Oracle® Fusion Middleware

Oracle Stream Analytics Install Guide for Hadoop 2.7 and Higher

F18428-06
April 2021

Oracle Fusion Middleware Oracle Stream Analytics Install Guide for Hadoop 2.7 and Higher,

F18428-06

Primary Author: Oracle Corporation

Contents

Preface

Audience v
Documentation Accessibility v
Conventions v
Related Documents vi

1 How to Install GoldenGate Stream Analytics?

1.1 Planning Your Installation 1-1
1.2 Installing GoldenGate Stream Analytics 1-2
1.3 Configuring the Metadata Store 1-2
1.3.1 Configuring ATP/ADW as Metadata Store 1-6
1.4 Initializing Metadata Store 1-7
1.5 Jetty Properties File 1-8
1.6 Adjusting Jetty Threadpool 1-9
1.7 Integrating Stream Analytics with Oracle GoldenGate 1-10
1.8 Maven Setting for GoldenGate Big Data Handlers 1-10
1.8.1 Set the Maven Home Path 1-10
1.8.2 Configure Maven Proxy Settings 1-11

2 Launching and Terminating GoldenGate Stream Analytics

2.1 Retaining https and Disabling http 2-1
2.2 Setting up Runtime for GoldenGate Stream Analytics Server 2-1
2.3 Validating Data Flow to GoldenGate Stream Analytics 2-5
2.4 Terminating GoldenGate Stream Analytics 2-6
2.5 Changing the Admin Password and Creating New Users 2-6

3 GoldenGate Stream Analytics Hardware Requirements for Enterprise Deployment

4 Upgrading GoldenGate Stream Analytics

4.1 Upgrading GoldenGate Stream Analytics 4-1

Preface

Installing Oracle Stream Analytics for Hadoop 2.7 and Higher describes the prerequisites for installing Oracle Stream Analytics, steps to install it, configure it, launch it, and terminate it.

Topics

- Audience
- Documentation Accessibility
- Conventions
- Related Documents

Audience

This document is intended for developers and users who need to install and configure Oracle Stream Analytics.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Accessible Access to Oracle Support

Oracle customers who have purchased support have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Related Documents

Documentation for Oracle Stream Analytics is available on Oracle Help Center.

Also see the following documents for reference:

- *Understanding Oracle Stream Analytics*
- *Developing Custom Stages and Functions*
- *Known Issues in Oracle Stream Analytics*
- *Spark Extensibility for CQL in Oracle Stream Analytics*
- *Using Oracle Stream Analytics*

1

How to Install GoldenGate Stream Analytics?

GoldenGate Stream Analytics (GGSA) is a scalable and reliable platform for building stream processing applications. Sample use cases include Real-time Fraud Detection, Real-time Location-based Marketing, Real-time Smart Inventory Management, Real-time Dynamic Pricing, Real-time Asset Tracking, etc. This document helps you with the installation of Oracle Stream Analytics.

Topics:

- Planning Your Installation
- Installing GoldenGate Stream Analytics
- Configuring the Metadata Store
- Initializing Metadata Store

1.1 Planning Your Installation

To plan the installation of GoldenGate Stream Analytics (GGSA) 19.1.0.0.6 efficiently, ensure that you have the required hardware and software. You should also perform the prerequisite procedures before starting the installation process.

You can use the information in the certification matrix before installing GoldenGate Stream Analytics 19.1.0.0.6. The certification matrix provides you useful links to support pages, supported software, and system requirements in general. The following software is required for operation of GGSA:

- Oracle JDK 8 Update 131 and higher versions
- Repository Database
  - Oracle Database versions 12.2.0.1 or higher, 12.1.0.1 or higher, and 11.2.0.4 or higher
  - Else, you can use MySQL version 5.6 or 5.7
- A running Hadoop cluster with version 2.7 or higher
- A running Kafka cluster with version 0.10.2 or higher
- Locally installed Spark Libraries with version 2.2.1 - 2.4.3, built for Hadoop 2.7. OSA does not package the Spark client libraries, so you will also need locally installed Spark with version 2.4.3 built for Hadoop 2.7.

> **Note:**
>
> Install Spark and JDK in the same node on which you plan to install Oracle Stream Analytics. See Installing GoldenGate Stream Analytics.

- Google Chrome browser with version 6.0 or higher

1.2 Installing GoldenGate Stream Analytics

After you have reviewed the above software prerequisites, please follow the steps below to install GoldenGate Stream Analytics 19.1.0.0.6:

1. Create a directory, for example, `spark-downloads`, and download Apache Spark into the newly created folder and as specified by versions below:
   - Spark release: `2.4.3`
   - Package type: `Pre-built for Apache Hadoop 2.7 and later`
   - Download Spark: `spark-2.4.3-bin-hadoop2.7.tgz`
2. Extract the Spark archive to a local directory.

   You can see a subfolder, `spark-2.4.3-bin-hadoop2.7`.
3. Create a new directory, for example, `OSA-19` and download `OSA-19.1.0.0.6.zip` from Oracle eDelivery and extract it into the newly created folder.

   You can find the `OSA-19.1.0.0.6-README.txt` file in the `OSA-19.1.0.0.6` zip file.
4. Extract the downloaded file. You should now see a subfolder `OSA-19.1.0.0.6`.
5. Review the file `OSA-19.1.0.0.6-README.txt` in the OSA-19 folder.
6. Set the environment variables:
   - Set the `SPARK_HOME` environment variable in the `OSA-19.1.0.0.6/osa-base/etc/osa-env.sh` file to point to the directory where you have extracted the Spark archive. For example:

     ```
     SPARK_HOME=/products/spark-downloads/spark-2.4.3-bin-hadoop2.7
     ```
7. Set the `JDK_HOME` environment variable in the `OSA-19.1.0.0.6/osa-base/etc/osa-env.sh` file to point to the directory where you have extracted the JDK archive. For example:

   ```
   JDK_HOME=/products/java-downloads/jdk1.8.0_131
   ```

1.3 Configuring the Metadata Store

Please follow steps below for configuring your metadata store.

