# Oracle® Financial Services Crime and Compliance Studio Application

Deployment Guide (Using Kubernetes)

Release 8.0.7.2.0

February 2020

**E91246-01**

Deployment Guide, Release 8.0.7.2.0

E91246-01

# Contents

# Document Control

| Version Number | Revision Date | Changes Done |
|---|---|---|
| 8.0.7.1.0 | Updated: October 2019 | Created the first version of Oracle Financial Services Crime and Compliance Studio Deployment Guide for v8.0.7.1.0 Release. |
| 8.0.7.2.0 | Updated: February 2020 | • Deploy Studio application with or without BD on the Kubernetes cluster. For more information, see Deploying Studio with BD and Deploying Studio without BD. <br> • The support for the Data Forward service has been deprecated. <br> • Configuration for the newly introduced interpreters such as Spark and PySpark interpreters. For more information, see Configuring Interpreters |

# Preface

This section provides supporting information for the Oracle Financial Services Crime and Compliance Studio application Installation Guide and includes the following topics:

- Summary
- Audience
- Related Documents

## Summary

Before you begin the deployment, ensure that you have access to the Oracle Support Portal with valid login credentials to quickly notify us of any issues at any stage. You can obtain the login credentials by contacting the Oracle Support.

## Audience

This document is intended for System Engineers who are responsible for deploying and configuring the OFS Crime and Compliance Studio application.

The document assumes that you have experience in installing Enterprise components. The basic knowledge of the following is recommended:

- UNIX commands
- Database concepts
- Big Data
- Kubernetes
- Docker

## Related Documents

You can access the following additional documents related to the OFS Crime and Compliance Studio application from the Oracle Help Center (OHC) Documentation Library:

- *Oracle Financial Services Crime and Compliance Studio Installation Guide*
- *Oracle Financial Services Crime and Compliance Studio Administration Guide*
- *Oracle Financial Services Crime and Compliance Studio User Guide*
- *Oracle Financial Services Crime and Compliance Studio Data Model Guides*
- *Oracle Financial Services Crime and Compliance Studio Release Notes and Readme*

# 1 Deployment Overview

This chapter provides the information required to understand the deployment of the Oracle Financial Services (OFS) Crime and Compliance Studio application on the Kubernetes cluster.

## Introduction

This release (v8.0.7.2.0) of the OFS Crime and Compliance Studio application pack can be used to deploy a new instance of the Studio application (v8.0.7.2.0) with or without BD on the Kubernetes cluster.

## Quick Start Steps to Deploy Studio with BD on Kubernetes Cluster

*Table 1–1 Quick Start Steps to Deploy Studio with BD on Kubernetes Cluster*

| Sl. No. | Steps | Reference Links |
|---|---|---|
| 1. | Prepare for Deployment | 1. Prerequisites<br>2. Hardware and Software Requirements<br>3. Prerequisite Environmental Settings<br>4. Performing Common Pre-installation Tasks<br>  1. Obtaining the Software<br>  2. Extracting the Software<br>5. Required File Structure<br>6. Interpreter Settings |
| 2. | Deploy the Studio application with BD on Kubernetes Cluster | 1. Deploying Studio with BD<br>  1. Configuring Wallet<br>  2. Configuring studio-env.yml File<br>  3. Installing ETL Services<br>  4. Verifying Resource Allocation for Studio Services<br>  5. Deploying Studio on Kubernetes Cluster<br>2. Verifying the Deployment<br>3. Launching the FCC Studio Application |
| 3. | Post-deployment configuration | • Configuring Interpreters<br>• Performing OFSAA Configuration for Batch Execution<br>• Performing Hive Data Movement |

## Quick Start Steps to Deploy Studio without BD on

## Kubernetes Cluster

***Table 1–2 Quick Start Steps to Deploy Studio without BD on Kubernetes Cluster***

| Sl. No. | Steps | Reference Links |
|---|---|---|
| 1. | Prepare for Deployment | 1. Prerequisites<br>2. Hardware and Software Requirements<br>3. Prerequisite Environmental Settings<br>4. Performing Common Pre-installation Tasks<br>   1. Obtaining the Software<br>   2. Extracting the Software<br>5. Interpreter Settings |
| 2. | Deploy the Studio application without BD on Kubernetes Cluster | 1. Deploying Studio without BD<br>   1. Configuring Wallet<br>   2. Configuring studio-env.yml File<br>   3. Verifying Resource Allocation for Studio Services<br>   4. Deploying Studio on Kubernetes Cluster<br>2. Verifying the Deployment<br>3. Launching the FCC Studio Application |
| 3. | Post-deployment configuration | Configuring Interpreters |

# 2 Preparing for Deployment

This chapter provides the necessary information to review before deploying the Studio application. It includes the following sections:

## Prerequisites

The Linux machine must satisfy the following conditions:

- The Studio application can be installed with or without BD. To install the Studio application with BD, ensure the BD application pack is installed.
- Kubernetes (k8s) cluster must be installed to include the following:
  - Registry to store docker images.
  - Minimum of 8 GB memory (inclusive of all nodes) available for the installation. The actual memory requirement depends on the workload/container size configuration.
  - Must not contain a namespace called, fccs. If it already exists, delete the namespace before running the deployment script.
- Docker and kubectl must be installed.
- kubectl is configured (that is, connected to cluster where you want to install FCC Studio).
- Docker has push access to a private registry.
- 12GB free space is available to store the Studio Installer zip file in some directory.
- 45GB free space is available in the docker root directory. Run `docker info` command to find the docker root directory.

## Hardware and Software Requirements

The hardware and software required to deploy Studio are as follows:

***Table 2–1 Hardware and Software Requirements***

| Hardware/Software Category | Component Version |
|---|---|
| Browser | • Chrome 57.x<br>• Firefox 52.x |
| Java Version | java 8 |
| Docker Registry | • Docker registry must be present to store docker images<br>• Min of 45GBspace is required to save docker images |
| Database Server | • Oracle Database Server 12c Release 2 (12.2.0.1+) Enterprise Edition<br>• Oracle R Enterprise 1.5.1 with Oracle R Distribution 3.3.0 |
| Hadoop Cluster | • HDP Version 2.5<br>• Hadoop-2.7.3+hdp2.5+844<br>• Hive-1.2.1+hdp2.5+350<br>• Sqoop1 V 1.4.4+hdp2.5+67<br>• Oracle Loader For Hadoop (OLH) V 3.2<br>• Hive JDBC Connectors V 2.5.15<br>• Spark 2.4.0 |
| **Kubernetes Cluster** | |
| Processing Server | • RHEL 7.4+<br>• SFTP<br>• Oracle JRE Standard Edition 1.8.x(with JCE)<br>• Kubernetes(K8s) cluster. For more information, see Prerequisites. |
| PGX (Graph) Server | • RHEL 7.4+<br>• Kubernetes(K8s) cluster. For more information, see Prerequisites. |
| ETL Namematching Server | • RHEL 7.4+<br>• Kubernetes(K8s) cluster to be able to run the ETL and Name Matching server. For more information, see Prerequisites. |
| **BIG DATA** | |
| Cloudera Distribution Hadoop 5.12 | • CDH Version 5.12<br>• Hadoop-2.5.0+cdh5.3.3+844<br>• Hive-0.13.1+cdh5.3.3+350<br>• Sqoop1 V 1.4.4+cdh5.3.3+67<br>• The `.profile` file must be present with the `SPARK_HOME` and `PYTHON_HOME` parameters already set.<br>• Set `spark2-shell alias` in the `.profile` file as follows:<br>`alias spark2-shell=spark-shell` |
| Cloudera Hive Connectors | Hive JDBC Connectors V 2.5.15 |
| Hadoop Security Protocol | • Kerberos R release 1.6.1<br>• Sentry-1.4.0 |

## Prerequisite Environmental Settings

The prerequisite environmental settings to be performed before beginning the deployment of the Studio application are as follows:

***Table 2–2 Prerequisite Information***

| Category | Expected Value |
|---|---|
| Java Settings | • `PATH` in the `.profile` file must be set to include `kubectl` and the Java Runtime Environment (java 8) absolute path.<br>**Note:**<br>• Ensure the absolute path to `JRE/bin` is set at the beginning of the `PATH` variable.<br>For example: `PATH=/usr/java/jre1.8/bin:$PATH`<br>• Ensure no SYMBOLIC links to JAVA installation are set in the `PATH` variable. |
| Oracle Database Settings | **Note**: This setting is required only if the Wallet has to be created on the same server as that of the Studio server.<br>**Oracle Processing Server**<br>• `ORACLE_HOME` must be set in the `.profile` file pointing to the appropriate Oracle DB Client installation.<br>• `PATH` in the `.profile` file must be set to include the appropriate `$ORACLE_HOME/bin` path. |
| Download Directory | Indicates the directory where the product installer zip file is downloaded/copied. The user permission must be set to 755 for this Download directory. |
| Installation Directory | Indicates the directory where the product installer zip file is extracted and the installation files are placed. The user permission must be set to 755 for this installation directory.<br>**Note**: The Installation and the Download Directory can be the same if the product installer zip file is not copied separately to another directory. |
| OS Locale | Linux: `en_US.utf8`<br>Execute the following command to check the locale installed:<br>`locale -a \| grep -i 'en_US.utf'`<br>The installed locale is displayed. |
| Studio Schema | **1.** Create a new Oracle Database schema user using the following script:<br>`CREATE USER <Studio Schema User Name> IDENTIFIED BY <Password>;`<br>A new oracle Database schema is created.<br>**2.** Grant the permissions that are given in the next row.<br>This newly created schema is referred to as Studio Schema. |

***Table 2–2 Prerequisite Information***

| Category | Expected Value |
|---|---|
| Oracle Database Schema Settings | Grant the following permissions to the newly created Oracle Database Schema:<br>GRANT create session TO <Studio Schema User>;<br>GRANT create table TO <Studio Schema User>;<br>GRANT create view TO <Studio Schema User>;<br>GRANT create any trigger TO <Studio Schema User>;<br>GRANT create any procedure TO <Studio Schema User>;<br>GRANT create sequence TO <Studio Schema User>;<br>GRANT execute on dbms_rls TO <Studio Schema User>;<br>GRANT execute on sys.dbms_session TO <Studio Schema User>;<br>ALTER USER <Studio Schema User> QUOTA 100M ON users;<br>GRANT create sequence TO <Studio Schema User>;<br>GRANT create SYNONYM TO <Studio Schema User>;<br>GRANT create any context TO <BD Schema User>;<br>GRANT execute on dbms_rls TO <BD Schema User>;<br>GRANT ALL privileges TO <Studio Schema User>; |
| Wallet Settings | Set a password store with Oracle Wallet. For more information, see Appendix D, "Setting Up Password Stores with Oracle Wallet" |

## Performing Common Pre-installation Tasks

The common pre-installation tasks that you must perform before installing Studio are:

- Obtaining the Software
- Extracting the Software

### Obtaining the Software

To download and copy the Studio application installer software, follow these steps:

1. Login to My Oracle Support with a valid Oracle account and search for the Bug ID **30877695** under the *Patches & Updates* tab.
2. Download the installer archive `OFS_FCCM_STUDIO_8.0.7.2.0_Linux.zip` file to the download directory (in Binary Mode) on the setup identified for Studio installation.

### Extracting the Software

1. Extract the contents of the `OFS_FCCM_STUDIO_8.0.7.2.0_LINUX.zip` installer archive file in the download directory using the following command:

```
unzip -a OFS_FCCM_STUDIO_8.0.7.2.0_LINUX.zip
```

The Studio installer file is extracted and the `OFS_FCCM_STUDIO` folder is obtained. The `OFS_FCCM_STUDIO` folder is referred to as `<Studio_Installation_Path>`.

**Note:**

Do not rename the application installer folder name on extraction from the archive.

2. Navigate to the download directory where the installer archive is extracted and assign execute permission to the installer directory using the following command:

   ```
   chmod 0755 OFS_FCCM_STUDIO -R
   ```

## Required File Structure

The Studio application must be installed with certain additional services such as Cross Language Name Matching, ETL, and Hadoop/Spark services.

