

**Oracle® Retail Dynamic Data Service**

Guide

Release 22.1.302.0

**F61200-01**

August 2022

Oracle Retail Dynamic Data Service Guide, Release 22.1.302.0

F61200-01

Copyright © 2022, Oracle and/or its affiliates. All rights reserved.

Primary Author: Gayathri Joshi, Sanal Parameswaran

Contributing Author: Nathan Young

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

### **Value-Added Reseller (VAR) Language**

#### **Oracle Retail VAR Applications**

The following restrictions and provisions only apply to the programs referred to in this section and licensed to you. You acknowledge that the programs may contain third party software (VAR applications) licensed to Oracle. Depending upon your product and its version number, the VAR applications may include:

- (i) the **MicroStrategy** Components developed and licensed by MicroStrategy Services Corporation (MicroStrategy) of McLean, Virginia to Oracle and imbedded in the MicroStrategy for Oracle Retail Data Warehouse and MicroStrategy for Oracle Retail Planning & Optimization applications.
- (ii) the **Wavelink** component developed and licensed by Wavelink Corporation (Wavelink) of Kirkland, Washington, to Oracle and imbedded in Oracle Retail Mobile Store Inventory Management.
- (iii) the software component known as **Access Via**<sup>™</sup> licensed by Access Via of Seattle, Washington, and imbedded in Oracle Retail Signs and Oracle Retail Labels and Tags.
- (iv) the software component known as **Adobe Flex**<sup>™</sup> licensed by Adobe Systems Incorporated of San Jose, California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

You acknowledge and confirm that Oracle grants you use of only the object code of the VAR Applications. Oracle will not deliver source code to the VAR Applications to you. Notwithstanding any other term or condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR

Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.



---

---

# Contents

|  |     |
|--|-----|
| <b>Send Us Your Comments</b> .....                                   | ix  |
| <b>Preface</b> .....   | xi  |
| Audience .....   | xi  |
| Customer Support .....   | xi  |
| Review Patch Documentation .....                                     | xi  |
| Improved Process for Oracle Retail Documentation Corrections .....   | xii |
| Oracle Help Center (docs.oracle.com) .....                           | xii |
| Conventions .....  | xii |
| <b>1 Introduction</b>  |     |
| What is DDS? .....   | 1-1 |
| Need for DDS .....   | 1-1 |
| Functionality Offered .....  | 1-1 |
| Accessibility .....  | 1-2 |
| <b>2 Installation Prerequisites</b>                                  |     |
| Installation and Setup Instructions .....                            | 2-1 |
| Prerequisites .....  | 2-1 |
| Installing WebLogic .....  | 2-1 |
| Creating the Required Schema Using Repository Creation Utility ..... | 2-5 |
| Creating a WebLogic Domain with JRF .....                            | 2-9 |
| <b>3 Deploying DDS Web Application</b>                               |     |
| Preparing the Database for DDS Installation .....                    | 3-1 |
| Deploying DDS Web Application on the WebLogic Servers .....          | 3-1 |
| Redeploying DDS Web Application .....                                | 3-4 |
| Testing the Deployment .....   | 3-4 |
| <b>4 Security</b>  |     |
| Access Levels .....  | 4-2 |
| Table Level Security .....   | 4-2 |
| Column Level Security .....  | 4-3 |
| Row Level Security .....   | 4-3 |

|                      |     |
|----------------------|-----|
| Security Setup ..... | 4-3 |
| System Options.....  | 4-4 |

## 5 RESTful Services

|                               |     |
|-------------------------------|-----|
| Discover.....                 | 5-1 |
| Get Data.....                 | 5-1 |
| Get Data from a Table.....    | 5-1 |
| Insert Data.....              | 5-2 |
| Update Data.....              | 5-2 |
| Delete Data .....             | 5-2 |
| Get Schemas .....             | 5-3 |
| Get Tables .....              | 5-3 |
| Get Columns for a Table ..... | 5-3 |
| Get System Options .....      | 5-3 |
| Create System Options.....    | 5-3 |
| Update System Options .....   | 5-3 |
| Delete System Options.....    | 5-4 |
| Reset Cache.....              | 5-4 |

## 6 Security RESTful Services

|  |     |
|--|-----|
| Get Security Groups .....                          | 6-1 |
| Get Security Group Permissions.....                | 6-1 |
| Get Access Levels .....                            | 6-1 |
| Get Security Config for a Schema.....              | 6-1 |
| Get All Security Config.....                       | 6-1 |
| Get All Table Level Securities .....               | 6-1 |
| Get Table Level Security for Table.....            | 6-2 |
| Get Table Level Security for Table and Group.....  | 6-2 |
| Get Column Level Security for Table and Group..... | 6-2 |
| Get Row Level Security for Table and Group .....   | 6-2 |
| Create Access Level.....                           | 6-2 |
| Update Access Level .....                          | 6-2 |
| Delete Access Level.....                           | 6-3 |
| Create Table Level Security .....                  | 6-3 |
| Create Table Level Securities .....                | 6-3 |
| Update Table Level Security .....                  | 6-4 |
| Update Table Level Securities.....                 | 6-4 |
| Delete Table Level Security.....                   | 6-4 |
| Delete Table Level Security by Id .....            | 6-5 |
| Delete Table Level Securities by Type .....        | 6-5 |
| Create Column Level Security .....                 | 6-5 |
| Update Column Level Security.....                  | 6-5 |
| Delete Column Level Security.....                  | 6-6 |
| Create Data Security Configuration .....           | 6-6 |
| Update Data Security Configuration.....            | 6-7 |
| Delete Data Security Configuration .....           | 6-7 |
| Create Security Group.....                         | 6-7 |