1. Configure your data source in `OSA-19.1.0.0.6/osa-base/etc/jetty-osa-datasource.xml` as per instructions below. This step is essential for creating OSA's database schema. The OSA database user referred to in the document will be created by the installation process.
2. Uncomment and edit one of the two Data source configurations, either for Oracle Database or MySQL depending on the database you want to use as metadata store. The uncommented fragment for Oracle database is shown below: **a.**

   ```
   <New id="osads"
   class="org.eclipse.jetty.plus.jndi.Resource">
             <Arg>
             <Ref refid="wac"/>
             </Arg>
   ```

```
              <Arg>jdbc/OSADataSource</Arg>
            <Arg>
             <New class="oracle.jdbc.pool.OracleDataSource">
               <Set
name="URL">jdbc:oracle:thin:@myhost.example.com:1521:OSADB</Set>
                <Set name="User">OSA_USER</Set>
                <Set name="Password">
                  <Call
class="org.eclipse.jetty.util.security.Password"
name="deobfuscate">
                    <Arg> OBF:OBFUSCATED_PASSWORD</
Arg>
                  </Call>
                </Set>
                <Set name="connectionCachingEnabled">true</
Set>
                <Set name="connectionCacheProperties">
                 <New class="java.util.Properties">
                 <Call name="setProperty"><Arg>MinLimit</
Arg><Arg>1</Arg></Call>
                 <Call name="setProperty"><Arg>MaxLimit</
Arg><Arg>15</Arg></Call>
                 <Call name="setProperty"><Arg>InitialLimit</
Arg><Arg>1</Arg></Call>
                 </New>
               </Set>
             </New>
            </Arg>
        </New>
```

3. Decide on an OSA schema username and a plain-text password. For illustration, say **osa** as schema user name and **alphago** as password.
   Change directory to top-level folder `OSA-19.1.0.0.6` and execute the following command:`java -cp ./lib/ jetty-util-9.4.17.v20190418.jar org.eclipse.jetty.util.security.Password osa <your password>`

   For example, `java -cp ./lib/ jetty-util-9.4.17.v20190418.jar org.eclipse.jetty.util.security.Password osa alphago`

   You should see results like below on console:

   ```
   2019-06-18 14:14:45.114:INFO::main: Logging initialized @1168ms to
   org.eclipse.jetty.util.log.StdErrLogalphago
   OBF:<obfuscated password>

   MD5:34d0a556209df571d311b3f41c8200f3

   CRYPT:osX/8jafUvLwA
   ```

4. Note down the obfuscated password string that is displayed (shown in bold), by copying it to clipboard or notepad.
5. Change database host, port, SID, osa schema user name and osa schema password fields marked in bold in the code in Step 2a.
   Example - jdbc:oracle:thin:@myhost.example.com:1521:ORCL

SAMPLE JETTY-OSA-DATASOURCE.XML

```
<?xml version="1.0"?>
<!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN" "http://
www.eclipse.org/jetty/configure_9_3.dtd">

<!-- =============================================================== -->
<!-- Configure jdbc/OSADataSource data source                        -->
<!-- =============================================================== -->
<Configure id="Server" class="org.eclipse.jetty.server.Server">

    <!-- SAMPLE OSA DATASOURCE CONFIGURATION FOR ORACLE-->
    <New id="osads" class="org.eclipse.jetty.plus.jndi.Resource">
         <Arg>
             <Ref refid="wac"/>
         </Arg>
         <Arg>jdbc/OSADataSource</Arg>
         <Arg>
             <New class="oracle.jdbc.pool.OracleDataSource">
                 <Set
name="URL">jdbc:oracle:thin:@myhost.example.com:1521:OSADB</Set>
                 <Set name="User">osa_prod</Set>
                 <Set name="Password">
                    <Call
class="org.eclipse.jetty.util.security.Password" name="deobfuscate">

<Arg>OBF:1ggz1j1u1k8q1leq1v2h1w8v1v1x1lcs1k5g1iz01gez</Arg>
                    </Call>
                </Set>
                <Set name="connectionCachingEnabled">true</Set>
                <Set name="connectionCacheProperties">
                    <New class="java.util.Properties">
                        <Call name="setProperty"><Arg>MinLimit</
Arg><Arg>1</Arg></Call>
                        <Call name="setProperty"><Arg>MaxLimit</
Arg><Arg>15</Arg></Call>
                        <Call name="setProperty"><Arg>InitialLimit</
Arg><Arg>1</Arg></Call>
                    </New>
                </Set>
             </New>
         </Arg>
     </New>


    <!-- SAMPLE OSA DATASOURCE CONFIGURATION FOR ADW-->
    <!--
    <New id="osads" class="org.eclipse.jetty.plus.jndi.Resource">
         <Arg>
             <Ref refid="wac"/>
         </Arg>
         <Arg>jdbc/OSADataSource</Arg>
         <Arg>
             <New class="oracle.jdbc.pool.OracleDataSource" type="adw">
```

```
                <Set name="URL">jdbc:oracle:thin:@oracletestdb_high?
TNS_ADMIN=/scratch/oracletest/Wallet_oracletestdb</Set>
                <Set name="User">{OSA_USER}</Set>
                <Set name="Password">
                   <Call
class="org.eclipse.jetty.util.security.Password" name="deobfuscate">
                       <Arg>{OBF:OBFUSCATE_PASSWORD}</Arg>
                   </Call>
               </Set>
               <Set name="connectionCachingEnabled">true</Set>
               <Set name="connectionCacheProperties">
                   <New class="java.util.Properties">
                       <Call name="setProperty"><Arg>MinLimit</
Arg><Arg>1</Arg></Call>
                       <Call name="setProperty"><Arg>MaxLimit</
Arg><Arg>15</Arg></Call>
                       <Call name="setProperty"><Arg>InitialLimit</
Arg><Arg>1</Arg></Call>
                   </New>
               </Set>
            </New>
         </Arg>
     </New>
     -->
    <!-- SAMPLE OSA DATASOURCE CONFIGURATION FOR MYSQL-->
    <!--

    <New id="osads" class="org.eclipse.jetty.plus.jndi.Resource">
        <Arg>
            <Ref refid="wac"/>
        </Arg>
        <Arg>jdbc/OSADataSource</Arg>
        <Arg>
            <New
class="com.mysql.cj.jdbc.MysqlConnectionPoolDataSource">
                <Set name="URL">jdbc:mysql://examplehost.com:3306/
OSADB</Set>
                <Set name="User">{OSA_USER}</Set>
                <Set name="Password">
                   <Call
class="org.eclipse.jetty.util.security.Password" name="deobfuscate">
                       <Arg>{OBF:OBFUSCATE_PASSWORD}</Arg>
                   </Call>
               </Set>
            </New>
        </Arg>
    </New>
-->

</Configure>
```