To install the additional services, you must obtain the required configuration files as follows from the Big Data installation path.

> **Note:**
>
> These files must be kept ready and provided in the following file structure which is used during Studio installation.

- Hadoop Cluster
  - `core-site.xml`
  - `hadoop-env.sh`
  - `hdfs-site.xml`
  - `hive-env.sh`
  - `hive-site.xml`
  - `log4j.properties`
  - `mapred-site.xml`
  - `redaction-rules.json`
  - `spark-defaults.conf`
  - `spark-env.sh`
  - `ssl-client.xml`
  - `topology.map`
  - `topology.py`
  - `yarn-site.xml`
- Kerberos Files
  - `krb5.conf`
  - `ofsaa.keytab`

> **Note**:
>
> Rename your `.keytab` file to `ofsaa.keytab`.

- Certificates
  - `key.store.jks`
  - `trusted.store.jks`
- Additional Jars

- `hive-exec-1.1.0-cdh5.13.0.jar`
- `HiveJDBC4.jar`
- `hive-metastore-1.1.0-cdh5.13.0.jar`
- `hive-service-1.1.0-cdh5.13.0.jar`

---

**Note**:

- The version of the jars is client/user-specific. These jars can be obtained from existing jars of Cloudera installation.
- The `HiveJDBC4.jar` file is not available in the Cloudera setup. You must download the same from the Cloudera website.

---

## Interpreter Settings

---

**Note:**

Perform the following pre-requisite settings only for the interpreters that you need.

---

***Table 2–3 Interpreter Settings***

| Interpreter | Prerequisite Settings |
|---|---|
| fcc-jdbc | No additional configuration is required.<br>**Note:** A new interpreter called vanilla jdbc interpreter is provided, which will connect to the Studio schema. Studio with non-BD can use this interpreter instead of the fcc-jdbc interpreter. |
| fcc-ore | see Appendix E, "Installing RServe Manually" |
| fcc-pyspark | • Install the py4j package in the Spark cluster.<br>• Install the Livy server (0.5.0) on the master node of the Big Data cluster. |
| fcc-python. | Install the py4j package. |
| fcc-spark-scala | Install the Livy server (0.5.0) on the master node of the Big Data cluster. |
| fcc-spark-sql | Install the Livy server (0.5.0) on the master node of the Big Data cluster. |
| jdbc | No additional configuration is required. |
| md | No additional configuration is required. |
| pgql | No additional configuration is required. |
| pgx-algorithm | No additional configuration is required. |
| pgx-java | No additional configuration is required. |
| pyspark | For the required configuration, see Configuring PySpark Interpreter. |
| spark: | For the required configuration, see Configuring Spark Interpreter. |

## Configuring Spark Interpreter

- Prerequisites
- Configuration

### Prerequisites

To operate the Spark interpreter in local mode or Yarn mode, the `spark.master` property must be set accordingly. For information on setting the Spark Master properties, see Appendix J, "Setting Spark Master".

#### Local Mode

No additional configuration is required to operate the Spark interpreter in local mode.

#### Yarn Mode

To operate the Spark interpreter in yarn mode, follow these steps:

- Provide custom Spark libraries. For more information, see Appendix I, "Providing Spark Libraries".
- The cluster's Hadoop client-side configuration files that include XML files such as `yarn-site.xml` are required and must be supplied with the Spark libraries. These files are available in the Hadoop configuration directory (`HADOOP_CONF_DIR`) of the cluster or can be downloaded from the cluster manager's UI if you are using a Cloudera cluster.

### Configuration

Spark interpreter configuration can be divided into the following categories:

- Configuration related to deployment

  These properties can be set either in the Spark libraries, for example, the `spark-defaults.conf` file, or through the system environment variable, `SPARK_CONF`, for example, `SPARK_CONF="--conf spark.driver.memory=2g"`.

  ---

  **Note:** These properties cannot be changed when the Spark interpreter is running.

  ---

- Configuration related to Spark runtime control

  These properties can be set from the ***Interpreters*** page of the Studio application UI, this includes properties such as `spark.executor.memory`.

  ---

  **Note:** The properties related to the driver cannot be set during runtime and are considered deployment configuration. The properties related to the executors can be set during runtime. Hence, the latter option of runtime control configuration is preferred.

  ---

A list of possible properties can be found in the Spark's Official Documentation. All the properties prefixed with the term "zeppelin" that are listed in the Zeppelin Spark Configuration Document, can also be set through the ***Interpreters*** page in the Studio application.

## Configuring PySpark Interpreter

- Prerequisites
- Configuration

### Prerequisites

The PySpark interpreter has the same prerequisites as that of the Spark interpreter. For more information, see Configuring Spark Interpreter. In addition, all Spark components must be configured to use the same Python version.

### Configuration

The PySpark interpreter can be configured through the Spark interpreter with the only exception being the Python version used. By default, the Python version is set to 3, which can be changed either in the interpreter JSON files before the startup or from the ***Interpreters*** page of the Studio application UI during runtime by changing the following properties:

- In the ***Spark Interpreter Settings*** page of the Studio application UI (or `spark.json` file), change the value of the `spark.pyspark.python` property to the path of the Python executable that is to be used by the Spark executors.
- In the ***PySpark Interpreter Settings*** page of the Studio application UI (or `pyspark.json` file), change the value of the `zeppelin.pyspark.python` property to the path of the Python executable that is to be used by the Spark driver.

To ensure that the two Python versions match in the case where your components run on different machines, you can use Appendix K, "Using Python Virtual Environments with PySpark". This step is not needed if different machines have matching python versions available.

# 3

# Deploying Studio with BD

This chapter provides the necessary information to deploy a new instance of the Crime and Compliance Studio application on the Kubernetes cluster. It includes the following sections:

1. Deploying Studio with BD
2. Verifying the Deployment
3. Launching the FCC Studio Application

## Deploying Studio with BD

To deploy Studio, follow these steps:

1. Configuring Wallet
2. Configuring studio-env.yml File
3. Installing ETL Services
4. Verifying Resource Allocation for Studio Services
5. Deploying Studio on Kubernetes Cluster

### Configuring Wallet

To configure wallets, follow these steps:

1. Create a wallet. For information on creating wallets, see Appendix D, "Setting Up Password Stores with Oracle Wallet".
2. Copy the wallet files, `cwallet.sso, ewallet.p12` and `tnsnames.ora`, and place in the `<Studio_Installation_Path>/configmaps/wallet` path.

### Configuring studio-env.yml File

To configure the `studio-env.yml` file, follow these steps:

1. Login to the server as a non-root user.
2. Navigate to the `<Studio_Installation_Path>/secrets/studio-env.yml` file.
3. Configure the `studio-env.yml` file as mentioned in Table 3–1.

**Note:**

- You must manually set the Interaction Variable parameter values. If a value is not applicable, enter NA and ensure that the value is not entered as NULL.
- Do not alter the parameter values that are already set in the `studio-env.yml` file

***Table 3–1 studio-env.yml Parameters***

| InteractionVariable Name | Significance | Required |
|---|---|---|
| apiVersion | For example: v1 | Yes |
| kind | For example: Secret | Yes |
| **metadata** | | |
| name | For example: studio-env | Yes |
| **stringData** | | |
| REALM | For example: `com.oracle.ofss.fccm.studio.datastudio.auth.FCCMRealm` | Yes |
| NON_OFSAA | Enter "false" to deploy the Studio application with BD on the Kubernetes cluster. | Yes |
| OFSAA_SERVICE_URL | Indicates the URL of the OFSAA instance. Do not enter '/' at the end of the URL. **Note:** • For OFSAAAI, the value must be in the following format: `https://<HostName>:<PortNo>/<ContextName>/rest-api` | Yes |
| LIVY_HOST_URL | Indicates the URL of the Livy application. The format for the URL is as follows: `http://<HostName>:<PortNo>` | Yes, only if the Spark-sql, Spark-scala and/or pyspark interpreters are to be used. |
| STUDIO_DB_ HOSTNAME | Indicates the hostname of the database where Studio schema is created. | Yes |
| STUDIO_DB_PORT | Indicates the port number where Studio schema is created. | Yes |
| STUDIO_DB_SERVICE_ NAME | Indicates the service name of the database where Studio schema is created. | Yes |
| STUDIO_DB_SID | Indicates the SID of the database where Studio schema is created. | Yes |
| STUDIO_DB_ USERNAME | Indicates the username of the Studio Schema (newly created Oracle Schema). | Yes |
| STUDIO_DB_ PASSWORD | Indicates the password for the newly created schema. | Yes |
| STUDIO_ALIAS_NAME | Indicates the Studio alias name. For more information, see Appendix D, "Setting Up Password Stores with Oracle Wallet". **Note**: Enter the alias name that was created during wallet creation. | Yes |
| STUDIO_WALLET_ LOCATION | **Note:** The value is already set. Do not change the value. | Yes |

*Table 3–1 (Cont.) studio-env.yml Parameters*

| InteractionVariable Name | Significance | Required |
|---|---|---|
| STUDIO_TNS_ADMIN_PATH | **Note:** The value is already set. Do not change the value. | Yes |
| BD_CONFIG_HOSTNAME | Indicates the hostname of the database where BD config schema is installed. | Yes |
| BD_CONFIG_PORT | Indicates the port of the database where BD config schema is installed. | Yes |
| BD_CONFIG_SERVICE_NAME | Indicates the service name of the database where BD config schema is installed. | Yes |
| BD_CONFIG_SID | Indicates the SID of the database where BD config schema is installed. | Yes |
| BD_CONFIG_USERNAME | Indicates the username for the BD config schema. | Yes |
| BD_CONFIG_PASSWORD | Indicates the password for the BD config schema. | Yes |
| BD_CONFIG_ALIAS_NAME | Indicates the BD config alias name. For more information, see Appendix D, "Setting Up Password Stores with Oracle Wallet". **Note**: Enter the alias name that was created during wallet creation. | Yes |
| BD_CONFIG_WALLET_LOCATION | **Note:** The value is already set. Do not change the value. | Yes |
| BD_CONFIG_TNS_ADMIN_PATH | **Note:** The value is already set. Do not change the value. | Yes |
| BD_ATOMIC_HOSTNAME | Indicates the BD atomic schema hostname. | Yes |
| BD_ATOMIC_PORT | Indicates the BD atomic schema port number. | Yes |
| BD_ATOMIC_SERVICE_NAME | Indicates the BD atomic schema service name. | Yes |
| BD_ATOMIC_SID | Indicates the BD atomic schema SID. | Yes |
| BD_ATOMIC_USERNAME | Indicates the username of the BD atomic schema. | Yes |
| BD_ATOMIC_PASSWORD | Indicates the password of the BD atomic schema. | Yes |
| BD_ATOMIC_ALIAS_NAME | Indicates the BD atomic alias name. For more information, see Appendix D, "Setting Up Password Stores with Oracle Wallet". **Note**: Enter the alias name that was created during wallet creation. | Yes |
| BD_ATOMIC_WALLET_LOCATION | **Note:** The value is already set. Do not change the value. | Yes |
| BD_ATOMIC_TNS_ADMIN_PATH | **Note:** The value is already set. Do not change the value. | Yes |
| FSINFODOM | Indicates the name of the OFSAA or BD Infodom. | Yes |
| FSSEGMENT | Indicates the name of the OFSAA or BD segment. | Yes |
| DATAMOVEMENT_LINK_NAME | • If the newly created schema is in a different database host, you must create a DB link and provide the same link in this parameter. Alternatively, you can provide the source schema name. If no DB link is present, provide the <SCHEMA_NAME> in this parameter. • If the newly created schema is in the same database host, the value for this parameter is the user name of the BD atomic schema. | Yes |

