizer-service.database.username": "trifactaoptimizerservice",
"optimizer-service.database.password": "<pwd_trifactaoptimizerservice>",
"optimizer-service.database.mySqlServerTimezone": "",
"optimizer-service.database.verifyServerCertificate": false,
"optimizer-service.database.poolMaxSize": 20,
"optimizer-service.database.additionalConnectionProperties": "",
```

Secure Token Service database

```
"secure-token-service.database.type": "POSTGRESQL",
"secure-token-service.database.host": "localhost",
"secure-token-service.database.port": "5432",
"secure-token-service.database.name": "trifactasecuretokenservice",
"secure-token-service.database.username": "trifactasecuretokenservice",
"secure-token-service.database.password": "<pwd_trifactasecuretokenservice>",
"secure-token-service.database.mySqlServerTimezone": "",
"secure-token-service.database.verifyServerCertificate": false,
"secure-token-service.database.poolMaxSize": 20,
"secure-token-service.database.additionalConnectionProperties": "",
```

Connector Configuration Service database

```
"connector-configuration-service.database.type": "POSTGRESQL",
"connector-configuration-service.database.host": "localhost",
"connector-configuration-service.database.port": "5432",
"connector-configuration-service.database.name": "trifactaconnectorconfigservice",
"connector-configuration-service.database.username": "trifactaconnectorconfigservice",
"connector-configuration-service.database.password": "<pwd_trifactaconnectorconfigservice>",
"connector-configuration-service.database.mySqlServerTimezone": "",
"connector-configuration-service.database.verifyServerCertificate": false,
"connector-configuration-service.database.poolMaxSize": 20,
"connector-configuration-service.database.additionalConnectionProperties": "",
```

Parameter Reference

The following generalized parameters apply to one or more of the databases.

Parameter	Description
host	Host of the database. Default value is <code>localhost</code> , meaning the database is hosted on the Trifacta node.
port	Port number for the database. Default value is <code>5432</code> for all databases, assuming that they are PostgreSQL. Modify this value accordingly if you have installed the databases onto a different type of database server.
name	Name of the database. This value should match what was used during installation.
user or username	The username to use to connect to the database.
password	Password to use to connect to the database. <div>NOTE: For each database, you should change the default password.</div> <p>This change must also be applied on the database server.</p> <ul style="list-style-type: none">• See <i>Change Database Passwords for PostgreSQL</i>.• See <i>Change Database Passwords for MySQL</i>.
type	The default database type is <code>POSTGRESQL</code> . <div>NOTE: Modify this value accordingly if you have installed the databases onto a different type of database server.</div> <div>NOTE: H2 database type is used for internal testing. It is not a supported database.</div>
driver	Name of the database. Do not modify.
mySqlServerTimezone	(MySQL DBs only) If populated, the MySQL connection string for the database is augmented with the specified timezone, which overrides the default mapping/detection of the timezone. If empty, the default MySQL timezone settings are used for the database. <div>NOTE: Leave this value as empty if the Trifacta node and your MySQL databases are located in the same timezone.</div> <p>For more information, see <i>Install Databases for MySQL</i>.</p>

verifyServerCertificate	<p>When true, the local database client does not check the validity of the database server certificate when connecting.</p> <p>Default is false.</p> <div> NOTE: Do not modify unless necessary for connectivity. </div>
poolMaxSize	<p>Maximum number of threads permitted in the connection pool for the database. Default is 20.</p> <div> NOTE: Do not modify unless necessary. </div>
additionalConnectionProperties	<p>Additional query parameters can be appended to the connection URL in the following form:</p> <div> param1Name=param1Value&param2Name=param2Value </div>

Install Databases on Amazon RDS

Contents:

- *Limitations*
 - *Pre-requisites*
 - *Initialize RDS instance*
 - *Configure the Trifacta platform for RDS*
 - *Create the databases*
 - *Configure non-default connections*
 - *Logging*
-

As needed, the Trifacta® databases can be installed as PostgreSQL DBs on Amazon RDS. Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud.

Limitations

Installation of the Trifacta databases on a remote server of which you do not have full administrative control requires additional configuration and cross-migration that is not covered in this documentation. Before you begin, please contact *Alteryx Support*.

- In Amazon RDS, databases must be installed in a supported version of PostgreSQL.

NOTE: For more information on upgrading the version of PostgreSQL in use on your RDS instance, see https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_UpgradeDBInstance.PostgreSQL.html

Pre-requisites

NOTE: You can use the suggested defaults below for sizing your RDS instance. If you have questions or concerns about sizing recommendations, please contact *Alteryx Support*.