|                                     |     |
|-------------------------------------|-----|
| Delete Security Group .....         | 6-7 |
| Create Row Level Security.....      | 6-7 |
| Update Row Level Security .....     | 6-8 |
| Delete Row Level Security .....     | 6-8 |
| Create Default Security Setup ..... | 6-8 |

## 7 User Interface

|                                  |     |
|----------------------------------|-----|
| <b>Admin Tab Guide</b> .....     | 7-1 |
| Initial Permissions Setup.....   | 7-1 |
| Manage Access Levels.....        | 7-1 |
| Manage Security Groups.....      | 7-2 |
| Manage Security Permissions..... | 7-3 |
| <b>Designer Tab Guide</b> .....  | 7-5 |
| Table Definition.....            | 7-5 |
| Query Builder .....              | 7-5 |

## 8 Advanced Backend Features

|   |     |
|---|-----|
| LIKE Statement.....                               | 8-1 |
| IN Statement.....                                 | 8-2 |
| Comparative Operators .....                       | 8-2 |
| Filtering DATE Column .....                       | 8-3 |
| Combinations of OR, AND & BETWEEN Statements..... | 8-3 |
| Viewing Clob Data.....                            | 8-4 |
| Viewing Query Response as JSON Data .....         | 8-5 |
| JOINing Two or More Tables .....                  | 8-5 |





---

---

# Send Us Your Comments

Oracle Retail Dynamic Data Service Guide, Release 22.1.302.0

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

---

---

**Note:** Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the new Applications Release Online Documentation CD available on My Oracle Support and [www.oracle.com](http://www.oracle.com). It contains the most current Documentation Library plus all documents revised or released recently.

---

---

Send your comments to us using the electronic mail address: [retail-doc\\_us@oracle.com](mailto:retail-doc_us@oracle.com)

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at [www.oracle.com](http://www.oracle.com).



---

---

# Preface

The *Dynamic Data Service (DDS) Guide* describes the integration and configuration information for Oracle Retail Dynamic Data Service.

## Audience

This guide is for:

- Systems administration and operations personnel
- Systems analysts
- Integrators and implementers
- Business analysts who need information about Product processes and interfaces

## Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

<https://support.oracle.com>

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

## Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 19.0.000) or a later patch release (for example, 19.0.001). If you are installing the base release and additional patch releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch releases can contain critical information related to the base release, as well as information about code changes since the base release.

## Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

Oracle Retail product documentation is available on the following web site:

<https://docs.oracle.com/en/industries/retail/index.html>

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of a document with part number E123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

## Oracle Help Center (docs.oracle.com)

Oracle Retail product documentation is available on the following web site:

<https://docs.oracle.com/en/industries/retail/index.html>

(Data Model documents can be obtained through My Oracle Support.)

## Conventions

The following text conventions are used in this document:

| Convention      | Meaning  |
|-----------------|--|
| <b>boldface</b> | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.         |
| <i>italic</i>   | 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. |

---

---

# Introduction

This chapter gives a brief introduction to the Dynamic Data Service Web Application.

## What is DDS?

Dynamic Data Service (DDS) is a cloud based web application with functionality to remotely interact with databases. It has the ability to interact with many schemas at once in a database. The functionality also has the ability to interact with different databases as well.

Data can be viewed and modified through the application. It also has security built into it to restrict data access and modification.

Dynamic Data Service provides access to data in a database through RESTful services. Users can perform CRUD operations on the data using RESTful services. From a security perspective, access to data can be restricted to users at the table, column and row level.

Dynamic Data Service allows users to access data from any configured databases the time of install. Dynamic Data Service UI provides functionality to setup the security and perform CRUD operations on the data.

## Need for DDS

DDS is useful for exposing data in a schema with data that is not directly visible to the user. Good examples for this would be a back-end schema of other Oracle applications. Any data that is not exposed by these applications can be viewed using DDS because users cannot connect to database schemas, from a cloud environment, using standard database tools without proper permissions.

## Functionality Offered

DDS offers various functionality regarding databases. It can connect to various database connections. The application then discovers the schema inside the databases and allows the user to work and interact with the tables and data within the tables. The high level list of functionalities offered:

- Schema Discovery
- Viewing Table Definition
- Querying table data
- Creation, modification and deletion of records
- Bulk Updates and Deletion of data in tables

- Security setup for limiting data access and modification

## Accessibility

Oracle JET components have built-in accessibility support that conforms with the Web Content Accessibility Guidelines version 2.0 at the AA level (WCAG 2.0 AA), developed by the World Wide Web Consortium (W3C).