*Table 3–1 (Cont.) studio-env.yml Parameters*

| InteractionVariable Name | Significance | Required |
|---|---|---|
| DATAMOVEMENT_ LINK_TYPE | If the DB link is used, enter DBLINK in this field. If the DB link is not used, enter SCHEMA in this field. | Yes |
| HADOOP_ CREDENTIAL_ PROVIDER_PATH | Indicates the path where Hadoop credential is stored. | Yes |
| HADOOP_PASSWORD_ ALIAS | Indicates the Hadoop alias given when creating the Hadoop credentials.<br>**Note**: Enter the alias name that was created during wallet creation.<br>For information on how to create credential keystore, see Creating Credential Keystore | Yes |
| Hive_Host_Name | Indicates the Hive hostname. | Yes |
| Hive_Port_number | Indicates the Hive port number.<br>Contact System Administrator to obtain the port number. | Yes |
| HIVE_PRINCIPAL | Indicates the Hive Principal.<br>Contact System Administrator to obtain HIVE_PRINCIPAL. | Yes |
| HIVE_SCHEMA | Indicates the new Hive schema name. | Yes |
| JAAS_CONF_FILE_ PATH | Created for future use. | No |
| Krb_Host_FQDN_Name | Indicates the Kerberos host FQDN name. | Yes |
| Krb_Realm_Name | Indicates the Kerberos realm name. | Yes |
| Krb_Service_Name | Indicates the Kerberos service name.<br>Example: Hive | Yes |
| KRB5_CONF_FILE_ PATH | Created for future use. | No |
| security_krb5_kdc_server | Created for future use. | No |
| security_krb5_realm | Created for future use. | No |
| server_kerberos_keytab_ file | Created for future use. | Yes |
| server_kerberos_principal | Created for future use. | Yes |
| SQOOP_ HOSTMACHINE_ USER_NAME | Indicates the user name of the Big Data server where SQOOP will run. | Yes |
| SQOOP_PARAMFILE_ PATH | **1.** Create a file with the name `sqoop.properties` in the Big Data server and add the following entry to the same:<br>`oracle.jdbc.mapDateToTimestamp=false`<br>**2.** Enter the location of the `sqoop.properties` file in the `SQOOP_ PARAMFILE_PATH` parameter.<br>Example: `/scratch/ofsaa/`<br>**Note**: Ensure that the location name ends with a '/'. | Yes |
| SQOOP_PARTITION_ COL | Indicates the column in which the HIVE table is partitioned.<br>The value must be `SNAPSHOT_DT` | Yes |
| SQOOP_TRG_ HOSTNAME | Indicates the hostname of the Big Data server where SQOOP will run.<br>Example: `<HostName>` | Yes |

***Table 3–1 (Cont.) studio-env.yml Parameters***

| InteractionVariable Name | Significance | Required |
|---|---|---|
| SQOOP_TRG_PASSWORD | Indicates the password of the user of the Big Data server where SQOOP will run. | Yes |
| SQOOP_WORKDIR_HDFS | Indicates the SQOOP working directory in HDFS.<br>Example: `/user/ofsaa` | Yes |
| AUTH_SERVICE_URL | **Note:** The value is already set. Do not change the value. | Yes |
| BATCH_SERVICE_URL | **Note:** The value is already set. Do not change the value. | Yes |
| META_SERVICE_URL | **Note:** The value is already set. Do not change the value. | Yes |
| SESSION_SERVICE_URL | **Note:** The value is already set. Do not change the value. | Yes |
| PGX_SERVER_URL | **Note:** The value is already set. Do not change the value. | Yes |
| RSERVE_USERNAME | Indicates the RServe username.<br>Value: oml<br>**Note:** The value is already set. Do not change the value. | Yes, only if the ORE interpreter is to be used. |
| RSERVE_PASSWORD | Indicates the RServe password.<br>Value: password<br>**Note:** The value is already set. Do not change the value. | Yes, only if the ORE interpreter is to be used. |
| HTTP_PROXY | Indicates the proxy for the host where Studio is deployed. | No |
| HTTPS_PROXY | Indicates the proxy for the host where Studio is deployed. | No |
| REPO_CRAN_URL | Indicates the URL from where the R packages are obtained.<br>The format for the REPO_CRAN_URL is as follows:<br>https://cran.r-project.org/ | No |
| USERS_LIB_PATH | Indicates the path where R packages are installed.<br>Value: /usr/lib64/R/library<br>**Note:** The value is already set. Do not change the value. | Yes, only if the ORE interpreter is to be used. |
| RSERVE_CONF_PATH | Indicates the path where the `Rserve.conf` file is present.<br>Value: /var/ore-interpreter/rserve<br>**Note:** The value is already set. Do not change the value. | Yes, only if the ORE interpreter is to be used. |

## Installing ETL Services

Installing the ETL service will also install the Cross Language Name Matching services also.

To install ETL Services, follow these steps:

1. Place the Hadoop Cluster files in the `<Studio_Installation_Path>/configmaps/spark` path. For more information on the file structure, see Required File Structure.
2. Place the Kerberos files in the `<Studio_Installation_Path>/configmaps/batchservice/user/conf/` path. For more information on the file structure, see Required File Structure.
3. Place the following jars in the `<Studio_Installation_Path>/docker/user/batchservice/lib/` path:
   - `hive-exec-1.1.0-cdh5.13.0.jar`
   - `HiveJDBC4.jar`

- `hive-metastore-1.1.0-cdh5.13.0.jar`
- `hive-service-1.1.0-cdh5.13.0.jar`

---

**Note:** •The version of the jars are client/user-specific. These jars can be obtained from existing jars of Cloudera installation.

- The `HiveJDBC4.jar` file is not available in the Cloudera setup. You must download the same from the Cloudera website.

---

4. Configure the `config.sh` file in `<Studio_Installation_Path>/bin` path to replace the placeholder values as described in the following table:

---

**Note:** Do not alter the parameter values that are already set in the `config.sh` file

---

***Table 3–2 Configuring config.sh File***

| Parameter | Description |
|---|---|
| HIVE_SCHEMA | Indicates the Hive schema as configured in the `studio-env.yml` file. |
| ETL_DRIVER_CORES | Indicates the number of cores present on the ETL Initiation machine host (master of CDH server) that is present/accessible on the server where ETL services are deployed. |
| ETL_DRIVER_MEMORY | Indicates how much memory is to be assigned to the ETL service.<br>**Note**: Ensure that the memory is slightly less than that of the ETL Initiation machine host. For example: 90g. |
| ETL_DRIVER_HOST | Indicates the IP address of the machine where initiation of the ETL occurs, that is, Kubernetes master. |
| ETL_SERVICE_PORT | Indicates the port that should be used by the ETL service & Spark driver, as defined in the etl.yml Kubernetes deployment script, e.g., 30724.<br>This is used for the "spark.driver.port" Spark configuration. |
| ETL_DRIVER_ BLOCKMANAGER_PORT | The port that should be used by the Spark driver blockmanager, as defined in the etl.yml Kubernetes deployment script, e.g., 30726.<br>This is used for the "spark.driver.blockManager.port" Spark configuration. |
| ETL_EXECUTOR_ INSTANCES | The number of Spark executor instances that should be used on the big data cluster during ETL.<br>This is used for the "spark.executor.instances" Spark configuration. |
| ETL_EXECUTOR_CORES | The number of cores that should be used by each Spark executor instance on the big data cluster during ETL.<br>This is used for the "spark.executor.cores" Spark configuration. |
| ETL_EXECUTOR_MEMORY | The amount of memory that should be used by each Spark executor instance on the big data cluster during ETL.<br>This is used for the "spark.executor.memory" Spark configuration. |

*Table 3–2 Configuring config.sh File*

| Parameter | Description |
|---|---|
| URL_GLOBAL_GRAPH_NODES_CSV | Indicates the HDFS URL where the CSV file of the global graph is stored at the end of the ETL. It can either be a local or hdfs path.<br>For example: `hdfs:///user/ofsaa/STUDIO_ETL/global_graph_nodes.csv`<br>**Note**: Ensure you have already created the `ETL_Directory` manually and have provided 775 permission. This directory is used to store the CSV file at the end of the ETL. |
| URL_GLOBAL_GRAPH_EDGES_CSV | Indicates the HDFS URL where the CSV file of the global graph is stored at the end of the ETL. It can either be a local or hdfs path.<br>For example: `hdfs:///user/ofsaa/STUDIO_ETL/global_graph_edges.csv`<br>**Note**: Ensure this location is already created and available to store the CSV file at the end of the ETL. |
| URL_GLOBAL_GRAPH_CONFIG_JSON | Indicates the HDFS URL where the PGX graph configuration .json file is stored at the end of the ETL. The location can be either local or hdfs path.<br>For example: `hdfs:///user/ofsaa/STUDIO_ETL/config.json`<br>**Note**: Ensure this location is already created and available to store the JSON file at the end of the ETL.<br>If you do not want a graph configuration file written, provide the value as follows:<br>`null://EMPTY` |
| URL_NAMES_CSV | Indicates the HDFS URL where the names CSV file is updated at the end of the ETL. It can either be a local or hdfs path.<br>For example: `hdfs:///user/ofsaa/STUDIO_ETL/name_index.csv`<br>**Note**: Ensure this location is already created and available, and the CSV file is already created and placed in this location. The CSV file values are replaced with the values in the new CSV file created at the end of the ETL.<br>For information on creating the CSV files, see Appendix H, "Creating Required Index Files". |
| URL_ADDRESS_CSV | Indicates the HDFS URL where the addresses CSV file is updated at the end of the ETL. It can either be a local or hdfs path.<br>For example: `hdfs:///user/ofsaa/STUDIO_ETL/address_index.csv`<br>**Note**: Ensure this location is already created and available, and the CSV file is already created and placed in this location. The CSV file values are replaced with the values in the new CSV file created at the end of the ETL.<br>For information on creating the CSV files, see Appendix H, "Creating Required Index Files". |
| PGX_SERVER_NUM_REPLICAS | For example: 1 |
| PGX_GLOBAL_GRAPH_NAME | For example: GlobalGraphIH |

5. Grant Execute permission to the `<Studio_Installation_Path>/bin` by executing the following command:

```
chmod 755 install.sh config.sh
```

6. Run the following command to replace the configured placeholders in the ETL config maps:

   ```
   ./install.sh
   ```

   > **Note:** Execution of the `install.sh` command does not generate any log file.

   The values configured in the `config.sh` file will auto-populate the values in the following property files for ETL, Cross Language Name Matching, and PGX respectively:

   - `<Studio_Installation_Path>/configmaps/etl/etl.properties`
   - `<Studio_Installation_Path>/configmaps/crosslangnamematch/NameMatchingLocations.properties`
   - `<Studio_Installation_Path>/configmaps/pgx-server/pgx.conf`

   > **Note:** The values for the `<URL_GLOBAL_GRAPH_CONFIG_JSON> and <GlobalGraphIH>` parameters are auto-populated with the values configured in the `<Studio_Installation_Path>/bin/config.sh` file.

   For more information on the `etl.properties and NameMatchingLocations.properties` file, see the *Studio Services* chapter in the *OFS Crime and Compliance Studio Administration Guide*.

7. Navigate to `<Studio_Installation_Path>/configmaps/pgx-server/` and modify the `pgx.conf` file as follows:

   Comment the following preload graph section:

   ```
   <!--
   "preload_graphs": [
       {
         "path": "<URL_GLOBAL_GRAPH_CONFIG_JSON>",
         "name": "<GlobalGraphIH"
       }
     ]
   -->
   ```

8. (Optional) Add a new data source. For more information, see *Configuring Data Sources for Graph* chapter in the *OFS Crime and Compliance Studio Administration Guide*.

## Verifying Resource Allocation for Studio Services

The required resources must be allocated to the Studio services as per the architecture. For information on resource allocation, see Appendix F, "Resource Allocation for Studio Services"

## Deploying Studio on Kubernetes Cluster

To deploy Studio on the Kubernetes cluster, follow these steps:

1. Navigate to the `<Studio_Installation_Path>`.
2. Execute the following command:

```
./fcc-studio.sh --registry <registry URL>:<registry port>
```

---

**Note:** Refer to `./fcc-studio.sh -h` for usage instructions.

---

After successful completion of deployment, the script displays a URL that can be used to launch the FCC Studio Application. For more information, see Launching the FCC Studio Application.