> **Note:**
>
> Do not use a hyphen in the OSA metadata username, in the jetty-osa-datasource.xml

.

1.3.1 Configuring ATP/ADW as Metadata Store

GoldenGate Stream Analytics creates the metadata schema, as part of initial configuration of the system, using the script: `${OSA_HOME}/osa-base/bin/configure.sh dbroot=<sys user of database> dbroot_password=<sys user password of the database>`

However, before running the above script, you must configure the datasource in the datasource configuration file at`${OSA_HOME}/osa-base/etc/jetty-osa-datasource.xml`.

To configure ATP/ADW as metadata store, first comment the Oracle and MYSQL sections, while uncommenting the ADW/APT section in `jetty-osa-datasource.xml` file.

Below is the template for the datasource configuration for ATP/ADW database:

`jetty-osa-datasource.xml`

```
<?xml version="1.0"?>
<!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN" "http://
www.eclipse.org/jetty/configure_9_3.dtd">
<Configure id="Server" class="org.eclipse.jetty.server.Server">
    <New id="osads" class="org.eclipse.jetty.plus.jndi.Resource">
         <Arg>
             <Ref refid="wac"/>
         </Arg>
         <Arg>jdbc/OSADataSource</Arg>
         <Arg>
             <New class="oracle.jdbc.pool.OracleDataSource" type="adw">
                 <Set name="URL">jdbc:oracle:thin:@{service_name}?
TNS_ADMIN={wallet_absolute_path}</Set>
                 <Set name="User">{osa_db_user}</Set>
                 <Set name="Password">
                    <Call
class="org.eclipse.jetty.util.security.Password" name="deobfuscate">
                        <Arg>{obfuscated_password}</Arg>
                    </Call>
                </Set>
                <Set name="connectionCachingEnabled">true</Set>
                <Set name="connectionCacheProperties">
                    <New class="java.util.Properties">
                        <Call name="setProperty"><Arg>MinLimit</
Arg><Arg>1</Arg></Call>
                        <Call name="setProperty"><Arg>MaxLimit</
Arg><Arg>15</Arg></Call>
                        <Call name="setProperty"><Arg>InitialLimit</
```

```
Arg><Arg>1</Arg></Call>
                    </New>
                  </Set>
                </New>
              </Arg>
        </New>
</Configure>
```

> **Note:**
>
> In the above template, replace the variables in {} as below:
>
> - {*service_name*} - one of the service names listed in the tnsnames.ora file inside the wallet
> - {*wallet_absolute_path*} - the absolute path of wallet folder on the machine where OSA is installed
> - {*osa_db_user*} - the username to create the osa metadata. This username and schema will be created by the 'dbroot' user provided in above script.
> - {*obfuscated_password*} - the Obfuscated password for {osa_db_user}

1.4 Initializing Metadata Store

This topic applies only to Oracle user-managed services.

After installing GGSA, you need to configure the metadata store with the database admin credential details and the version of GGSA as required.

To initialize the metadata store, you need database admin credentials with sysdba privileges:

1. Change directory to `OSA-19.1.0.0.6/osa-base/bin`.
2. Execute the following command:`./start-osa.sh dbroot=<db sys user> dbroot_password=<db sys password>` For example, `./start-osa.sh dbroot=AlphaUser dbroot_password=AlphaPassword`
3. Following console messages will be displayed indicating the OSA schema was created and prompting for password for **osaadmin** user.

   The following console messages indicates that the GGSA schema is created and the metadata store is successfully initialized:

   ```
   JAVA_HOME: JAVA_HOME: /<jdk install path>/jdk1.8.0_121.
            SPARK_HOME: /<Spark install path>/spark-2.4.3-bin-hadoop2.7
            The RELEASE file exists:
            /<Spark install path>/spark-2.4.3-bin-hadoop2.7/RELEASE
            SPARK_VERSION: Spark 2.4.3 built for Hadoop 2.7.3
            2019-06-23 08:48:51.444:INFO::main:
            Logging initialized @305ms to
   org.eclipse.jetty.util.log.StdErrLog
   ```

```
        OSA DB user created:osa
        The OSA application administrative user with the
predefined name "osaadmin" is going to be created
        You have not specified a password
        for the "osaadmin" user on the command line. Please enter
it below.
```

4. Enter password:
5. Reenter password:
6. Run `./stop-osa.sh` to complete schema creation and metadata initialization.
7. If you don't see the above messages, check the OSA-19.1.0.0.6/osa-base/logs folder to identify the cause and potential solution.