- Admin access to an Amazon RDS account
- You must install the appropriate database client:
 - For more information, see *Install Database Client for PostgreSQL*.

Initialize RDS instance

Steps:

1. In your RDS dashboard, click **Launch a DB instance**.

NOTE: The RDS instance must be launched in the same Amazon region as the Trifacta node.

2. For Select Engine: Select **PostgreSQL**.
3. For Production?: Choose **Yes** if you are deploying the database for a production instance of the Trifacta platform. Otherwise, select **No**.

4. DB Engine: `postgres`
5. For the DB details, see below:

NOTE: Except as noted below, properties should be specified according to your enterprise requirements.

a. **Instance Specifications:**

- i. License Model: `postgresql-license`
 - ii. DB Engine Version: For more information on the supported versions of PostgreSQL, see *System Requirements* in the Planning Guide.
 - iii. Allocated Storage: at least 10 GB
6. For Advanced Settings, please apply the following settings:
- a. **Network and Security:**
- i. VPC security group must allow for access from the Trifacta platform.
- b. **Database Options:**
- i. Database Name: `trifacta`
 - ii. Database Port: 5432
1. The port number can be changed as needed. See *System Ports* in the Planning Guide.
7. Populate other properties according to your enterprise requirements.
8. To complete the set up click **Launch DB Instance**.
9. Create a master username and password for the DB instance.

NOTE: In the configuration instructions listed below, this master username is referenced as: `trifacta_rds`.

10. When the RDS DB instance is up and running, please collect the following information, which is used later:
- a. Public DNS
 - b. Port Number
 - c. Admin username
 - d. Admin password

Configure the Trifacta platform for RDS

Please complete the following steps to integrate the Trifacta platform with the DB instance you just created.

Steps:

1. In the RDS console, you must find the Public DNS endpoint for the RDS instance you created:
 - a. Under Instances, expand the name of the instance you created.
 - b. The DNS endpoint should be listed under the name in the Endpoint section.
2. Set the `host` for each database to the Public DNS endpoint for the RDS instance:

Database	Parameter
Main database	<code>webapp.database.host</code>
Jobs database	<code>batch-job-runner.database.host</code>
Scheduling database	<code>scheduling-service.database.host</code>
Time-based Trigger database	<code>time-based-trigger-service.database.host</code>
Configuration Service database	<code>configuration-service.database.host</code>

Artifact Storage Service database	artifact-storage-service.database.host
Job Metadata Service database	job-metadata-service.database.host
Authorization Service database	authorization-service.database.host
Orchestration Service database	orchestration-service.database.host
Optimizer Service database	optimizer-service.database.host
Secure Token Service database	secure-token-service.database.host
Connector Configuration Service database	connector-configuration-service.database.host

For more information, see *Database Parameter Reference*.

3. To set custom database names, usernames, and passwords:

- a. Edit `/opt/trifacta/conf/trifacta-conf.json`.
- b. For each database, you can review the database name, username, and password.

Database	Parameter area
Main database	webapp.database.*
Jobs database	batch-job-runner.database.*
Scheduling database	scheduling-service.database.*
Time-Based Trigger database	time-based-trigger-service.database.*
Configuration Service database	configuration-service.database.*
Artifact Storage Service database	artifact-storage-service.database.*
Job Metadata Service database	job-metadata-service.database.*
Authorization Service database	authorization-service.database.*
Orchestration Service database	orchestration-service.database.*
Optimizer Service database	optimizer-service.database.*
Secure Token Service database	secure-token-service.database.*
Connector Configuration Service database	connector-configuration-service.database.*

- c. Make changes in the file as needed and save.

Create the databases

NOTE: For the host, port, username, and password values in the following configuration, use the public DNS value for the RDS instance.

From the Trifacta node, switch to the Postgres user and run the following commands against the RDS Host for the master username:

```
su - postgres
psql -h <endpoint of RDS Instance> -U trifacta_rds postgres
```

NOTE: postgres is the name of the existing/default database.

Please do the following for each database, which initializes the DB and assigns the `trifacta` role to the `trifacta_rds` master username:

Trifacta database:

```
CREATE ROLE trifacta LOGIN ENCRYPTED PASSWORD '<pwd_trifacta>';
GRANT trifacta TO trifacta_rds;
CREATE DATABASE "trifacta" WITH OWNER trifacta;
```

Jobs database:

```
CREATE ROLE trifactaactiviti LOGIN ENCRYPTED PASSWORD '<pwd_trifactaactiviti>';
GRANT trifactaactiviti TO trifacta_rds;
CREATE DATABASE "trifacta-activiti" WITH OWNER trifactaactiviti;
```

Time-based trigger service database:

```
CREATE ROLE trifactatimebasedtriggerservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactatimebasedtriggerservice>';
GRANT trifactatimebasedtriggerservice TO trifacta_rds;
CREATE DATABASE "trifactatimebasedtriggerservice" WITH OWNER trifactatimebasedtriggerservice;
```

Scheduling service database:

```
CREATE ROLE trifactaschedulingservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaschedulingservice>';
GRANT trifactaschedulingservice TO trifacta_rds;
CREATE DATABASE "trifactaschedulingservice" WITH OWNER trifactaschedulingservice;
```

Configuration Service database:

```
CREATE ROLE trifactaconfigurationsservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaconfigurationsservice>';
GRANT trifactaconfigurationsservice TO trifacta_rds;
CREATE DATABASE trifactaconfigurationsservice WITH OWNER trifactaconfigurationsservice;
```

Artifact Storage Service database:

```
CREATE ROLE trifactaartifactstorageservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaartifactstorageservice>';
GRANT trifactaartifactstorageservice TO trifacta_rds;
CREATE DATABASE trifactaartifactstorageservice WITH OWNER trifactaartifactstorageservice;
```

Job Metadata Service database:

```
CREATE ROLE trifactajobmetadataservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactajobmetadataservice>';
GRANT trifactajobmetadataservice TO trifacta_rds;
CREATE DATABASE trifactajobmetadataservice WITH OWNER trifactajobmetadataservice;
```

Authorization Service database:


```
CREATE ROLE trifactaauthorizationservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaauthorizationservice>';
GRANT trifactaauthorizationservice TO trifacta_rds;
CREATE DATABASE trifactaauthorizationservice WITH OWNER trifactaauthorizationservice;
```

Orchestration Service database:

```
CREATE ROLE trifactaorchestrationservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaorchestrationservice>';
GRANT trifactaorchestrationservice TO trifacta_rds;
CREATE DATABASE trifactaorchestrationservice WITH OWNER trifactaorchestrationservice;
```

Optimizer Service database:

```
CREATE ROLE trifactaoptimizerservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaoptimizerservice>';
GRANT trifactaoptimizerservice TO trifacta_rds;
CREATE DATABASE trifactaoptimizerservice WITH OWNER trifactaoptimizerservice;
```

Secure Token Service database:

```
CREATE ROLE trifactasecuretokenservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactasecuretokenservice>';
GRANT trifactasecuretokenservice TO trifacta_rds;
CREATE DATABASE trifactasecuretokenservice WITH OWNER trifactasecuretokenservice;
```

Connector Configuration Service database:

NOTE: The default values for this database do not follow the naming conventions for other databases.

```
CREATE ROLE trifactaconnectorconfigservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaconnectorconfigservice>';
GRANT trifactaconnectorconfigservice TO trifacta_rds;
CREATE DATABASE trifactaconnectorconfigservice WITH OWNER trifactaconnectorconfigservice;
```

Configure non-default connections

If you have used non-default values for the username, password, host, or port value for either database, you must update platform configuration. For more information, see *Database Parameter Reference*.

Logging

1. To review database logs in RDS, locate the Instance details page in the RDS console.
2. Click **Recent Events and Logs**.
3. If your account has the appropriate permissions, all Trifacta database logs are available here.

Manual Database Install

Contents:

- *Prerequisites*
 - *Required special database permissions*
- *Manual DB Install for PostgreSQL*
 - *Launch PostgreSQL*
 - *Trifacta database*
 - *Jobs database*
 - *Scheduling database*
 - *Time-based Trigger database*
 - *Configuration Service database*
 - *Artifact Storage Service database*
 - *Job Metadata Service database*
 - *Authorization Service database*
 - *Orchestration Service database*
 - *Optimizer Service database*
 - *Secure Token Service database*
 - *Connector Configuration Service database*
- *Manual DB Install for MySQL*
 - *Launch MySQL*
 - *Trifacta database*
 - *Jobs database*
 - *Scheduling database*
 - *Time-based Trigger database*
 - *Configuration Service database*
 - *Artifact Storage Service database*
 - *Job Metadata Service database*
 - *Authorization Service database*
 - *Orchestration Service database*
 - *Optimizer Service database*
 - *Secure Token Service database*
 - *Connector Configuration Service database*

As needed, you can perform manual creation of users and individual databases used by the Trifacta® platform.

Tip: In most installation scenarios, manual database installs are not needed.

Prerequisites

1. The database server must be installed and available:
 - a. See *Install Databases for PostgreSQL*.
 - b. See *Install Databases for MySQL*.
2. The databases must be initialized.

NOTE: If you are using non-standard database names, users, passwords, port numbers, that information must be applied during installation and afterwards in the Trifacta platform configuration.

For more information, see *Configure the Databases*.

Required special database permissions

If you are manually installing any of the following databases, special database permissions are required for the user performing the installation and access. When the Trifacta platform is initialized for the first time, a cross-migration of data is performed between these databases, which is required for successful installation and enablement of the related services. This cross-migration process requires special database privileges.

To execute the cross-migration, the database user for the following databases requires special privileges:

- trifacta database
- trifactaauthorization service database

The privileges required for cross-migration depend on the database server on which these databases are being installed:

Database Type	Required permissions for database user
PostgreSQL	SUPERUSER
MySQL	FILE

If the cross-migration is run as part of the normal install or upgrade script, then these permissions are already set for the above databases. If for some reason, you are manually installing or upgrading these databases, the above privileges must be applied.

After install or upgrade, the above privileges are not required for normal operations on these databases. These privileges must be re-applied if cross-migrations need to be performed in the future.

Manual DB Install for PostgreSQL

Use the following steps to manually create the users and databases required by the platform in PostgreSQL.

Launch PostgreSQL

Prerequisites for PostgreSQL:

- The installing user must have write permissions to the directory from which the commands are executed.
- The installing user must have sudo privileges.

1. Switch to the Postgres user. Launch `psql`.