3. Verify the deployment. See Verifying the Deployment.
4. If you have added new data sources at this stage, you must redeploy FCC Studio. For more information, see Chapter 7, "Redeploying Studio Application".

## Verifying the Deployment

To verify the deployment, follow these steps:

1. Wait for a minimum of 10 minutes, after completing the execution of the `./fcc-studio.sh` command, and run the following command:

   ```
   kubectl get pods -n fccs
   ```

   The pod details are displayed to indicate the status of the services. You can also check the logs of the Studio services from the Kubernetes Dashboard. For more information, see Appendix B, "Checking Logs of Studio Services".

---

**Note:**

- Ensure all the pods are ready before launching the Studio application.

---

## Launching the FCC Studio Application

To launch the FCC Studio application, follow these steps:

1. Enter the URL obtained after the successful deployment of the Studio application in the following format into the browser:

   ```
   https://<Master_Node>:30078
   ```

   The OFS Crime and Compliance Studio application is launched. You can check the logs of the Studio services from the Kubernetes Dashboard. For more information, see Appendix B, "Checking Logs of Studio Services".

# 4 Post-deployment Configuration for Studio with BD

On the successful deployment of the Studio application, perform the following post-deployment configuration:

- Configuring Interpreters
- Performing OFSAA Configuration for Batch Execution
- Performing Hive Data Movement

## Configuring Interpreters

After starting the Studio application, the configuration for the interpreters can be performed from the user interface (UI). For information on configuring interpreters, see the *Configuring Interpreters* chapter in the *Oracle Financial Services Crime and Compliance Studio Administration Guide*.

## Performing OFSAA Configuration for Batch Execution

To perform OFSAA configuration for batch execution, follow these steps:

1. Copy the files in the `<Studio_Installation_Path>/out/ficdb/bin` path to the server where the BD pack is installed and to the `$FIC_DB_HOME/bin` path of the OFSAA setup.
2. Execute the following command to grant Execute permission to the files:

   ```
   chmod +x <filenames>
   ```

3. Copy all the files in the `<Studio_Installation_Path>/out/ficdb/lib` path to the `$FIC_DB_HOME/lib` path.

   For information on running Studio Batches, see *Managing Studio Batches* chapter in the *OFS Crime and Compliance Studio Administration Guide*.

## Performing Hive Data Movement

To perform Hive data movement, follow these steps:

- Configuring Schema Creation
- Creating Credential Keystore
- Configuring ETL

## Configuring Schema Creation

- Configuring Schema Creation from Studio Server
- Configuring Schema Creation from OFSAA Server

### Configuring Schema Creation from Studio Server

To configure Schema creation from Studio server, follow these steps:

1. Set `FIC_DB_HOME` path to `<Studio_Installation_Path>/out/ficdb`.

   ---

   **Note:** The `$FIC_DB_HOME` path can be set from the `.profile` file as well.

   ---

2. Create a Hive Schema with the name mentioned in the `HIVE_SCHEMA` parameter in the `studio-env.yml` file.

   For information on `studio-env.yml` file, see Configuring studio-env.yml File.

3. Execute the following shell script in the `<Studio_Installation_Path>/out/ficdb/bin/` path to create tables in Hive Schema:

   ```
   FCCM_Studio_SchemaCreation.sh HIVE
   ```

   This creates tables in the Hive Schema.

4. Check Batch Service logs for more information.

### Configuring Schema Creation from OFSAA Server

To configure Schema creation from OFSAA server, follow these steps:

1. Copy all the jar files located in the `<Studio_Installation_Path>/out/ficdb/lib` path to the `<OFSAA_FIC_HOME_PATH>/ficdb/lib` path.
2. Copy all the .sh files located in the `<Studio_Installation_Path>/out/ficdb/bin` path to the `<OFSAA_FIC_HOME_PATH>/ficdb/bin` path.
3. Create a Hive Schema with the name mentioned in the `HIVE_SCHEMA` parameter in the `studio-env.yml` file.

   For information on `studio-env.yml` file, see Configuring studio-env.yml File.

4. Execute the following shell script in the `<OFSAA_FIC_HOME_PATH>/ficdb/bin` path to create tables in the Hive Schema:

   ```
   FCCM_Studio_SchemaCreation.sh HIVE
   ```

   This creates tables in the Hive Schema.

5. Check Batch Service logs for more information.

## Creating Credential Keystore

To create a credential keystore, follow these steps:

1. Login as HDFS SuperUser.
2. Create a credential keystore on HDFS by executing the following command:

   ```
   hadoop credential create mydb.password.alias -provider
   jceks://hdfs/user/root/oracle.password.jceks
   ```

3. Verify the credential keystore file by executing the following command:

```
hadoop credential list -provider
jceks://hdfs/user/root/oracle.password.jceks
```

4. Grant Read permission to the keystore file by executing the following command:

```
hadoop fs -chmod 744 /user/root/oracle.password.jceks
```

---

**Note:** Ensure the correct values of the credential keystore file path and the alias are provided in the `studio-env.yml` file.

---

## Configuring ETL

- Configuring Data Movement and Graph Load
- Configuring FILEPATH for ICIJ
- Configuring Pre-load Global Graph for PGX Server

### Configuring Data Movement and Graph Load

---

**Note:** The Big Data System Administrator must place the `batchservice-8.0.7.2.0.jar` file in all nodes of the Spark cluster. Ensure that the path of the jar file is present in the Spark `classpath` in the `spark-defaults.conf` file.

---

To configure the Data Movement and Graph Load, follow these steps:

1. Copy the required `FCCM_Studio_SqoopJob.sh` files from the `<Studio_Installation_Path>/out/ficdb/bin` path to the`<FIC_HOME of OFSAA_Installed_Path>/ficdb/bin` path.

   For information on performing Data Movement and Graph Load, see the *Data Movement and Graph Loading for Big Data Environment* section in the *OFS Crime and Compliance Studio Administration Guide*.

### Configuring FILEPATH for ICIJ

---

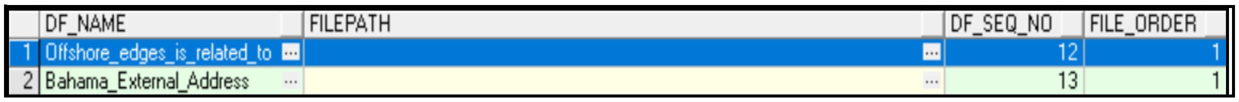
**Note:** The FCC Studio graph model is configured to include ICIJ watch list files.

---

To configure FILEPATH for ICIJ, follow these steps:

- If watch list files are present, follow these steps:
  1. Place the watch list file in HDFS, which is accessible by the user.
  2. Update the `FILEPATH` of the watch list files in the `fcc_studio_etl_files` table.

***Figure 4–1 fcc_studio_etl_files Table***



| | DF_NAME | FILEPATH | DF_SEQ_NO | FILE_ORDER |
|---|---|---|---|---|
| 1 | Offshore_edges_is_related_to | | 12 | 1 |
| 2 | Bahama_External_Address | | 13 | 1 |

- If watch list files are absent, follow these steps:

1. Edit the `<Studio_Installation_Path>/configmaps/etl/etl.properties` file as follows:

   Change the following:

   ```
   connectors=paradise;bahama;offshore;panama;fcdm
   ```

   to

   ```
   connectors=fcdm
   ```

   **Note:** Ignore the properties that start with parameter values like `bahama, offshore, paradise`, and `panama`.

### Configuring Pre-load Global Graph for PGX Server

To configure pre-load global graph for PGX server, follow these steps:

1. Navigate to `<Studio_Installation_Path>/configmaps/pgx-server/` and modify the `pgx.conf` file as follows:

   **Note:** The values for the `<URL_GLOBAL_GRAPH_CONFIG_JSON> and <GlobalGraphIH>` parameters are auto-populated with the values configured in the `<Studio_Installation_Path>/bin/config.sh` file.

   Uncomment the following preload graph section in the `pgx.conf` file:

   ```
   "preload_graphs": [
       {
         "path": "<URL_GLOBAL_GRAPH_CONFIG_JSON>",
         "name": "<GlobalGraphIH>"
       }
     ]
   ```

2. Execute the following command to delete namespace of FCC Studio:

   ```
   kubectl delete namespace fccs
   ```

3. Navigate to `<Studio_Installation_Path>/secrets` folder and re-enter the values of sensitive information in the `studio-env.yml` file. For example, `STUDIO_DB_PASSWORD`, `HADOOP_PASSWORD-ALIAS` and so on.
4. Redeploy the Studio application. For more information, see Deploying Studio on Kubernetes Cluster.
5. Check the Kubernetes Dashboard, Appendix B, "Checking Logs of Studio Services", to ensure pgx-server service is up and running.

## Performing Configuration to Run Scenario Notebooks

To perform the configuration required to run scenario notebooks, follow these steps:

1. Copy the required `FCCM_Studio_NotebookExecution.sh` file from the `<Studio_Installation_Path>/out/ficdb/bin` path to the `<FIC_HOME of OFSAA_Installed_Path>/ficdb/bin` path.

For information on performing Data Movement and Graph Load, see the *Executing Published Scenario Notebook* section in the *OFS Crime and Compliance Studio Administration Guide*.

5

# Deploying Studio without BD

This chapter provides the necessary information to deploy a new instance of the Studio application without BD on the Kubernetes cluster. It includes the following sections:

1. Deploying Studio without BD
2. Verifying the Deployment
3. Launching the FCC Studio Application

## Deploying Studio without BD

To deploy Studio, follow these steps:

1. Configuring Wallet
2. Configuring studio-env.yml File
3. Deploying Studio on Kubernetes Cluster

### Configuring Wallet

To configure wallets, follow these steps:

1. Create a wallet. For information on creating wallets, see Appendix D, "Setting Up Password Stores with Oracle Wallet".
2. Copy the wallet files, `cwallet.sso, ewallet.p12` and `tnsnames.ora`, and place in the `<Studio_Installation_Path>/configmaps/wallet` path.

### Configuring studio-env.yml File

To configure the `studio-env.yml` file, follow these steps:

1. Login to the server as a non-root user.
2. Navigate to the `<Studio_Installation_Path>/secrets/studio-env.yml` file.
3. Configure the `studio-env.yml` file as mentioned in Table 5–1.

**Note:**

- You must manually set the Interaction Variable parameter values. If a value is not applicable, enter NA and ensure that the value is not entered as NULL.
- Do not alter the parameter values that are already set in the `studio-env.yml` file

*Table 5–1 studio-env.yml Parameters*

| InteractionVariable Name | Significance | Required |
|---|---|---|
| apiVersion | For example: v1 | Yes |
| kind | For example: Secret | Yes |
| **metadata** | | |
| name | For example: studio-env | Yes |
| **stringData** | | |
| REALM | For example: `com.oracle.ofss.fccm.studio.datastudio.auth.DemoRealm` | Yes |
| NON_OFSAA | Enter "true" to deploy Studio application without BD on the Kubernetes cluster. | Yes |
| OFSAA_SERVICE_URL | Indicates the URL of the OFSAA instance. Do not enter '/' at the end of the URL.<br>**Note:**<br>• For OFSAAAI, the value must be in the following format:<br>`https://<HostName>:<PortNo>/<ContextName>/rest-api` | Yes |
| LIVY_HOST_URL | Indicates the URL of the Livy application.<br>The format for the URL is as follows:<br>`http://<HostName>:<PortNo>` | Yes, only if the Spark-sql, Spark-scala and/or PySpark interpreters are to be used. |
| STUDIO_DB_ HOSTNAME | Indicates the hostname of the database where Studio schema is created. | Yes |
| STUDIO_DB_PORT | Indicates the port number where Studio schema is created. | Yes |
| STUDIO_DB_SERVICE_ NAME | Indicates the service name of the database where Studio schema is created. | Yes |
| STUDIO_DB_SID | Indicates the SID of the database where Studio schema is created. | Yes |
| STUDIO_DB_ USERNAME | Indicates the username of the Studio Schema (newly created Oracle Schema). | Yes |
| STUDIO_DB_ PASSWORD | Indicates the password for the newly created schema. | Yes |
| STUDIO_ALIAS_NAME | Indicates the Studio alias name. For more information, see Appendix D, "Setting Up Password Stores with Oracle Wallet".<br>**Note**: Enter the alias name that was created during wallet creation. | Yes |
| STUDIO_WALLET_ LOCATION | Indicates the location of the wallet created for the Studio Schema. | Yes |
| STUDIO_TNS_ ADMIN_PATH | Indicates the Studio TNS admin path. | Yes |
| BD_CONFIG_ HOSTNAME | Indicates the hostname of the database where BD config schema is installed.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_CONFIG_PORT | Indicates the port of the database where BD config schema is installed.<br>**Note:** Ensure to provide the value as NA. | Yes |