> **Note:**
>
> If you do not have the database admin credentials, ask your database administrator to create a Oracle Stream Analytics database user by using the SQL scripts available in the `OSA-19.1.0.0.6/osa-base/sql` folder. The Oracle Stream Analytics database username must match the one configured in `jetty-osa-datasource.xml`.

1.5 Jetty Properties File

Use the jetty properties available at `OSA-19.1.0.0.6/osa-base/etc/jetty.properties`, to modify certain security features.

> **Note:**
>
> It is recommended that you configure these properties at the installation stage, to avoid restarting your server, if configured at a later stage.

Following are the available properties:

- **jetty.session.timeout**
  You can set the timeout for OSA web session. This sets the timeout for OSA web session. By default the timeout is set to 30 minutes. The value can be changed to any integer greater than 1.
- **host.headers.whitelist**
  You can restrict the x-forwarded-host header values to the values defined with this property.

  Example: `host.headers.whitelist= www.oracle.com, www.microsoft.com, localhost:9080`

  Here the value of the host header can be only of these three domains listed. Commenting out this property with a # will allow all values for the header.

> **Note:**
>
> If you do not specify explicitly the host header in your request, the default value is `host-server:port`, where the OSA jetty server is running. Hence you must specify the port number along with the server address.

- **xforwarded.host.headers.whitelist**
  You can restrict the x-forwarded-host header values to the values defined with this property.

  Example: `xforwarded.host.headers.whitelist= www.oracle.com, www.microsoft.com, localhost`

  Here the value of the x-forwarded-host header can be only of these three domains listed. Commenting out this property with a # will allow all values for the header. If no domain is entered, that is, if the value of the property is empty, then this header is not supported.

- **response.headers.list**
  A comma separated list of response headers, which will be sent along with response for every request.

  Example: `response.headers.list="x-frame-options: sameorigin, X-Content-Type-Options: nosniff"`

  By default the above 2 response headers are set.

  – `x-frame-options`: sameorigin will prevent clickjack attacking.

  – `X-Content-Type-Options`: nosniff will prevent sniffing of the response content by the browsers.

1.6 Adjusting Jetty Threadpool

Edit `OSA-19.1.0.0.6/etc/jetty-threadpool.xml` to change minimum and maximum thread configuration to 100 and 2000 respectively. Sample shown below.

```
<New id="threadPool"
    class="org.eclipse.jetty.util.thread.QueuedThreadPool">
 <Set name="minThreads" type="int"><Property
name="jetty.threadPool.minThreads"
      deprecated="threads.min" default="100"/></Set>
<Set name="maxThreads" type="int"><Property
name="jetty.threadPool.maxThreads"
      deprecated="threads.max" default="2000"/></Set>
<Set name="reservedThreads" type="int"><Property
      name="jetty.threadPool.reservedThreads" default="-1"/></Set>
 <Set name="idleTimeout" type="int"><Property
      name="jetty.threadPool.idleTimeout" deprecated="threads.timeout"
      default="60000"/></Set>
<Set name="detailedDump" type="boolean"><Property
      name="jetty.threadPool.detailedDump" default="false"/></Set>
</New>
</Configure>
```

1.7 Integrating Stream Analytics with Oracle GoldenGate

Follow the below steps to integrate Oracle Goldengate with Stream Analytics:

1. Download and install Oracle GoldenGate Big Data. For a compatible version of Oracle GoldenGate Big Data, see the latest certification matrix.

   > **Note:**
   >
   > Install Oracle GoldenGate Big Data on the same machine and with the same user as OSA.

2. Set the following environment variables:
   - *KAFKA_HOME* – set this variable to the path where Kafka is installed.
     Example :`export KAFKA_HOME=/u01/app/kafka`.
   - *LD_LIBRARY_PATH* – set this variable to the directory path that contains JVM shared library.
     Example: `export LD_LIBRARY_PATH=/u01/app/java/jre/lib/amd64/server:$LD_LIBRARY_PATH`
   - *GGBD_HOME* – set this variable to the path where Goldengate for Bigdata is installed.
     Example: `export GGBD_HOME=/u01/app/OGG_BigData_Linux_x64_19.1.0.0.0`
3. Start the manager process on port 7801.

For installation steps, see Installing GoldenGate for Big Data.

1.8 Maven Setting for GoldenGate Big Data Handlers

Maven is required to download third-party client libraries for the GGBD handlers to work.

1.8.1 Set the Maven Home Path

To configure maven home:

Update the `OSA-19.1.0.0.6/osa-base/bin/configure-osa.sh` with the correct M2_HOME path, as below:
Change the path from

```
OSA_HOME="$( cd "$(dirname "../../")" >/dev/null 2>&1 ; pwd -
P )"
```

to

```
OSA_HOME="$( cd "$(dirname "../../../")" >/dev/null 2>&1 ; pwd
-P )"
```

> **Note:**
>
> Update the maven home path before initialization of the metadata store, or you will have to restart GGSA after this update.

1.8.2 Configure Maven Proxy Settings

If your GGSA installation is behind proxy, to use the GGBD handlers, you have to configure the `settings.xml` that comes with the Maven distribution.

Update the `<OSA_INSTALLATION_PATH>/apache-maven-3.6.3/conf/settings.xml` with the correct proxy entries in the `<proxies> </proxies>` section, as shown below:

```
<proxy>
      <id>optional</id>
      <active>true</active>
      <protocol>http</protocol>
      <username>proxyuser</username>
      <password>proxypass</password>
      <host>proxy.host.net</host>
      <port>80</port>
      <nonProxyHosts>local.net|some.host.com</nonProxyHosts>
    </proxy>
```

> **Note:**
>
> Username and password field is required if the proxy is protected.