Accessibility involves making your application usable for persons with disabilities such as low vision or blindness, deafness, or other physical limitations. This means, for example, creating applications that can be:

- Used without a mouse (keyboard only).
- Used with assistive technologies such as screen readers and screen magnifiers.
- Used without reliance on sound, color, animation, or timing.

DDS provides the ability to support the above accessibility in the applications.

Users should be able to navigate to all parts and functions of the application using the Tab and arrow keys, without using any keyboard shortcuts. In addition to that, keyboard shortcuts merely provide an additional way to access a function quickly.

Keyboard shortcuts provide an alternative to pointing devices for navigating the page. There are five types of keyboard shortcuts that can be provided in OJET applications:

- Tab traversal, using Tab and Shift+Tab keys: Moves the focus through UI elements on a screen.
- Accelerator keys (hot keys): bypasses menu and page navigation, and performs an action directly, for example, Ctrl+C for Copy.
- Access keys: Moves the focus to a specific UI element, for example, Alt+F for the File menu.
- Default cursor/focus placement: Puts the initial focus on a component so that keyboard users can start interacting with the page without excessive navigation.
- Enter key: Triggers an action when the cursor is in certain fields or when the focus is on a link or button.

---

---

## Installation Prerequisites

This chapter describes the procedure to install the WebLogic 12c runtime and deploy the tool's EAR file. For more information about domain creation and other server related information, see the WebLogic application server documents.

### Installation and Setup Instructions

This section describes the installation and setup instructions including the installation pre-requisites, preparing the WebLogic server, creating a WebLogic domain, verifying installation of JRF runtime libraries and deploying the EAR file. It also describes guidelines to set up security.

---

---

**Note:** The windows included in the following procedures are for example purposes only. Because these procedures must be followed for each application, valid values vary. Therefore, consider the illustrations as guides only; the values shown may not always apply.

---

---

### Prerequisites

DDS Web Application requires Oracle WebLogic Server 12c (12.2.1.4.0), built with Java 8 (JDK 1.8 64 bit with the latest security updates).

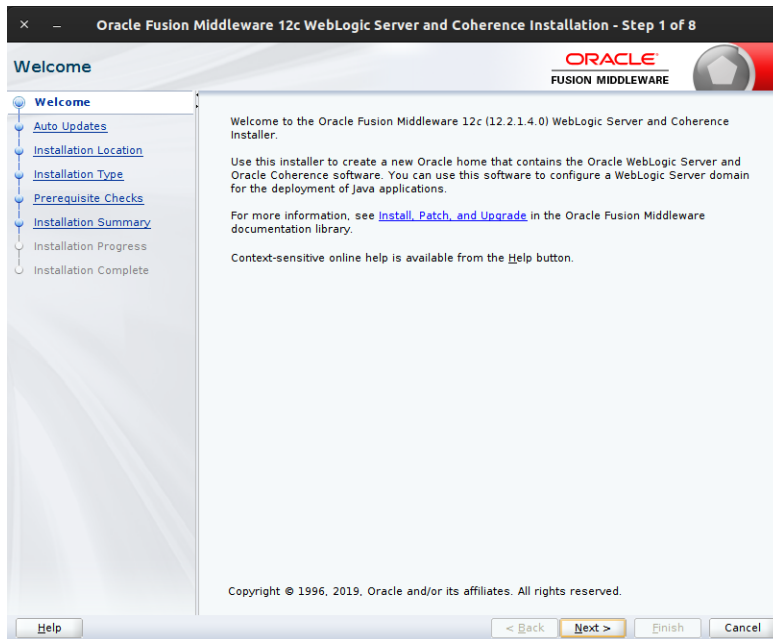
### Installing WebLogic

To get the JRF runtime option while creating the domain, install the Application Development Runtime. To obtain Application Development Runtime, go to the Oracle Technology Network and take the following steps:

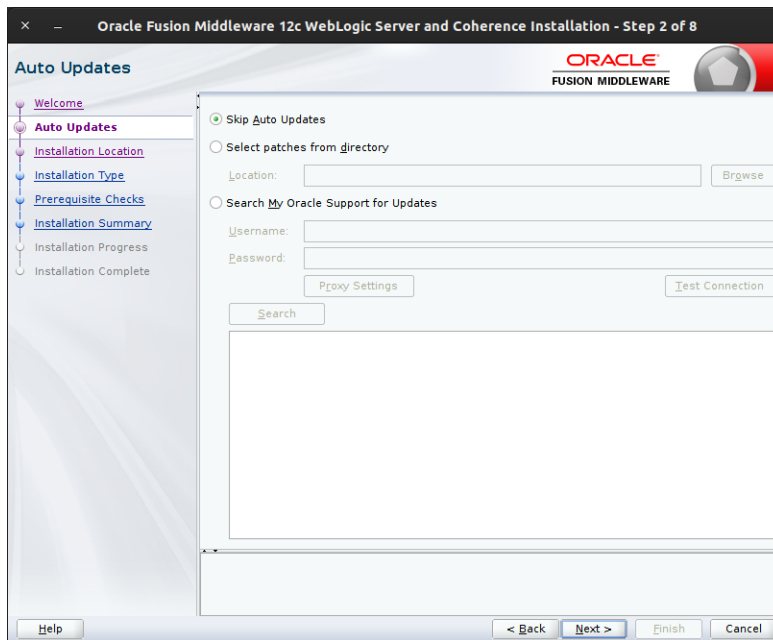
1. Find `fmw_12.2.1.4.0_infrastructure_Disk1_1of1.zip` and download this file to your system.
2. Extract the contents of this zip file to your system. You will use the `fmw_12.2.1.4.0_infrastructure.jar` file to run the installer.
3. Run the installer by executing the jar file:  

```
java -jar fmw_12.2.1.4.0_infrastructure.jar
```

The Welcome window displays.