*Table 5–1 (Cont.) studio-env.yml Parameters*

| InteractionVariable Name | Significance | Required |
|---|---|---|
| BD_CONFIG_SERVICE_NAME | Indicates the service name of the database where BD config schema is installed.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_CONFIG_SID | Indicates the SID of the database where BD config schema is installed.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_CONFIG_USERNAME | Indicates the username for the BD config schema.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_CONFIG_PASSWORD | Indicates the password for the BD config schema.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_CONFIG_ALIAS_NAME | Indicates the BD config alias name. For more information, see Appendix D, "Setting Up Password Stores with Oracle Wallet".<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_CONFIG_WALLET_LOCATION | **Note:** Ensure to provide the value as NA. | Yes |
| BD_CONFIG_TNS_ADMIN_PATH | **Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_HOSTNAME | Indicates the BD atomic schema hostname.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_PORT | Indicates the BD atomic schema port number.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_SERVICE_NAME | Indicates the BD atomic schema service name.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_SID | Indicates the BD atomic schema SID.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_USERNAME | Indicates the username of the BD atomic schema.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_PASSWORD | Indicates the password of the BD atomic schema.<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_ALIAS_NAME | Indicates the BD atomic alias name. For more information, see Appendix D, "Setting Up Password Stores with Oracle Wallet".<br>**Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_WALLET_LOCATION | **Note:** Ensure to provide the value as NA. | Yes |
| BD_ATOMIC_TNS_ADMIN_PATH | **Note:** Ensure to provide the value as NA. | Yes |
| FSINFODOM | Indicates the name of the OFSAA or BD Infodom.<br>**Note:** Ensure to provide the value as NA. | Yes |
| FSSEGMENT | Indicates the name of the OFSAA or BD segment.<br>**Note:** Ensure to provide the value as NA. | Yes |

***Table 5–1 (Cont.) studio-env.yml Parameters***

| InteractionVariable Name | Significance | Required |
|---|---|---|
| DATAMOVEMENT_LINK_NAME | • If the newly created schema is in a different database host, you must create a DB link and provide the same link in this parameter. Alternatively, you can provide the source schema name.<br>If no DB link is present, provide the <SCHEMA_NAME> in this parameter.<br>• If the newly created schema is in the same database host, the value for this parameter is the user name of the BD atomic schema.<br>**Note:** Ensure to provide the value as NA. | Yes |
| DATAMOVEMENT_LINK_TYPE | If the DB link is used, enter DBLINK in this field. If the DB link is not used, enter SCHEMA in this field.<br>**Note:** Ensure to provide the value as NA. | Yes |
| HADOOP_CREDENTIAL_PROVIDER_PATH | Indicates the path where Hadoop credential is stored.<br>**Note:** Ensure to provide the value as NA. | Yes |
| HADOOP_PASSWORD_ALIAS | Indicates the Hadoop alias given when creating the Hadoop credentials.<br>**Note**: Enter the alias name that was created during wallet creation.<br>For information on how to create credential keystore, see Creating Credential Keystore<br>**Note:** Ensure to provide the value as NA. | Yes |
| Hive_Host_Name | Indicates the Hive hostname.<br>**Note:** Ensure to provide the value as NA. | Yes |
| Hive_Port_number | Indicates the Hive port number.<br>Contact System Administrator to obtain the port number.<br>**Note:** Ensure to provide the value as NA. | Yes |
| HIVE_PRINCIPAL | Indicates the Hive Principal.<br>Contact System Administrator to obtain HIVE_PRINCIPAL.<br>**Note:** Ensure to provide the value as NA. | Yes |
| HIVE_SCHEMA | Indicates the new Hive schema name.<br>**Note:** Ensure to provide the value as NA. | Yes |
| JAAS_CONF_FILE_PATH | Created for future use. | No |
| Krb_Host_FQDN_Name | Indicates the Kerberos host FQDN name.<br>**Note:** Ensure to provide the value as NA. | Yes |
| Krb_Realm_Name | Indicates the Kerberos realm name.<br>**Note:** Ensure to provide the value as NA. | Yes |
| Krb_Service_Name | Indicates the Kerberos service name.<br>**Note:** Ensure to provide the value as NA.<br>Example: Hive | Yes |
| KRB5_CONF_FILE_PATH | Created for future use. | No |
| security_krb5_kdc_server | Created for future use. | No |
| security_krb5_realm | Created for future use. | No |

***Table 5–1 (Cont.) studio-env.yml Parameters***

| InteractionVariable Name | Significance | Required |
|---|---|---|
| server_kerberos_keytab_file | Created for future use.<br>**Note:** Ensure to provide the value as NA. | Yes |
| server_kerberos_principal | Created for future use.<br>**Note:** Ensure to provide the value as NA. | Yes |
| SQOOP_HOSTMACHINE_USER_NAME | Indicates the user name of the Big Data server where SQOOP will run.<br>**Note:** Ensure to provide the value as NA. | Yes |
| SQOOP_PARAMFILE_PATH | **1.** Create a file with the name `sqoop.properties` in the Big Data server and add the following entry to the same:<br>`oracle.jdbc.mapDateToTimestamp=false`<br>**2.** Enter the location of the `sqoop.properties` file in the `SQOOP_PARAMFILE_PATH` parameter.<br>Example: `/scratch/ofsaa/`<br>**Note**: Ensure that the location name ends with a '/'.<br>**Note:** Ensure to provide the value as NA. | Yes |
| SQOOP_PARTITION_COL | Indicates the column in which the HIVE table is partitioned.<br>The value must be `SNAPSHOT_DT`<br>**Note:** Ensure to provide the value as NA. | Yes |
| SQOOP_TRG_HOSTNAME | Indicates the hostname of the Big Data server where SQOOP will run.<br>Example: `<HostName>`<br>**Note:** Ensure to provide the value as NA. | Yes |
| SQOOP_TRG_PASSWORD | Indicates the password of the user of the Big Data server where SQOOP will run. | Yes |
| SQOOP_WORKDIR_HDFS | Indicates the SQOOP working directory in HDFS.<br>Example: `/user/ofsaa` | Yes |
| AUTH_SERVICE_URL | **Note:** The value is already set. Do not change the value. | Yes |
| BATCH_SERVICE_URL | **Note:** The value is already set. Do not change the value. | Yes |
| META_SERVICE_URL | **Note:** The value is already set. Do not change the value. | Yes |
| SESSION_SERVICE_URL | **Note:** The value is already set. Do not change the value. | Yes |
| PGX_SERVER_URL | **Note:** The value is already set. Do not change the value. | Yes |
| RSERVE_USERNAME | Indicates the RServe username.<br>Value: oml<br>**Note:** The value is already set. Do not change the value. | Yes, only if the ORE interpreter is to be used. |
| RSERVE_PASSWORD | Indicates the RServe password.<br>Value: password<br>**Note:** The value is already set. Do not change the value. | Yes, only if the ORE interpreter is to be used. |
| HTTP_PROXY | Indicates the proxy for the host where Studio is deployed. | No |
| HTTPS_PROXY | Indicates the proxy for the host where Studio is deployed. | No |

*Table 5–1 (Cont.) studio-env.yml Parameters*

| InteractionVariable Name | Significance | Required |
|---|---|---|
| REPO_CRAN_URL | Indicates the URL from where the R packages are obtained.<br>The format for the REPO_CRAN_URL is as follows:<br>https://cran.r-project.org/ | No |
| USERS_LIB_PATH | Indicates the path where R packages are installed.<br>Value: /usr/lib64/R/library<br>**Note:** The value is already set. Do not change the value. | Yes, only if the ORE interpreter is to be used. |
| RSERVE_CONF_PATH | Indicates the path where the `Rserve.conf` file is present.<br>Value: /var/ore-interpreter/rserve<br>**Note:** The value is already set. Do not change the value. | Yes, only if the ORE interpreter is to be used. |

## Verifying Resource Allocation for Studio Services

The required resources must be allocated to the Studio services as per the architecture. For information on resource allocation, see Appendix F, "Resource Allocation for Studio Services"

## Deploying Studio on Kubernetes Cluster

To deploy Studio on the Kubernetes cluster, follow these steps:

1. Navigate to the `<Studio_Installation_Path>`.
2. Execute the following command:

   ```
   ./fcc-studio.sh --registry <registry URL>:<registry port>
   ```

   **Note:** Refer to `./fcc-studio.sh -h` for usage instructions.

   After successful completion of deployment, the script displays a URL that can be used to launch the FCC Studio Application. For more information, see Launching the FCC Studio Application.
3. Verify the deployment. See Verifying the Deployment.
4. If you have added new data sources at this stage, you must redeploy FCC Studio. For more information, see Chapter 7, "Redeploying Studio Application".

## Verifying the Deployment

To verify the deployment, follow these steps:

1. Wait for a minimum of 10 minutes, after completing the execution of the `./fcc-studio.sh` command, and run the following command:

   ```
   kubectl get pods -n fccs
   ```

   The pod details are displayed to indicate the status of the services. You can also check the logs of the Studio services from the Kubernetes Dashboard. For more information, see Appendix B, "Checking Logs of Studio Services".

**Note:** Ensure the metaservice is up and running before launching the Studio application.

## Launching the FCC Studio Application

To launch the FCC Studio application, follow these steps:

1. Enter the URL obtained after the successful deployment of the Studio application in the following format into the browser:

   ```
   https://<Master_Node>:30078
   ```

   The OFS Crime and Compliance Studio application is launched. You can check the logs of the Studio services from the Kubernetes Dashboard. For more information, see Appendix B, "Checking Logs of Studio Services".

# 6 Post-deployment Configuration for Studio without BD

On the successful deployment of the Studio application, perform the following post-deployment configuration:

## Configuring Interpreters

After starting the Studio application, the configuration for the interpreters can be performed from the user interface (UI). For information on configuring interpreters, see the *Configuring Interpreters* chapter in the *Oracle Financial Services Crime and Compliance Studio Administration Guide*.

# 7 Redeploying Studio Application

If the deployment of the Studio application is unsuccessful, you must redeploy the application after performing the required cleanup tasks.

To redeploy the Studio application, follow these steps:

1. Execute the following command to delete namespace of FCC Studio:

   ```
   kubectl delete namespace fccs
   ```

2. Navigate to the `<Studio_Installation_Path>` path and correct the parameters or files as suggested by the error.
3. Navigate to `<Studio_Installation_Path>/secrets` folder and re-enter the values of sensitive information in the `studio-env.yml` file. For example, `STUDIO_DB_PASSWORD`, `HADOOP_PASSWORD-ALIAS` and so on.
4. Perform database cleanup by performing the following:

***Table 7–1***

| Schema | Applicable for BD | Applicable for non-BD |
|---|---|---|
| Cleanup for Studio Schema | Yes | Yes |
| Cleanup for BD Atomic Schema | Yes | No |
| Cleanup for BD Config Schema | Yes | No |

5. Redeploy the Studio application. For more information, see Deploying Studio on Kubernetes Cluster.

## Cleanup for Studio Schema

To cleanup the Studio schema, follow these steps:

1. Drop the existing Studio schema and create a new Studio schema.

   **Note:**

   The username and password credentials of the Studio Schema in the wallet files must be updated accordingly. (If applicable)

2. Grant the following permissions to the newly created Oracle Database Schema:
   - `GRANT create session TO <Schema User>;`
   - `GRANT create table TO <Schema User>;`

- `GRANT create view TO <Schema User>;`
- `GRANT create any trigger TO <Schema User>;`
- `GRANT create any procedure TO <Schema User>;`
- `GRANT create sequence TO <Schema User>;`
- `GRANT execute on dbms_rls TO <Schema User>;`
- `GRANT execute on sys.dbms_session TO <Schema User>;`
- `ALTER USER <Schema User> QUOTA 100M ON users;`
- `GRANT create sequence TO <Schema User>;`
- `GRANT create SYNONYM TO <Schema User>;`
- `GRANT ALL privileges TO <Studio Schema User>;`

**Note:**

If dropping the schema is not an option, drop the tables and sequences as mentioned in the Studio Schema Tables section.