> **Note:**
>
> Update the `settings.xml` before initialization of the metadata store, or you will have to restart GGSA after this update.

2

Launching and Terminating GoldenGate Stream Analytics

Once you have completed the post installation tasks, you are ready to launch GoldenGate Stream Analytics (GGSA), and start using it. Launching and terminating GGSA is easy and you just need to run simple commands to do them.

Topics

- Retaining https and Disabling http
- Setting up Runtime for GoldenGate Stream Analytics Server
- Validating Data Flow to GoldenGate Stream Analytics
- Terminating GoldenGate Stream Analytics
- Changing the Admin Password and Creating New Users

2.1 Retaining https and Disabling http

1. By default, the GGSA web application is available on both http (port 9080) and https (port 9443). Follow the procedure below if you intend to disable http.
2. Edit file `osa-base/start.d/http.ini`.
3. Comment out as follows: `##--module=http`.
4. Start GGSA web server by running `osa-base/bin/start-osa.sh`.

2.2 Setting up Runtime for GoldenGate Stream Analytics Server

Before you start using GoldenGate Stream Analytics, you need to specify the runtime server, environment, and node details. You must do this procedure right after you launch Oracle Stream Analytics for the first time.

1. Change directory to `OSA-19.1.0.0.6/osa-base/bin and run ./start-osa.sh`. You should see the following message on console.

   ```
   Supported OSA schema versions are: [18.4.3, 18.1.0.1.0, 18.1.0.1.1,
   19.1.0.0.0, 19.1.0.0.1, 19.1.0.0.2, 19.1.0.0.3, 19.1.0.0.5,
   19.1.0.0.6]

   The schema is preconfigured and current. No changes or updates are
   required.
   ```

   If you do not see the above message, please check the log file in `OSA-19.1.0.0.6/osa-base/logs` folder.
2. the Chrome browser, enter `localhost:9080/osa` to access Oracle Stream Analytics login page, and login using your credentials.

> **Note:**
>
> The password is a plain-text password.

3. Click the user name at the top right corner of the screen.
4. Click **System Settings**.
5. Click **Environment**.
6. Select the **Runtime Server**. See the sections below for Yarn and Spark Standalone runtime configuration details.

Yarn Configuration

1.
   - **YARN Resource Manager URL**: Enter the URL where the YARN Resource Manager is configured.
   - **Storage**: Select the storage type for pipelines. To submit a GGSA pipeline to Spark, the pipeline has to be copied to a storage location that is accessible by all Spark nodes.
     - **If the storage type is WebHDFS**:
       - **Path**: Enter the WebHDFS directory (hostname:port/path), where the generated Spark pipeline will be copied to and then submitted from. This location must be accessible by all Spark nodes. The user specified in the authentication section below must have read-write access to this directory.
       - **HA Namenodes**: Set the HA namenodes. If the hostname in the above URL refers to a logical HA cluster, specify the actual namenodes here, in the format:`Hostname1:Port, Hostname2:Port`.
     - **If storage type is HDFS**:
       - **Path**: The path could be `<HostOrIPOfNameNode><HDFS Path>`. For example, `xxx.xxx.xxx.xxx/user/oracle/ggsapipelines`. Hadoop user must have `Write` permissions. The folder will automatically be created if it does not exist.
       - **HA Namenodes**: If the hostname in the above URL refers to a logical HA cluster, specify the actual namenodes here, in the format:`Hostname1:Port, Hostname2:Port`.
     - **If storage type is NFS**:
       Path: The path could be `/oracle/spark-deploy`.

     > **Note:**
     >
     > /oracle should exist and spark-deploy will automatically be created if it does not exist. You will need `Write` permissions on the /oracle directory.

2. **Hadoop Authentication:**
   - **Simple** authentication credentials:

  - **Protection Policy**: Select a protection policy from the drop-down list. This value should match the value on the cluster.
  - **Username**: Enter the user account to use for submitting Spark pipelines. This user must have read-write access to the Path specified above.
- **Kerberos** authentication credentials:
  - **Protection Policy**: Select a protection policy from the drop-down list. This value should match the value on the cluster.
  - **Kerberos Realm**: Enter the domain on which Kerberos authenticates a user, host, or service. This value is in the `krb5.conf file`.
  - **Kerberos KDC**: Enter the server on which the Key Distribution Center is running. This value is in the `krb5.conf file`.
  - **Principal**: Enter the GGSA service principal that is used to authenticate the GGSA web application against Hadoop cluster, for application deployment. This user should be the owner of the folder used to deploy the GGSA application in HDFS. You have to create this user in the yarn node manager as well.
  - **Keytab**: Enter the keytab pertaining to GGSA service principal.
  - **Yarn Resource Manager Principal**: Enter the yarn principal. When Hadoop cluster is configured with Kerberos, principals for hadoop services like hdfs, https, and yarn are created as well.

3. **Yarn master console port**: Enter the port on which the Yarn master console runs. The default port is `8088`.
4. Click **Save**.

Spark Standalone

1. Select the **Runtime Server** as **Spark Standalone**, and enter the following details:
   - **Spark REST URL**: Enter the Spark standalone REST URL. If Spark standalone is HA enabled, then you can enter comma-separated list of active and stand-by nodes.
   - **Storage**: Select the storage type for pipelines. To submit a GGSA pipeline to Spark, the pipeline has to be copied to a storage location that is accessible by all Spark nodes.
     - **If the storage type is WebHDFS**:
       * **Path**: Enter the WebHDFS directory (hostname:port/path), where the generated Spark pipeline will be copied to and then submitted from. This location must be accessible by all Spark nodes. The user specified in the authentication section below must have read-write access to this directory.
       * **HA Namenodes**: If the hostname in the above URL refers to a logical HA cluster, specify the actual namenodes here, in the format:`Hostname1:Port, Hostname2:Port`.
     - **If storage type is HDFS**:
       * **Path**: The path could be `<HostOrIPOfNameNode><HDFS Path>`. For example, `xxx.xxx.xxx.xxx/user/oracle/ggsapipelines`. Hadoop user must have `Write` permissions. The folder will automatically be created if it does not exist.