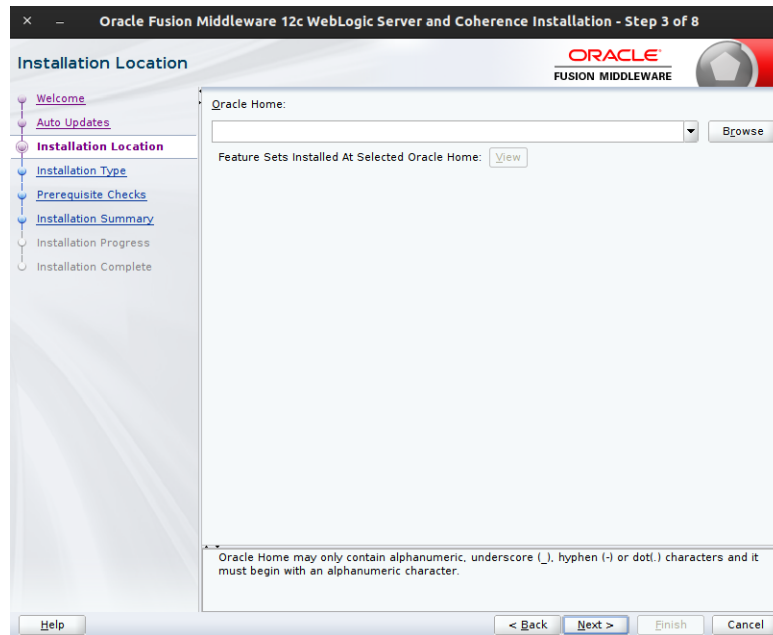


4. Click **Next**. The Auto Updates window displays. Select the appropriate option.

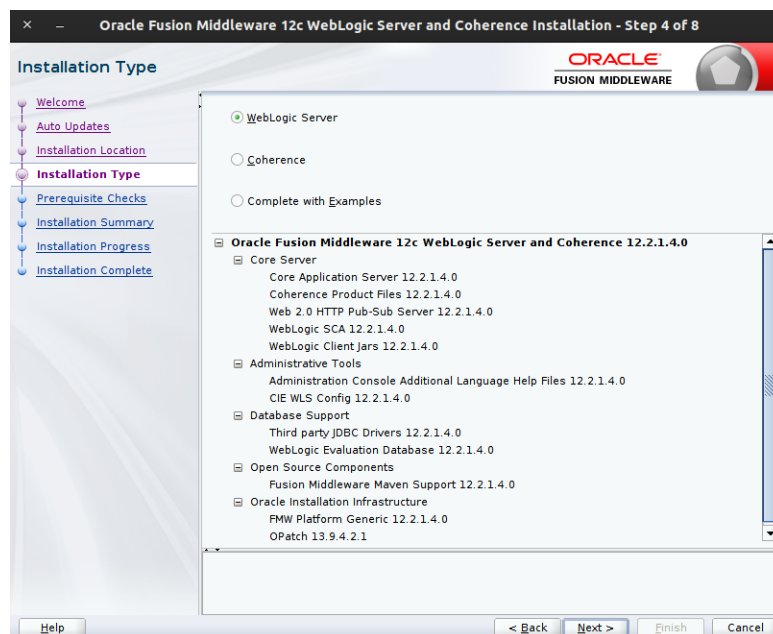


5. Click **Next**. The Installation Location window displays. Click **Browse** to select the Oracle Home location where the WebLogic Server is to be installed.