## Cleanup for BD Atomic Schema

To cleanup the BD Atomic Schema, follow these steps:

1. Login to the BD Atomic Schema.
2. Truncate the DATABASECHANGELOG and DATABASECHANGELOGLOCK tables using the following command:

   ```
   TRUNCATE TABLE DATABASECHANGELOGLOCK;
   
   TRUNCATE TABLE DATABASECHANGELOG;
   ```

## Cleanup for BD Config Schema

To cleanup the BD Config schema:

1. Login to the BD Config Schema.
2. Truncate the DATABASECHANGELOG and DATABASECHANGELOGLOCK tables using the following command:

   ```
   TRUNCATE TABLE DATABASECHANGELOGLOCK;
   
   TRUNCATE TABLE DATABASECHANGELOG;
   ```

# A

# Starting/Stopping Studio Services

This section describes how to start and stop the services needed for the Studio application.

- Starting a Service
- Stopping a Service
- Starting Studio Application

## Starting a Service

1. Navigate to the `<Studio_Installation_Path>/deploy.sh` file

   The `deploy.sh` file contains details of the commands to start Studio services.

2. Navigate to the `<Studio_Installation_Path>/out` directory.
3. Run the start command in the console to start a service:

   For example:

   ```
   kubectl -n fccs apply -f  deployments/etl.yml
   ```

- You can check the logs of the Studio services from the Kubernetes Dashboard. For more information, see Appendix B, "Checking Logs of Studio Services"

## Stopping a Service

1. Navigate to the `<Studio_Installation_Path>/out` directory.
2. Run the stop command in the console to stop a service:

   For example:

   ```
   kubectl -n fccs delete -f  deployments/etl.yml
   ```

## Starting Studio Application

To start the Studio application:

1. Navigate to the `<Studio_Installation_Path>/bin/` path.
2. Run the following command:

   ```
   ./fcc-studio.sh --registry <registry URL>:<registry port>
   ```

   The Studio application is restarted.

   Check the logs of the Studio services from the Kubernetes Dashboard. For more information, see Appendix B, "Checking Logs of Studio Services"

Once all the services are up and running, the Studio application can be accessed using the following URL:

```
http://<HostName>:30078
```

# B

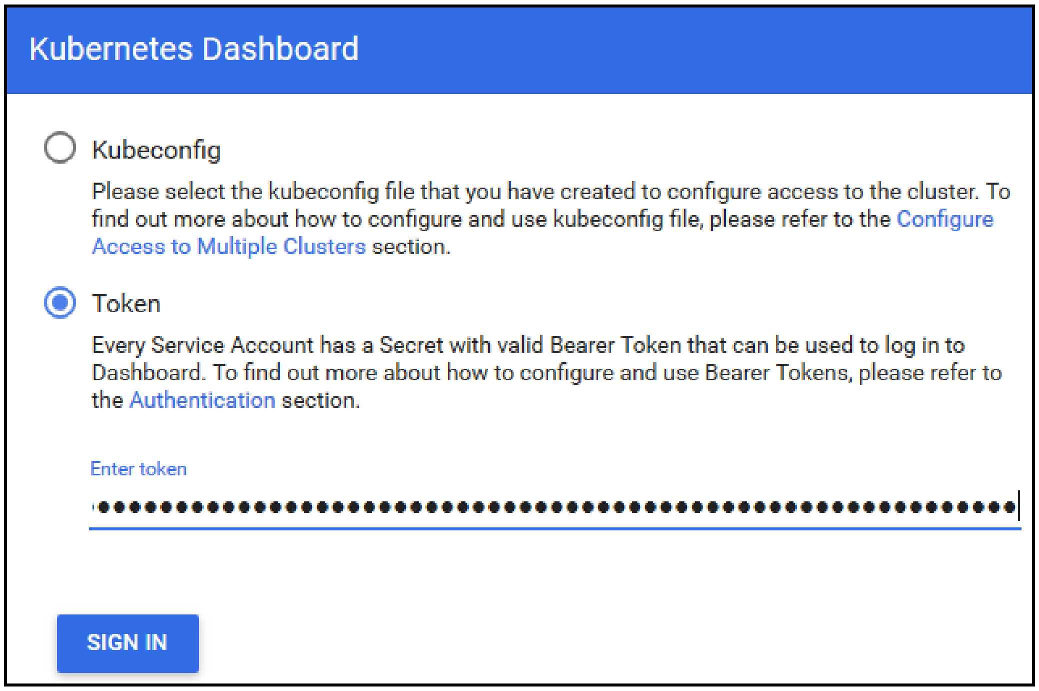
# Checking Logs of Studio Services

The Kubernetes UI enables you to view the logs of the services installed as part of the Studio application.

To check the logs of the Studio services from the Kubernetes Dashboard, follow these steps:

1. Access the Kubernetes Dashboard.

   The **Kubernetes Dashboard** login page is displayed.

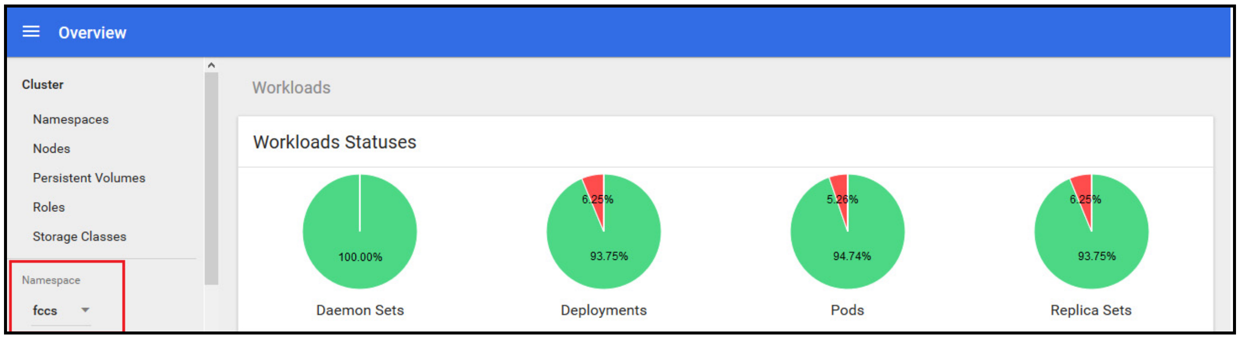


Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the Configure Access to Multiple Clusters section.

Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the Authentication section.

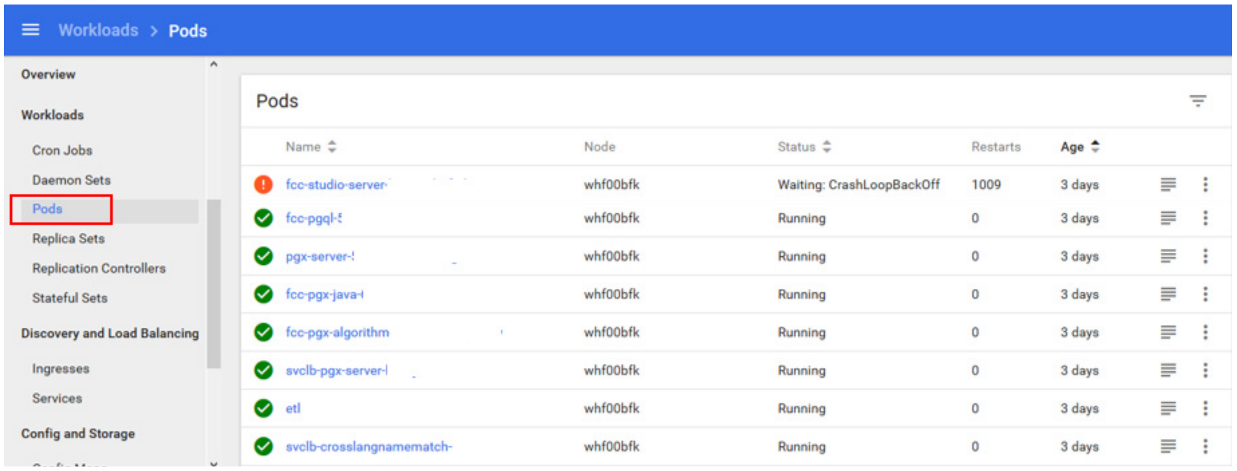
2. Select **Token** and enter the Admin user secret token in the **Enter Token** field.
3. Click **SIGN IN**.

   The Kubernetes Dashboard page is displayed.
4. Select **fccs** from the **Namespace** drop-down list on the menu items displayed on the LHS.



5. Navigate to **Workloads** > **Pods** from the menu items displayed on the LHS.

   The **Pods** page is displayed with the details of all the services installed as part of the Studio installation.



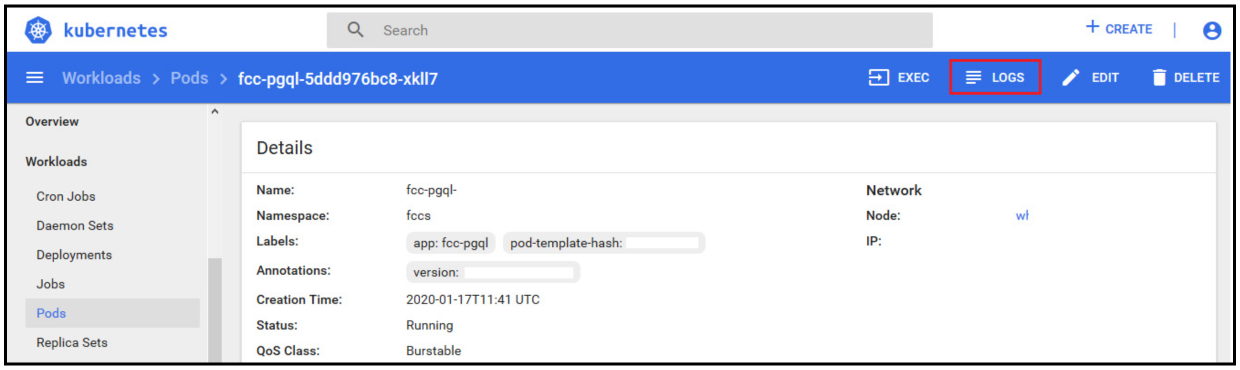
Pods

| Name | Node | Status | Restarts | Age |
|---|---|---|---|---|
| fcc-studio-server- | whf00bfk | Waiting: CrashLoopBackOff | 1009 | 3 days |
| fcc-pgql-5 | whf00bfk | Running | 0 | 3 days |
| pgx-server-! | whf00bfk | Running | 0 | 3 days |
| fcc-pgx-java-( | whf00bfk | Running | 0 | 3 days |
| fcc-pgx-algorithm | whf00bfk | Running | 0 | 3 days |
| svclb-pgx-server-l | whf00bfk | Running | 0 | 3 days |
| etl | whf00bfk | Running | 0 | 3 days |
| svclb-crosslangnamematch- | whf00bfk | Running | 0 | 3 days |

6. Click the service name from the **Name** column.

   The service details are displayed.