    - **HA Namenodes**: If the hostname in the above URL refers to a logical HA cluster, specify the actual namenodes here, in the format:`Hostname1:Port, Hostname2:Port`.

      This field is applicable only when the storage type is HDFS.

  - **Hadoop Authentication for WebHDFS and HDFS Storage Types:**
    - **Simple** authentication credentials:
      - **Protection Policy**: Select a protection policy from the drop-down list.
      - **Username**: Enter the user account to use for submitting Spark pipelines. This user must have read-write access to the Path specified above.
    - **Kerberos** authentication credentials:
      - **Protection Policy**: Select a protection policy from the drop-down list.
      - **Kerberos Realm**: Enter the domain on which Kerberos authenticates a user, host, or service. This value is in the `krb5.conf file`.
      - **Kerberos KDC**: Enter the server on which the Key Distribution Center is running. This value is in the `krb5.conf file`.
      - **Principal**: Enter the GGSA service principal that is used to authenticate the GGSA web application against Hadoop cluster, for application deployment. This user should be the owner of the folder used to deploy the GGSA application in HDFS. You have to create this user in the yarn node manager as well.
      - **Keytab**: Enter the keytab pertaining to GGSA service principal.
      - **Yarn Resource Manager Principal**: Enter the yarn principal. When Hadoop cluster is configured with Kerberos, principals for hadoop services like hdfs, https, and yarn are created as well.
  - **If storage type is NFS**:
    Path: The path could be `/oracle/spark-deploy`.

    > **Note:**
    >
    > /oracle should exist and spark-deploy will automatically be created if it does not exist. You will need `Write` permissions on the /oracle directory.

2. **Spark standalone master console port**: Enter the port on which the Spark standalone console runs. The default port is `8080`.

   > **Note:**
   >
   > The order of the comma-separated ports should match the order of the comma-separated spark REST URLs mentioned in the **Path**.

3. **Spark master username**: Enter your Spark standalone server username.

4. **Spark master password**: Click **Change Password**, to change your Spark standalone server password.

> **Note:**
>
> You can change your Spark standalone server username and password in this screen. The username and password fields are left blank, by default.

5. Click **Save**.

2.3 Validating Data Flow to GoldenGate Stream Analytics

After you have configured GoldenGate Stream Analytics (GGSA) with the runtime details, you need to ensure that sample data is being detected and correctly read by GGSA.

To validate data flow into GoldenGate Stream Analytics, use the following steps:

1. Copy the six lines below into a CSV file, for example `sample.csv`.

```
ProductLn,ProductType,Product,OrderMethod,CountrySold,QuantitySold,U
nitSalePrice
Personal Accessories,Watches,Legend,Special,Brazil,1,240
Outdoor Protection,First Aid,Aloe Relief,E-mail,United States,3,5.23
Camping Equipment,Lanterns,Flicker Lantern,Telephone,Italy,3,35.09
Camping Equipment,Lanterns,Flicker Lantern,Fax,United States,4,35.09
Golf Equipment,Irons,Hailstorm Steel Irons,Telephone,Spain,5,461
```

2. In the **Catalog**, as shown in the image below, click **Create New Item**, and then click **Stream**. create a stream of type File.
3. In the **Type Properties** page of the **Create Stream** dialog box, provide the **Name**, **Description**, and **Tags** for the Stream, select the **Stream Type** as **File**, and then select **Create Pipeline with this source (Launch Pipeline Editor)**.
4. Click the **Next** button to navigate to the **Source Details** page of the **Create Stream** dialog box.
5. In the **Source Details** page, click **Upload file** to upload the `sample.csv` file, and then click **Next** to navigate to the **Data Format** page.
6. In the **Data Format** page, select the **CSV Predefined Format** as **Default** and select the **First record as header**, and then click **Next** to navigate to the **Shape** page.
7. In the **Shape** page, verify that the shape of the event is successfully inferred as in the following image, and then click **Save**.
8. In the **Create Pipeline** dialog box, enter the **Name**, **Description**, **Tags** of the pipeline, select the **Stream** that you created, and then click **Save**:

   You can see the pipeline editor and you can see the message `Starting Pipeline` followed by the message `Listening to Events`.

> **Note:**
>
> This is the first access of the cluster and it takes time to copy libraries, please be patient. You should eventually see the screenshot below with single node representing the stream source.

To complete your pipeline, see Creating a Pipeline.

2.4 Terminating GoldenGate Stream Analytics

You can terminate Oracle Stream Analytics by running a simple command.