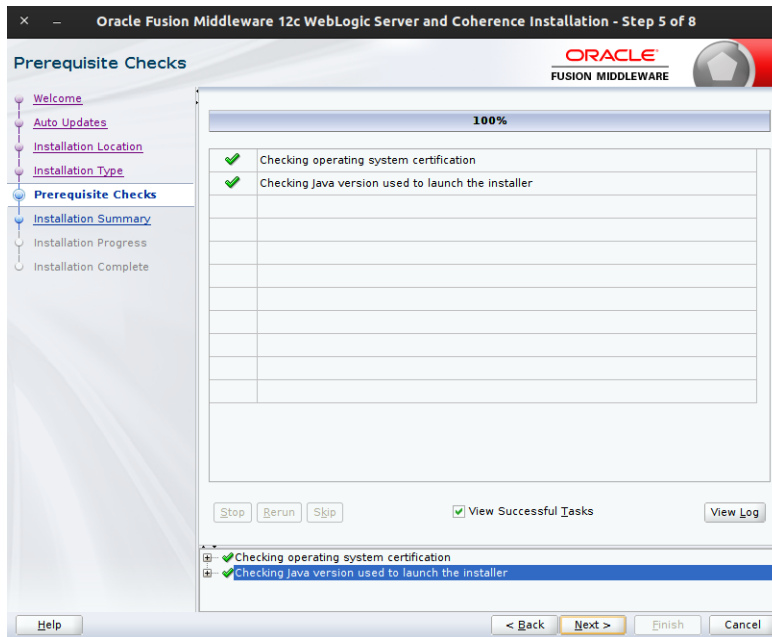




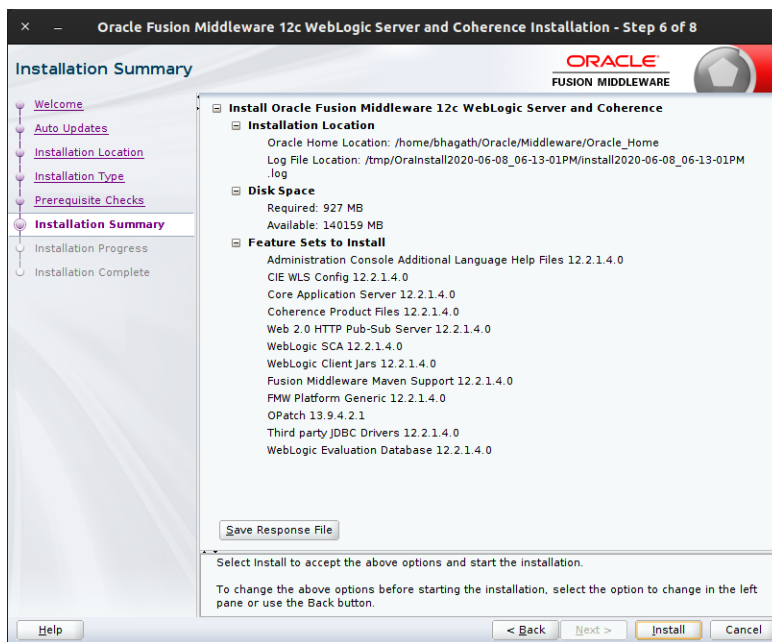
6. Click **Next**. The Installation Type window displays. Select the type of installation.



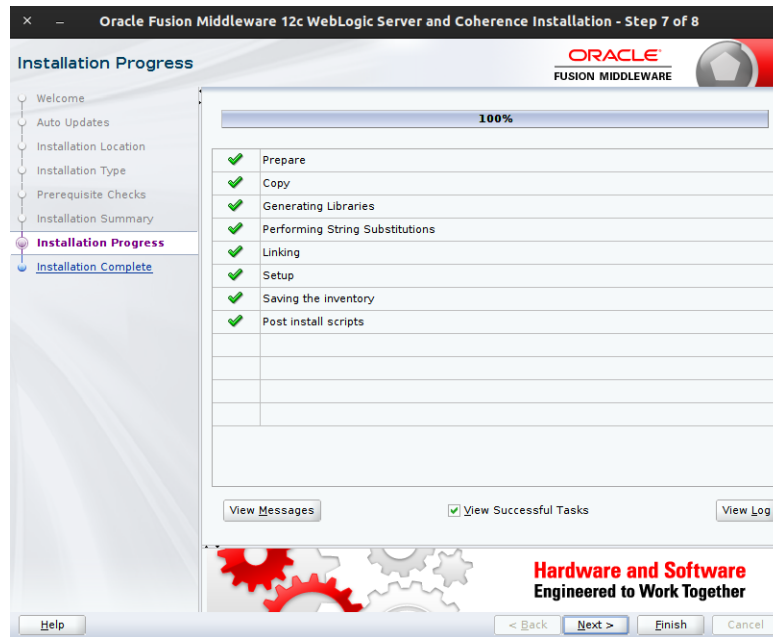
7. Click **Next**. The installer performs the pre-requisite checks and ensures all required conditions are satisfied.



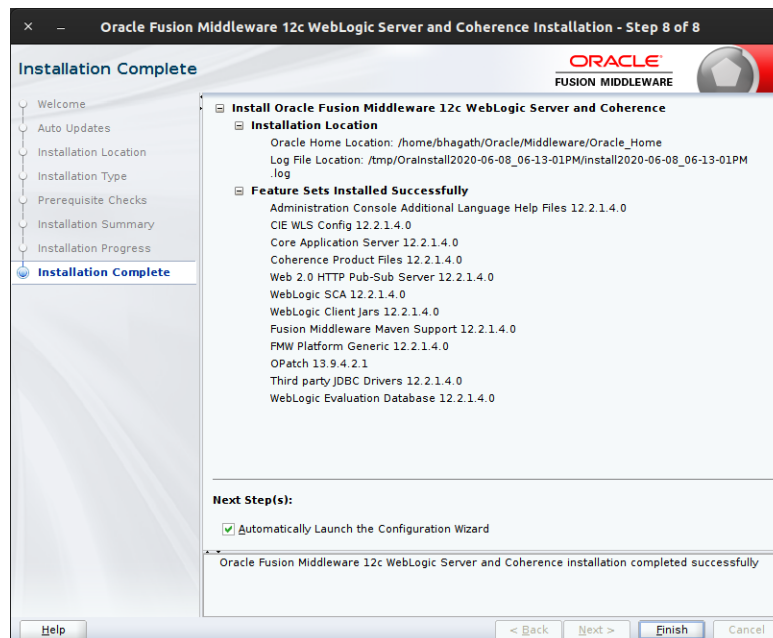
8. When the pre-requisite check completes successfully, click **Next**. The Security Updates window will display. Enter the information as required.
9. Click **Next**. The Installation Summary window displays.