7. Click the **Logs** tab.



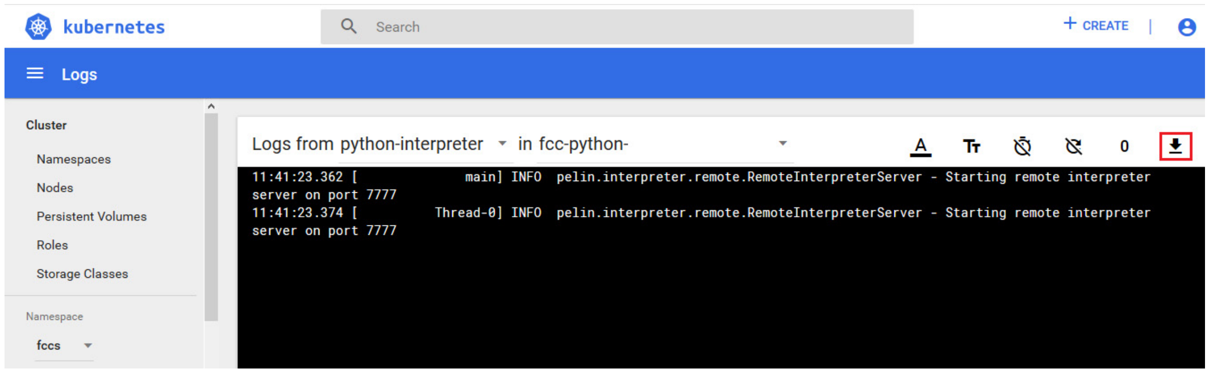
Details

| | |
|---|---|
| Name: | fcc-pgql- |
| Namespace: | fccs |
| Labels: | app: fcc-pgql   pod-template-hash: |
| Annotations: | version: |
| Creation Time: | 2020-01-17T11:41 UTC |
| Status: | Running |
| QoS Class: | Burstable |

Network

| | |
|---|---|
| Node: | wl |
| IP: | |

The service logs are displayed and you can download the logs by clicking **Download Logs** icon.



```
11:41:23.362 [            main] INFO  pelin.interpreter.remote.RemoteInterpreterServer - Starting remote interpreter
server on port 7777
11:41:23.374 [        Thread-0] INFO  pelin.interpreter.remote.RemoteInterpreterServer - Starting remote interpreter
server on port 7777
```

# C Tables and Sequences

The list of tables and sequences that are to be dropped during redeployment of the Studio application are as follows:

- Studio Schema Tables
- Studio Schema Sequences

## Studio Schema Tables

The following table includes the details of the Studio Schema tables that must be dropped during the redeployment of the Studio application:

***Table 7–2 Studio Schema Tables***

| | | |
|---|---|---|
| DS_PARAGRAPH | DS_NOTEBOOK_TAGS | DS_TASK_RESULTS |
| DS_ENTITY_PERMISSIONS | DS_ROLE | DS_PERMISSION_ACTIONS |
| DS_GROUP | DS_IS_PERMITTED | DS_PERMISSION_MAPPING |
| DS_USER_PERMS_MAP | DS_USER_ROLES | DS_NOTEBOOK |
| DS_INTERPRETER_RESULT_MSGS | DS_USER | DS_PERMS_MAP_ACTIONS |
| DS_ENTITY_PERMS_MAP | DS_TASK | DS_GRAPH |
| DS_INTERPRETER_RESULT | DS_GROUP_PERMS_MAP | DS_NOTEBOOK_RELATIONS |
| DS_INTERPRETER_PROPS | DS_JOB | DS_PERMISSION |
| DS_ROLE_PERMS_MAP | DS_VISUALIZATION_TEMPLATE | DS_RESULT_MESSAGE |
| DS_INTERPRETER_ABILITIES | DATABASECHANGELOG | DATABASECHANGELOGLOCK |
| DS_USER_GROUPS | DS_INTERPRETER_VARIANT | DS_COMMENT |
| DS_PARAGRAPH_RELATIONS | | |

## Studio Schema Sequences

The following table includes the details of the Studio Schema sequences that must be dropped during the redeployment of the Studio application:

***Table 7–3 Studio Schema Sequences***

| | | |
|---|---|---|
| SEQ_COMMENT | SEQ_ENTITY_PERMISSIONS | SEQ_GRAPH |

*Table 7–3 Studio Schema Sequences*

| | | |
|---|---|---|
| SEQ_GROUP | SEQ_INTERPRETER_RESULT | SEQ_INTERPRETER_VARIANT |
| SEQ_JOB | SEQ_NOTEBOOK | SEQ_PARAGRAPH |
| SEQ_PERMISSION | SEQ_PERMISSION_MAPPING | SEQ_RESULT_MESSAGE |
| SEQ_ROLE | SEQ_TASK | SEQ_USER |
| SEQ_VISUALIZATION_TEMPLATE | | |

# D

# Setting Up Password Stores with Oracle Wallet

This section includes the following topics:

## Overview

As part of an application installation, administrators must set up password stores for database user accounts using Oracle Wallet. These password stores must be installed on the application database side. The installer handles much of this process, the administrators must perform some additional steps.

A password store for the application and application server user accounts must also be installed; however, the installer takes care of this entire process.

## Setting Up Password Stores for Database User Accounts

After the database is installed and the default database user accounts are set up, administrators must set up a password store using the Oracle Wallet. This involves assigning an alias for the username and associated password for each database user account. The alias is used later during the application installation. This password store must be created on the system where the application server and database client are installed.

This section describes the steps to set up a wallet and the aliases for the database user accounts. For more information on configuring authentication and password stores, refer to the Oracle Database Security Guide.

> **Note:** In this section, `<wallet_location>` is a placeholder text for illustration purposes. Before running the command, ensure that you have already created the `<wallet_location>` folder where you want to create and store the wallet.

To set up a password store for the database user accounts, follow these steps:

1. Login to the server as a Linux user.
2. Create a wallet in the `<wallet_location>` using the following command:

   ```
   mkstore -wrl  -create
   ```

After you run the command, a prompt appears. Enter a password for the Oracle Wallet in the prompt.

> **Note:** The mkstore utility is included in the Oracle Database Client installation.

The wallet is created with the auto-login feature enabled. This feature enables the database client to access the wallet contents without using the password. For more information, refer to the Oracle Database Advanced Security Administrator's Guide.

3. Create the database connection credentials in the wallet using the following command:

   ```
   mkstore -wrl <wallet_location> -createCredential <alias-name>
   <database-user-name>
   ```

   Run the above command for the following <alias-name>:

*Table 7–4*

| Schema | Applicable for BD | Applicable for non-BD |
|---|---|---|
| BD_Config_Schema | Yes | No |
| BD_Atomic_Schema | Yes | No |
| Studio_Schema | Yes | Yes |

After you run the command, a prompt appears. Enter the password associated with the database user account in the prompt. You are prompted to re-enter the password. Then you are prompted for the wallet password used in Step 1.

4. Repeat step 2 for all the database user accounts.
5. Update the `tnsnames.ora` file to include the following entry for each alias name to be set up.

   ```
   <alias-name> =

   (DESCRIPTION =

   (ADDRESS_LIST =

   (ADDRESS = (PROTOCOL = TCP) (HOST = <host>) (PORT = <port>))

   )

   (CONNECT_DATA =

   (SERVICE_NAME = <service>)

   )

   )
   ```

> **Note:**
>
> - You can either update the existing `tnsnames.ora` file with the above details or create a new `tnsnames.ora` file and make required entries.
> - `<alias-name>` is a user-defined value.

## Verifying the Connectivity of the Wallet

To verify the connectivity of the wallet, follow these steps:

1. Create a `sqlnet.ora` in the wallet directory using the following content:

   ```
   WALLET_LOCATION = (SOURCE = (METHOD = FILE) (METHOD_DATA = (DIRECTORY =
   <Wallet_Location>)) )
   ```

   ```
   SQLNET.WALLET_OVERRIDE=TRUE
   ```

   ```
   SSL_CLIENT_AUTHENTICATION=FALSE
   ```

2. Test the connectivity using the following command:

   > **Note:** The `ORACLE_HOME` used with the wallet must be the same version or higher than what the wallet was created with.

   ```
   $ export WALLET_LOCATION=<wallet_location>
   ```

   `$ export TNS_ADMIN=<tnsnames.ora_location>,` Here ensure to use the wallet to point to the alternate `tnsnames.ora` as created above.

   ```
   $ sqlplus /@<alias_name>
   ```

   The output is similar to:

   ```
   SQL*Plus: Release 11
   ```

   ```
   Connected to:
   ```

   ```
   Oracle Database 12c
   ```

   To verify if you are connected to the correct user:

   ```
   SQL> show user
   ```

   The output is similar to:

   ```
   USER is "<database-user-name>"
   ```

# E

# Installing RServe Manually

## Overview

You must install RServe manually on a host to expose the local R installation on that host to the network, so that remote RServe clients such as the R interpreter can use the local R installation.

The R interpreter always connects to an RServe instance and runs the R code remotely. The interpreter needs to be configured with the hostname or IP and the port of the remote instance (where RServe is running). When the interpreter is initialized, it connects to the remote instance.

This section includes the following topics:

- Prerequisites
- Installing RServe
- Configuring RServe
- Starting RServe
- Adding the Certificate to the Keystore
- Installing Additional Libraries

## Prerequisites

The following is a list of prerequisites required before beginning the installation of RServe:

- Ensure that Oracle Linux 7.x and Oracle JDK 8 are validated against Oracle Linux 7.4 and Oracle JDK 8u161.
- The user must be a root user.
- 800 MB disk space is required for package installation.

The following subsections provide more details for prerequisites:

- Installing Oracle R Distribution
- Installing Dependencies
- Installing ORE Client

### Installing Oracle R Distribution

To install the Oracle R Distribution (ORD), enable the **addons** and **optional_latest** channels in yum as shown in the following:

~~~
```bash
~~~

```
(root)# yum-config-manager --enable ol7_addons

(root)# yum-config-manager --enable ol7_optional_latest
```

After completing the previous step, pull ORD from the yum repository using the following command:

```bash
(root)# yum install R.x86_64 R-core-extra
```

To install ORD, see https://docs.oracle.com/cd/E83411_01/OREAD/installing-R-for-ORE.htm#OREAD129.

## Installing Dependencies

RServe has certain dependencies to run correctly. The **openssl-devel** is required for SSL support. The dependencies change based on the libraries you have installed. For example, to let **knitr** send plots as **base64 encoded** strings, you require **pango-devel**.

The following dependent packages must be installed for RServe to support SSL:

```bash
(root)# yum install openssl openssl-devel pango-devel
```

## Installing ORE Client

To connect to ORE through RServe, follow these steps:

- Install the corresponding client libraries. For more information, see https://docs.oracle.com/cd/E83411_01/OREAD/installing-ORE-client.htm#OREAD167 from the ORE project.
- Install the knitR and PrintR packages.

# Installing RServe

To install RServe, call the following code in your R shell:

```R
> install.packages('Rserve', repos='https://www.rforge.net/')
```

If you are behind a proxy, ensure that R is communicated about it when you start the R shell.

For example, you can start R shell as shown in the following before installing any package:

```bash
$ http_proxy=http://your-proxy:80 R
```

## Configuring RServe

You can base your config on the following example configuration, which you should store in an `Rserve.conf` file. You will require `Rserve.conf` as a reference when you start RServe.

~~~
```

auth required

plaintext disabled

pwdfile /path/to/Rserve.pwd

remote enable

switch.qap.tls enable

tls.port 6311

qap disable

interactive no

rsa.key /path/to/server.key

tls.key /path/to/server.key

tls.cert /path/to/server.crt
~~~

This configuration tells RServe to encrypt the communication with TLS and listen for incoming connections on port **6311**. The `Rserve.pwd` file appears as shown in the following example:

~~~
```

user $5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8

```
~~~

The file contains one line per user, where the first part is the username and the second part is the password. The password can either be plain text or an **MD5/SHA1** hash. In this example, the password, `password` is hashed with SHA1. If you use hashed passwords, the password must start with a `$` sign.

The `rsa.key, tls.ke,` and `tls.cert` settings point to the private key files you require for TLS. These keys can be generated using the **openssl** command-line tool as shown in the following example:

~~~
```bash

$ openssl genrsa -out server.key 2048

$ openssl req -new -key server.key -out server.csr

$ openssl x509 -req -days 265 -in server.csr -signkey server.key -out
server.crt

```
~~~

The preceding sample is an example and for a production deployment, you should use relevant certificates. You can find more information about configuration options on the RServe homepage - https://www.rforge.net/Rserve/doc.html.