```
sudo su - postgres
```

NOTE: Unless the port number for postgres has been modified, it should be listening at the default value: 5432.

2. Launch `psql` using the following command, applying the admin password when prompted:

```
psql --host=${POSTGRES_HOST_NAME} --port=${POSTGRES_HOST_PORT} --username=${POSTGRES_ADMIN_USERNAME} --password=${POSTGRES_ADMIN_PASSWORD}
```

where:

Parameter	Description
host	Public DNS value for database server instance
port	Port number value for database server instance
username	Admin username for database server instance
password	Password for admin username

3. Execute the following commands using the `postgres` user.

NOTE: The values in platform configuration must match the values that you use below. Below are the default values.

Trifacta database

```
CREATE ROLE trifacta LOGIN ENCRYPTED PASSWORD '<pwd_trifacta>';
CREATE DATABASE trifacta WITH OWNER trifacta;
```

Jobs database

```
CREATE ROLE trifactaactiviti WITH LOGIN ENCRYPTED PASSWORD '<pwd_trifactaactiviti>';
CREATE DATABASE "trifacta-activiti" WITH OWNER trifactaactiviti;
```

Scheduling database

```
CREATE ROLE trifactaschedulingservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaschedulingservice>';
CREATE DATABASE trifactaschedulingservice WITH OWNER trifactaschedulingservice;
```

Time-based Trigger database

```
CREATE ROLE trifactatimebasedtriggerservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactatimebasedtriggerservice>';
CREATE DATABASE trifactatimebasedtriggerservice WITH OWNER trifactatimebasedtriggerservice;
```

Configuration Service database

```
CREATE ROLE trifactaconfigurationservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaconfigurationservice>';
CREATE DATABASE trifactaconfigurationservice WITH OWNER trifactaconfigurationservice;
```

Artifact Storage Service database

```
CREATE ROLE trifactaartifactstorageservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaartifactstorageservice>';
CREATE DATABASE trifactaartifactstorageservice WITH OWNER trifactaartifactstorageservice;
```

Job Metadata Service database

```
CREATE ROLE trifactajobmetadataservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactajobmetadataservice>';
CREATE DATABASE trifactajobmetadataservice WITH OWNER trifactajobmetadataservice;
```

Authorization Service database

```
CREATE ROLE trifactaauthorizationservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaauthorizationservice>';  
CREATE DATABASE trifactaauthorizationservice WITH OWNER trifactaauthorizationservice;
```

Orchestration Service database

```
CREATE ROLE trifactaorchestrationservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaorchestrationservice>';  
CREATE DATABASE trifactaorchestrationservice WITH OWNER trifactaorchestrationservice;
```

Optimizer Service database

```
CREATE ROLE trifactaoptimizerservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaoptimizerservice>';  
CREATE DATABASE trifactaoptimizerservice WITH OWNER trifactaoptimizerservice;
```

Secure Token Service database

```
CREATE ROLE trifactasecuretokenservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactasecuretokenservice>';  
CREATE DATABASE trifactasecuretokenservice WITH OWNER trifactasecuretokenservice;
```

Connector Configuration Service database

NOTE: The default values for this database do not follow the naming conventions for other databases.