10. Click **Install**. The Installation Progress window displays.



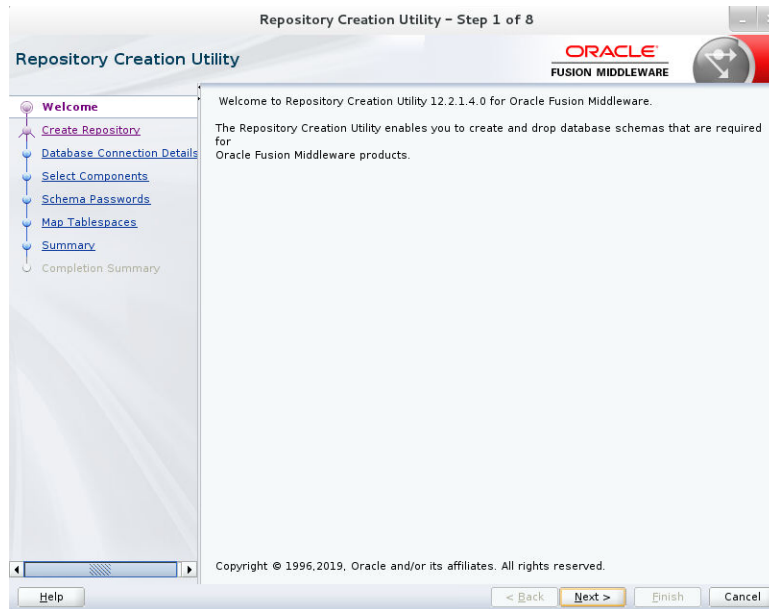
11. Click **Next** when the installation completes. The Installation Complete window displays.



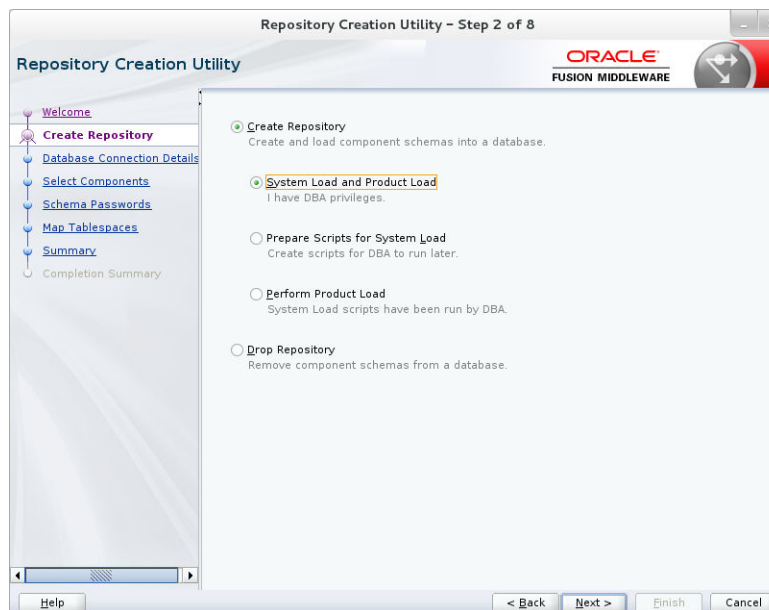
## Creating the Required Schema Using Repository Creation Utility

To create a schema user for the dynamic\_data\_service domain, take the following steps:

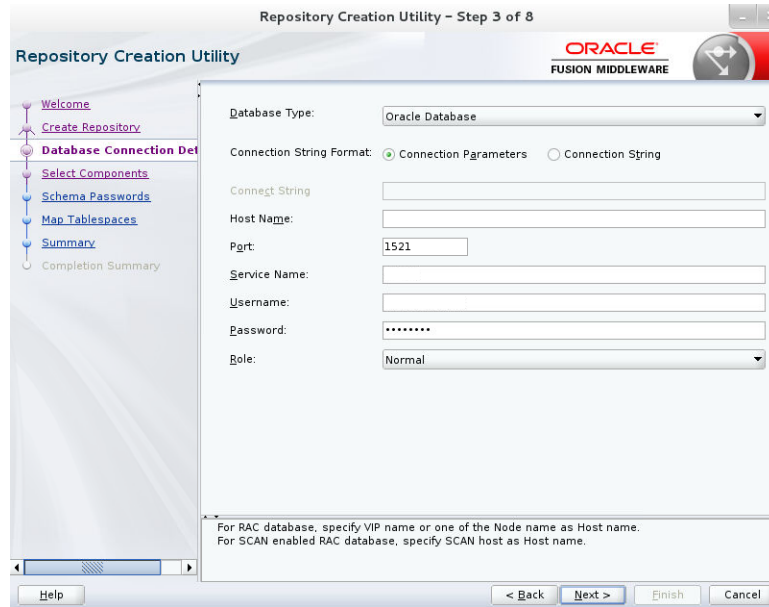
1. Run the RCU from the <MW\_HOME>/oracle\_common/bin folder. The Welcome window displays.