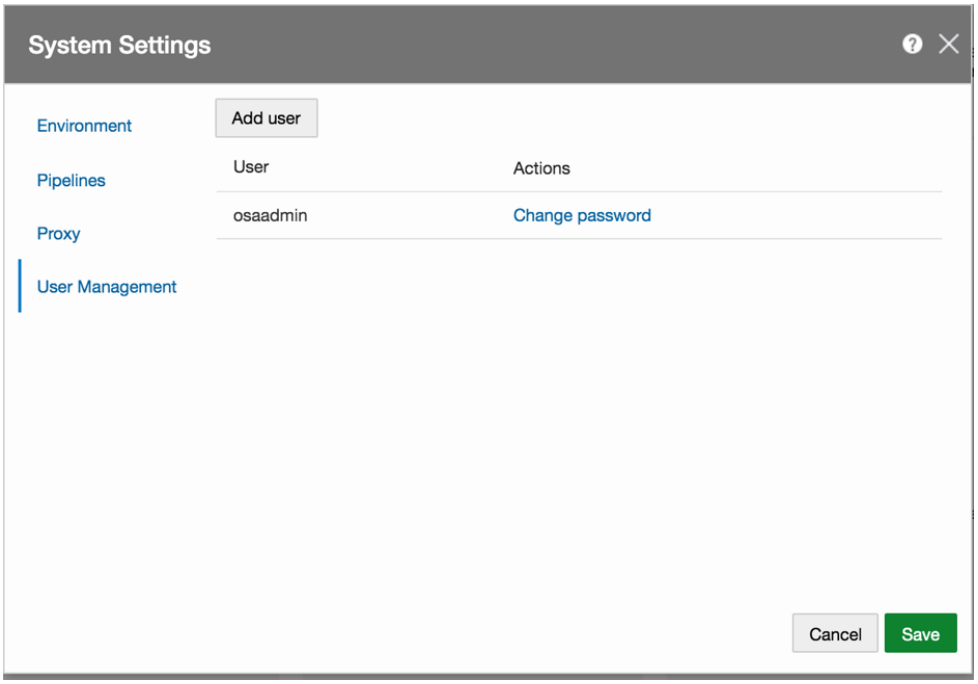
Use the following command to terminate GoldenGate Stream Analytics:

```
./stop-osa.sh from OSA-19.1.0.0.6/osa-base/bin folder
```

2.5 Changing the Admin Password and Creating New Users

You can change the Admin password and add new users from the System Settings dialog box.

1. Click the user profile at the top right corner, and then click **System Settings**.
2. On the **System Settings** dialog box, click **User Management**.



3. Add new users or change password as required.

3

GoldenGate Stream Analytics Hardware Requirements for Enterprise Deployment

This chapter provides the hardware requirements for GoldenGate Stream Analytics Design and Data tiers.

Design Tier

GoldenGate Stream Analytics' Design-tier is a multi-user environment that allows users to implement and test dataflow pipelines. The design-tier also serves dashboards for streaming data. Multiple users can build, test, and deploy pipelines based on the capacity of the Runtime-tier (YARN/Spark cluster) simultaneously.

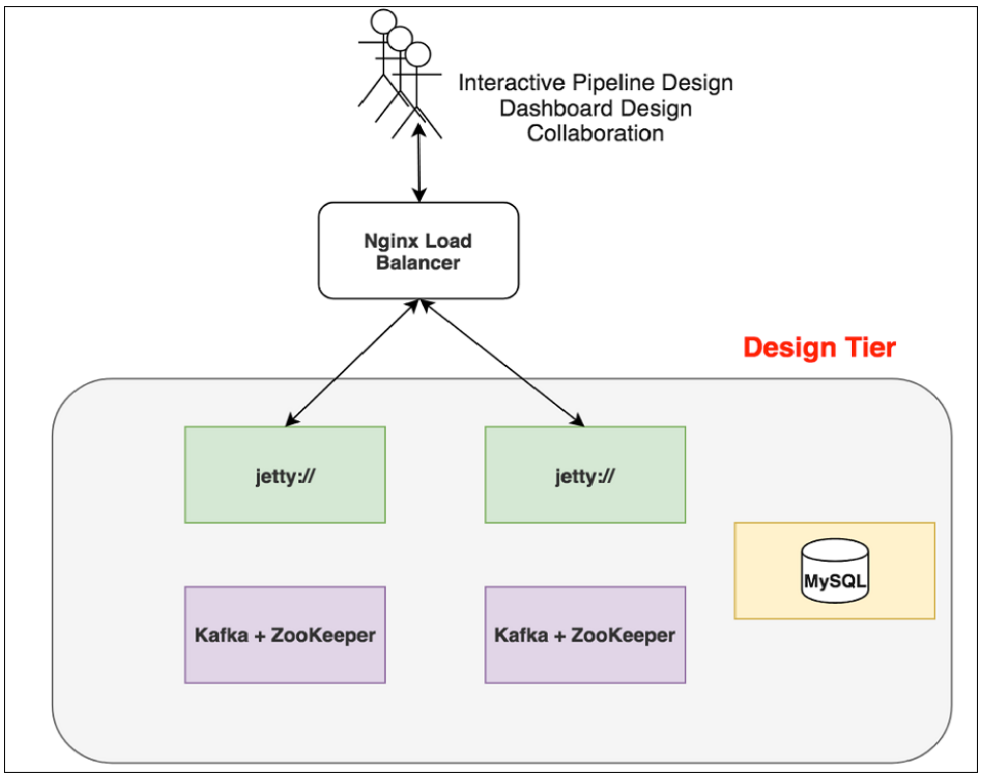
GGSA uses Jetty as the web-server with support for HA. For production deployments of GGSA Design-tier, you require the minimum hardware configuration listed below:

- Web server – Jetty with High Availability (HA) support
- 2 nodes with 4+ cores and 32+ GB of RAM for running two instances of Jetty.
- 1 node with 4+ cores and 16+ GB of RAM for running MySQL or Oracle meta-store.
- 2 nodes with 4+ cores and 16+ GB of RAM for running two instances of Kafka and 3 instances of ZooKeeper. Please note this is a separate Kafka cluster for GGSA's internal use and for interactively designing pipelines. ZooKeeper end-point of this Kafka cluster must be specified in GGSA's system settings UI.

> **Note:**
>
> The two-node Kafka cluster can be avoided if customer already has a Kafka cluster in place and is fine with OSA leveraging that cluster for its internal usage.