## Starting RServe

After installing RServe and placing configuration files in the correct location, start the RServe as given here:

~~~
```bash
~~~

```
$ R CMD Rserve --no-save --RS-conf /path/to/Rserve.conf
```

RServe starts in the background. After it starts, the R Interpreter is able to connect to it. The RServe process is running before you start the R Interpreter.

## Adding the Certificate to the Keystore

The certificates that were generated in the previous step to configure RServe to encrypt the communication must be added to the Java keystore in order to be used by the R interpreter. The add procedure depends on your setup.

Add the certificate to a keystore as given here:

~~~
```bash

$ $JAVA_HOME/bin/keytool -import -alias rserve -file /path/to/server.crt
-keystore /path/to/keystore-storepass storepassword -noprompt

```
~~~

The certificate must be imported correctly and the correct keystore is used by the Java process you use to start the R interpreter. Else, you will get SSL related exceptions when the interpreter attempts to connect to RServe.

You can specify the keystore when starting the R interpreter as shown in the following example:

~~~
```bash

$ $JAVA_HOME/bin/java -Djavax.net.ssl.trustStore=/path/to/keystore -

Djavax.net.ssl.trustStorePassword=storepassword <additional paramters>

```
~~~

## Installing Additional Libraries

Depending on your use case, you must install other R libraries. For example, you can install **knitr** or **ggplot2**, in the same manner, that you installed RServe previously. You can use the *package.install* within your R shell to perform the installation.

For example:

~~~
```R

> install.packages('knitr')

```
~~~

# F

# Resource Allocation for Studio Services

## Resource Limits

For FCC Studio to run reliably, the available resources of the Kubernetes cluster have to be allocated accordingly. The components are mainly memory intensive and therefore we recommend setting memory constraints for each component.

## Resource Types

Each container requires a memory request and memory limit size as defined by the Kubernetes API. In short, containers specify a request, which is the amount of that resource that the system will guarantee to the container and a limit which is the maximum amount that the system will allow the container to use. For more information on troubleshooting tips, see Managing Compute Resources for Containers.

Some components require additional resource limits which are set as environment variables.

## Resource Parameters in FCC Studio

After extracting the Studio application installer software, the resource limits have to be adjusted for each component. The configuration files can be found in the `<Studio_Installation_Path>` folder.

**Note:** The sizing recommendations are preliminary. In the case of deployment failures, a manual configuration of the sizing parameters is needed.

*Table 7–5 Resource Parameters in FCC Studio*

| Configuration File/Container | Parameter type | Parameter Name | Description | Recommendation |
|---|---|---|---|---|
| server.yml / server | k8 | spec.containers[].resources.requests.memory | Memory request size for the FCC server (web application) component | default |
| | k8 | spec.containers[].resources.requests.memory | Memory limit size for the FCC server (web application) component | default |
| agent.yml / agent | k8 | spec.containers[].resources.requests.memory | Memory request size for the Agent (manages all interpreters) component | default |

**Table 7–5 Resource Parameters in FCC Studio**

| Configuration File/Container | Parameter type | Parameter Name | Description | Recommendation |
|---|---|---|---|---|
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the Agent (manages all interpreters) component | default |
| pgx-server.yml / pgx-server | k8 | spec.containers[].resources.requests.memory | Memory request size for the PGX server (manages graph processing) component | Slightly less than the memory of the PGX server as calculated in the sizing guide. |
| | k8 | spec.containers[].resources.requests.memory | Memory limit size for the PGX server (manages graph processing) component | The same as the request size above. |
| | ENV VAR (JAVA_OPTS) | -Xmx<br>-Xms | The maximum and minimum heap memory size (mainly used for storing graphs' string properties) for the Java process of PGX. | 58% of the container's memory limit size above.<br><br>For better understanding of this sizing parameter, please consult the PGX 19.2.1 Memory Consumption documentation. |
| | ENV VAR (JAVA_OPTS) | -Dpgx.max_off_heap_size | The maximum off-heap memory size in megabytes (mainly used for storing graphs except for their string properties) that PGX tries to respect. | 42% of the container's memory limit size above.<br><br>For better understanding of this sizing parameter, please consult the PGX 19.2.1 Memory Consumption documentation. |
| fcc-pgx.yml / pgx-interpreter | k8 | spec.containers[].resources.requests.memory | Memory request size for the PGX interpreter | 4Gi |

*Table 7–5 Resource Parameters in FCC Studio*

| Configuration File/Container | Parameter type | Parameter Name | Description | Recommendation |
|---|---|---|---|---|
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the PGX interpreter | 16Gi<br><br>Sizing should depend on the number and behavior (memory requirements of sessions) of concurrent users |
| etl.yml / etl | k8 | spec.containers[].resources.requests.memory | Memory request size for the ETL Initiation (mainly used to create the graph for PGX) component | Slightly less than the memory of the ETL Initiation server as calculated in the sizing guide. |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the ETL Initiation (mainly used to create the graph for PGX) component | The same as the request size above. |
| authservice.yml / authservice | k8 | spec.containers[].resources.requests.memory | Memory request size for the authservice (used for getting roles of a user from DB) component | default |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the authservice (used for getting roles of a user from DB) component | default |
| metaservice.yml / metaservice | k8 | spec.containers[].resources.requests.memory | Memory request size for the metaservice (used for custom interpreter api's like loaddataset, listdataset in scala interpreter etc.) component | default |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the metaservice (used for custom interpreter api's like loaddataset, listdataset in scala interpreter etc.) component | default |
| sessionservice.yml / sessionservice | k8 | spec.containers[].resources.requests.memory | Memory request size for the sessionservice (used for managing session between pgx and scala interpreter) component | default |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the sessionservice (used for managing session between pgx and scala interpreter) component | default |

*Table 7–5 Resource Parameters in FCC Studio*

| Configuration File/Container | Parameter type | Parameter Name | Description | Recommendation |
|---|---|---|---|---|
| batchservice.yml / batchservice | k8 | spec.containers[].resources.requests.memory | Memory request size for the batchservice (used for managing batches like sqoopjob, graph load, notebook execution etc)component | default |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the batchservice (used for managing batches like sqoopjob, graph load, notebook execution etc)component | default |
| fcc-jdbc.yml / fcc-jdbc | k8 | spec.containers[].resources.requests.memory | Memory request size for the jdbc connection | default |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the jdbc connection | default |
| fcc-livy.yml / fcc-livy | k8 | spec.containers[].resources.requests.memory | Memory request size for the livy connection to big data Spark cluster. | default |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the livy connection to big data Spark cluster. | default |
| fcc-markdown.yml / markdown-interpreter | k8 | spec.containers[].resources.requests.memory | Memory request size for the Markdown interpreter | default |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the Python interpreter | default |
| fcc-ore.yml / ore-interpreter | k8 | spec.containers[].resources.requests.memory | Memory request size for the ore connection | default |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the ore connection | default |
| fcc-python.yml / python-interpreter | k8 | spec.containers[].resources.requests.memory | Memory request size for the Python interpreter | depending on use case |
| | k8 | spec.containers[].resources.limits.memory | Memory limit size for the Python interpreter | depending on use case |

# G Uninstalling FCC Studio

To uninstall FCC Studio, follow these steps:

**Note:** Uninstalling the Studio application deletes all the data from FCC Studio namespace.

1. Delete the FCC Studio namespace using the following command:

   ```
   kubectl delete namespace fccs
   ```

2. Manually delete the FCC studio images for each Kubernetes node using the following command:

   ```
   docker rmi <Image ID>
   ```

   You can get the list of image IDs by running the docker images.

# H

# Creating Required Index Files

## Creating Required Index Files

To create the required index files, follow these steps:

1. Create the index files, `name_index.csv` and `address_index.csv` with the column names are per the configuration mentioned in the `<ETL_Installation_Path>/crosslangnamematch/conf/NameMatchingLocations.properties` file with dummy values.

   For more information on `NameMatchingLocations.properties` file, see the *Cross Language Name Matching Service* section in the *Studio Services* chapter in the *OFS Crime and Compliance Studio Administration Guide.*

   **Note:** The entries in the CSV file must be tab-separated.

   For example:

***Table H–1 name_index.csv***

| node_id | name | source |
|---|---|---|
| 1 | dummy | dummy |

***Table H–2 address_index.csv***

| node_id | address | source |
|---|---|---|
| 1 | dummy | dummy |

I

# Providing Spark Libraries

To provide your own Spark libraries and/or Hadoop client-configuration files to connect to a Yarn cluster, follow these steps to create a new Init Container/Configmap that copies your Spark libraries folder and/or Hadoop client-configuration folder into the right location, where the Spark interpreter can access it.

To provide Spark libraries, follow these steps:

1. Download the desired Spark libraries from the Spark Official Website.
2. Prepare Spark libraries.
    1. Place your libraries in the `<Studio_Installation_Path>/docker/user/spark-interpreter-libraries/` directory.

       **Note:** Ensure that the folder name of the Spark library is prefixed with the term ’spark’.

    2. To use separate Hadoop libraries, download them as well and place the Hadoop libraries folder in the same directory.
    3. Be cautious when linking the two libraries, since the path where they are located is in the K8s pod, and the location is as follows:

       `/var/olds-spark-interpreter/interpreter/spark/libs/`
3. Change the image of the Spark interpreter Init Container in the `<Studio_Installation_Path>/deployments/spark.yml` file to `{{imageRepository}}/fcc-studio/3rdparty:init`.
4. (Optional) Place your Hadoop Client Configuration files in the `<Studio_Installation_Path>/configmaps/spark-interpreter-conf/` path.

J

# Setting Spark Master

You can update the Spark Master by changing the default value of the `spark.master` property in the Interpreter Settings of the Spark interpreter (spark.json) or the ***Interpreters*** page of the Studio application UI after startup.

- The default value of `spark.master` property is `local[*]`, which means that the interpreter will run in local mode.
- For a Yarn cluster, you must change the default value of `spark.master` property to yarn-client.

**Note:** The Hadoop client-configuration files are required to connect to a Yarn cluster.

# K Using Python Virtual Environments with PySpark

To use Python Virtual Environments with PySpark, follow these steps:

1. Creating a Virtual Environment with Conda
2. Including Virtual Environment in the Init Container
3. Updating Interpreter Properties

## Creating a Virtual Environment with Conda

**Note:** You can also use virtualenv to create your virtual environment instead of Conda.

1. Ensure that you have Conda and Conda-Pack installed.
2. Create your virtual environment using the following command:

   ```
   conda create -y -n <environment-name> python=<python-version>
   <additional-packages>
   ```

   **Note:** The `<environment-name>` can be chosen freely and subsequently must be used in further commands.

3. Activate your virtual environment using:

   ```
   conda activate <environment-name>
   ```

4. Execute the following command to obtain the path to your virtual environment:

   ```
   which python
   ```

   The obtained result is referred to as `<environment-abs-path>`

5. Compress your virtual environment using the following command:

   ```
   conda pack -n <environment-name> -o
   <environment-abs-path>/<environment-name>.tar.gz
   ```

## Including Virtual Environment in the Init Container

To include the virtual environment in the Init container, you must place the Virtual Environment in the same path as the Spark libraries. For more information, see Appendix I, "Providing Spark Libraries"

## Updating Interpreter Properties

All the properties can either be configured in the interpreter JSON files or from the ***Interpreters*** page in the Studio application UI after starting the Studio application.

- In the ***Spark Interpreter Settings*** page of the Studio application UI (or `spark.json` file), change the following:
  - Change the value of the `spark.yarn.dist.archives` property to `/var/olds-spark-interpreter/interpreter/spark/libs/<environment-name>/<environment-name>.tar.gz#<environment-name>`
  - Change the value of the `spark.pyspark.python` property to `./<environment-name>/bin/python`
- In the ***PySpark Interpreter Settings*** page of the Studio application UI (or `pyspark.json` file), change the value of the `zeppelin.pyspark.python` property to `/var/olds-spark-interpreter/interpreter/spark/libs/<environment-name>/bin/python`.