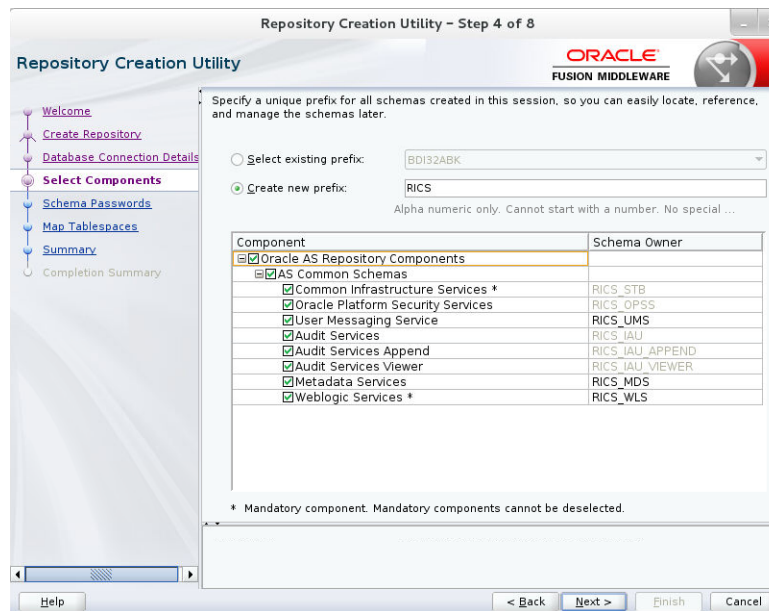
2. Click **Next** and select the **Create Repository** option.



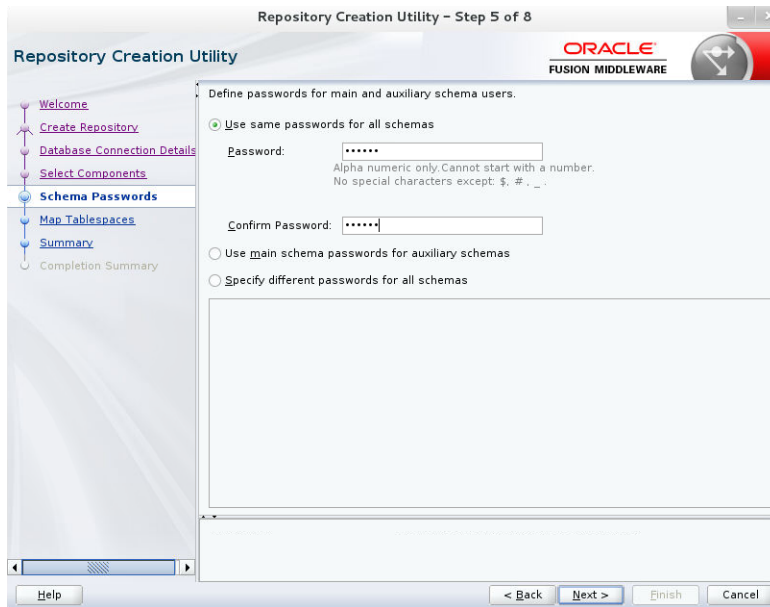
3. Click **Next**. Enter the database credentials where the schema user has to be created.



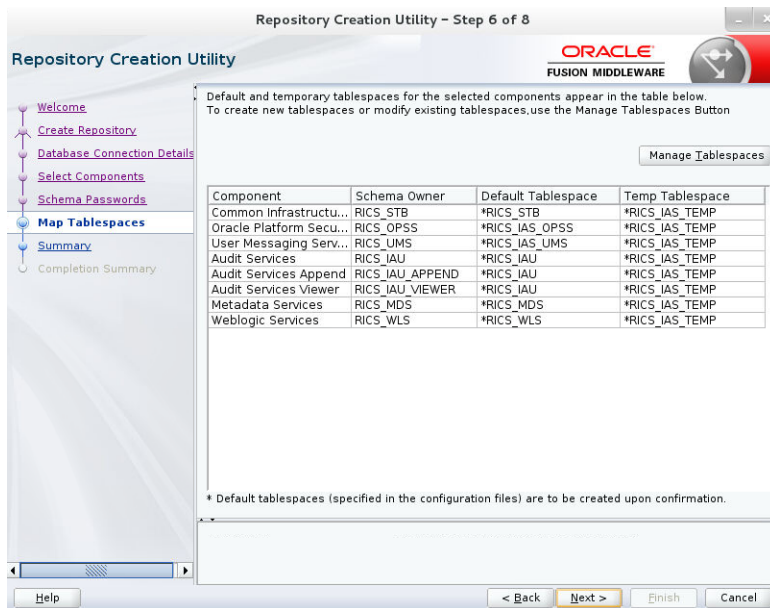
4. Click **Next**. Specify the prefix to be used for the schema user creation. For example, INT. Select Metadata Services, WebLogic Services, and Oracle Platform Security Services.