Based on the above estimates, total cores for design-tier is 12 and approximate memory is 112 GB RAM. Jetty instances can be independently scaled as the

number of users increase. Diagram below illustrates GGSA's Design-tier topology.



Data Tier

The deployed pipelines are run on the YARN or Spark cluster. You can use existing YARN/Spark clusters if you have sufficient spare capacity.

Sizing Guidelines

Use the following sizing guidelines to run GGSA pipelines. Ensure that the pipelines are deployed on shared storage, so that the pipeline code and libraries are accessible from all nodes in the YARN/Spark cluster. GGSA supports NFS for shared storage but if you want to use HDFS, the hardware needs two more nodes.

- – 2 nodes with 4+ cores, 16+GB RAM, and 500 GB local disk to run HDFS cluster, two instances of HDFS name and data nodes.

The Spark tier is where work happens and the Spark cluster size depends on

- Number of pipelines that will simultaneously run
- Logic in each pipeline
- Desired degree of parallelism

For each streaming pipeline the number of cores and memory gets computed based on a required degree of parallelism. As an example, consider a pipeline ingesting data from customer's Kafka topic T with 3 partitions using direct ingestion. Direct ingestion is where no Spark Receivers are used. In this case, the minimum number of processes that you need to run for optimal performance is as follows: 1 Spark Driver Process + 3 Executor processes, 1 for each Kafka Topic partition. Each Executor process needs a minimum of 2 cores.

The number of cores for a pipeline can be computed as

--executor-cores = 1 + Number of Executors * 2

In case of Receiver-based ingestion as in JMS, it is computed as

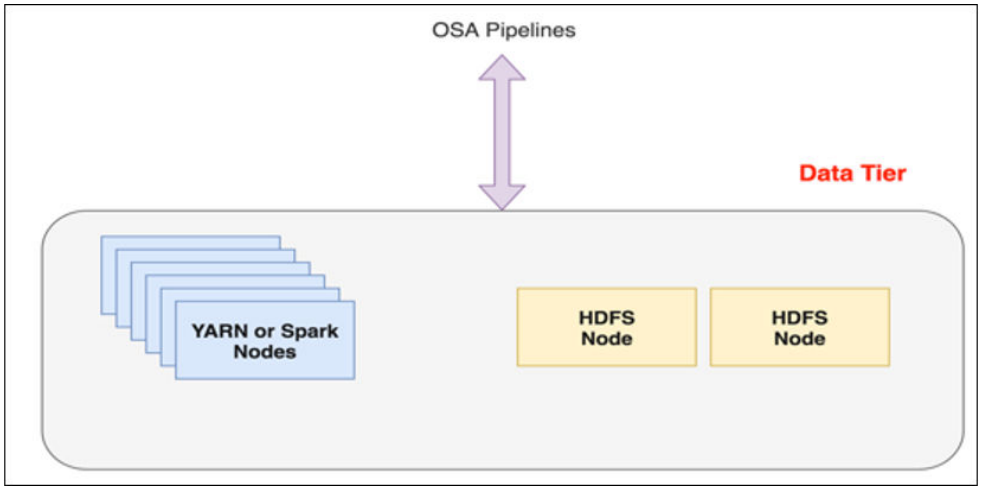
--executor-cores = 2 + Number of Executors * 2

This is rough estimates and environments where fine-grained scheduling is not available. In environments like Kubernetes, we have the luxury of more fine-grained scheduling.

The formula for sizing memory is

(Number of Windows * Average Window Range * Event Rate * Event Size) + (Number of Lookup/Reference Objects being cached * Size of Lookup Object).

Diagram below illustrates GGSA's Data-tier topology.



If you are considering GGSA for POCs and not production, then you can use the following configuration:

Design Tier

- An instance of the Jetty running on a 4+ core node with a 32+ GB of RAM.
- An instance of MySQL/Oracle for metadata store on a 4+ core node with a 16+ GB of RAM.
- A node of the Kafka cluster running on a 4+ core node with 16+ GB of RAM.

> **Note:**
>
> This is a separate Kafka cluster for GGSA's internal use and for interactively designing pipelines.

Data Tier

- A Hadoop Distributed File System (HDFS) cluster node running on 4+ core physical node with 16+ GB of RAM.
- 2 nodes of the YARN/Spark cluster each running on a 4+ core physical node with a 16+ GB of RAM.

Development Mode Configurations

Design Tier

- 1 node with 4+ cores and 16+ GB of RAM for 1 instance of Jetty, 1 instance of MySQL DB, and 1 instance of Kafka+ZooKeeper

Data Tier

- 1 node with 4+ cores and 16+ GB of RAM for 1 instance of HDFS and 1 instance of YARN/Spark.

4
Upgrading GoldenGate Stream Analytics

If you have an existing GoldenGate Stream Analytics 18.1.0.0.1 or higher installation, use the following steps to upgrade to GoldenGate Stream Analytics 19.1.0.0.6:

4.1 Upgrading GoldenGate Stream Analytics

To upgrading from an existing version of GGSA to newer version:

1. Backup your metadata store using any of Oracle or MySQL backup tools. The backup is required to restore the tools in case the upgrade fails.
2. Run the ./stop-osa.sh command to stop the GGSA server.
3. Create a **OSA-19** folder and download the `OSA-19.1.0.0.6.zip` file into the newly-created folder.
4. Unzip to extract the contents of the `OSA-19.1.0.0.6.zip` file.
5. Copy `<YourVersion>/osa-base/etc/osa-env.sh to OSA-19.1.0.0.6/osa-base/etc`.
6. Copy `<YourVersion>/osa-base/etc/jetty-osa-datasource.xml` to `OSA-19.1.0.0.6/osa-base/etc`.
7. Run the `./start-osa.sh` command to start the OSA server. This performs the schema migration.
8. To validate the upgrade, see Validating your Installation.