```
CREATE ROLE trifactaconnectorconfigservice LOGIN ENCRYPTED PASSWORD '<pwd_trifactaconnectorconfigservice>';  
CREATE DATABASE trifactaconnectorconfigservice WITH OWNER trifactaconnectorconfigservice;
```

Manual DB Install for MySQL

Use the following steps to manually initialize the databases required by the platform in MySQL.

Launch MySQL

Prerequisites for MySQL:

- The installing user must have write permissions to the directory from which the commands are executed.
- The installing user must have sudo privileges.

1. Login to MySQL as the root user. When prompted, enter the password for root user:

```
mysql -u root -p
```

2. Execute the following commands using the root user.

NOTE: The values in platform configuration must match the values that you use below. Below are the default values.

Trifacta database

```
Create USER 'trifacta' IDENTIFIED BY '<pwd_trifacta>';  
CREATE DATABASE trifacta;  
Grant all on trifacta.* to trifacta;
```

Jobs database

```
Create USER 'trifactaactiviti' IDENTIFIED BY '<pwd_trifactaactiviti>';  
CREATE DATABASE trifacta-activiti;  
Grant all on trifacta-activiti.* to trifactaactiviti;
```

Scheduling database

```
Create USER 'trifactaschedulingservice' IDENTIFIED BY '<pwd_trifactaschedulingservice>';  
CREATE DATABASE trifactaschedulingservice;  
Grant all on trifactaschedulingservice.* to trifactaschedulingservice;
```

Time-based Trigger database

```
Create USER 'trifactatimebasedtriggerservice' IDENTIFIED BY '<pwd_trifactatimebasedtriggerservice>';  
CREATE DATABASE trifactatimebasedtriggerservice;  
Grant all on trifactatimebasedtriggerservice.* to trifactatimebasedtriggerservice;
```

Configuration Service database

```
Create USER 'trifactaconfigurationservice' IDENTIFIED BY '<pwd_trifactaconfigurationservice>';  
CREATE DATABASE trifactaconfigurationservice;  
Grant all on trifactaconfigurationservice.* to trifactaconfigurationservice;
```

Artifact Storage Service database

```
Create USER 'trifactaartifactstorageservice' IDENTIFIED BY '<pwd_trifactaartifactstorageservice>';  
CREATE DATABASE trifactaartifactstorageservice;  
Grant all on trifactaartifactstorageservice.* to trifactaartifactstorageservice;
```

Job Metadata Service database

```
Create USER 'trifactajobmetadataservice' IDENTIFIED BY '<pwd_trifactajobmetadataservice>';  
CREATE DATABASE trifactajobmetadataservice;  
Grant all on trifactajobmetadataservice.* to trifactajobmetadataservice;
```

Authorization Service database

```
Create USER 'trifactaauthorizationservice' IDENTIFIED BY '<pwd_trifactaauthorizationservice>';  
CREATE DATABASE trifactaauthorizationservice;  
Grant all on trifactaauthorizationservice.* to trifactaauthorizationservice;
```

Orchestration Service database

```
Create USER 'trifactaorchestrationservice' IDENTIFIED BY '<pwd_trifactaorchestrationservice>';  
CREATE DATABASE trifactaorchestrationservice;  
Grant all on trifactaorchestrationservice.* to trifactaorchestrationservice;
```

Optimizer Service database

```
Create USER 'trifactaoptimizerservice' IDENTIFIED BY '<pwd_trifactaoptimizerservice>';  
CREATE DATABASE trifactaoptimizerservice;  
Grant all on trifactaoptimizerservice.* to trifactaoptimizerservice;
```

Secure Token Service database

```
Create USER 'trifactasecuretokenservice' IDENTIFIED BY '<pwd_trifactasecuretokenservice>';  
CREATE DATABASE trifactasecuretokenservice;  
Grant all on trifactasecuretokenservice.* to trifactasecuretokenservice;
```

Connector Configuration Service database

NOTE: The default values for this database do not follow the naming conventions for other databases.

```
Create USER 'trifactaconnectorconfigservice' IDENTIFIED BY '<pwd_trifactaconnectorconfigservice>';  
CREATE DATABASE trifactaconnectorconfigservice;  
Grant all on trifactaconnectorconfigservice.* to trifactaconnectorconfigservice;
```



Copyright © 2022 - Trifacta, Inc.
All rights reserved.