5. Click **Next**. Specify the password.



6. Click **Next**. This window provides the details for tablespaces created as part of schema creation.



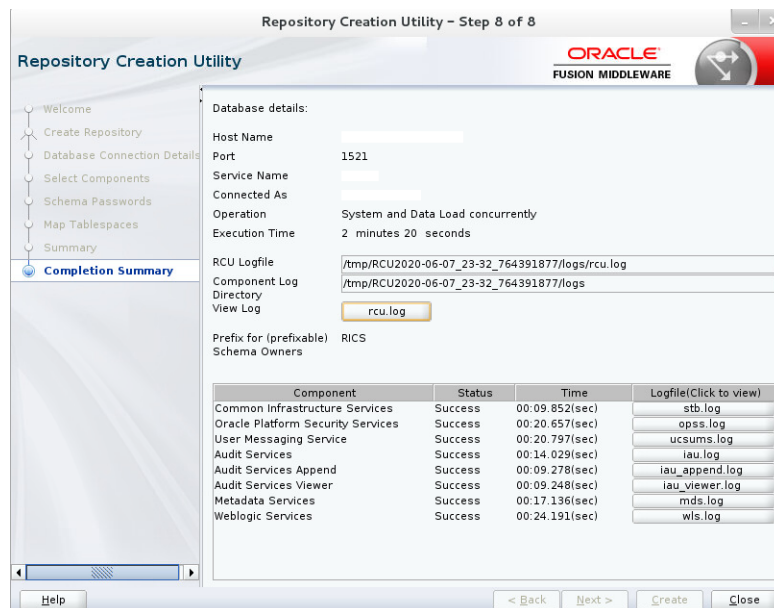
7. Click **Next**. The Confirmation window displays.



8. Click **OK**. The Summary window displays.



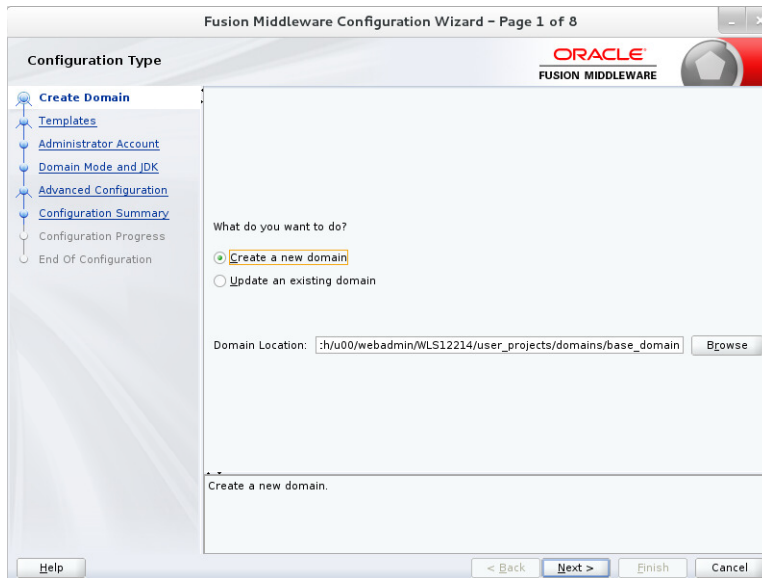
9. Click **Create** to create the schema. This could take a while to complete. When complete the Summary window displays.



## Creating a WebLogic Domain with JRF

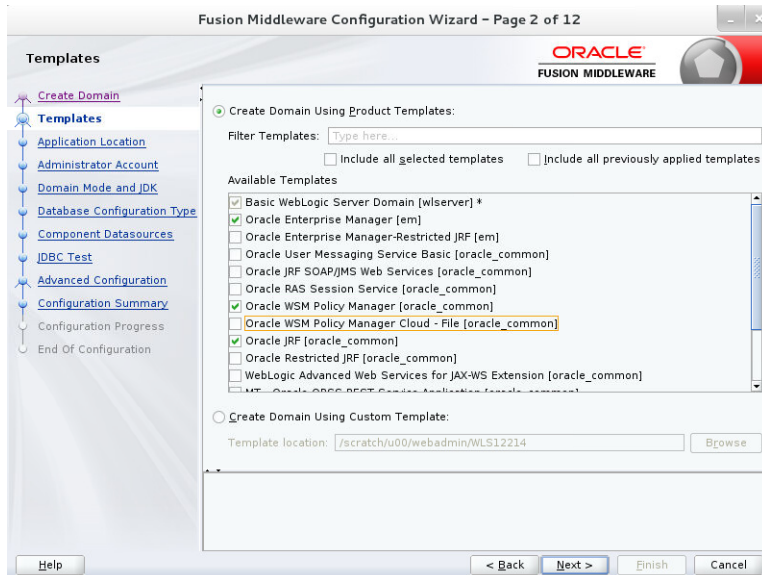
To create a new WebLogic domain with ADF runtime libraries, take the following steps:

1. Run the config.sh from the <ORACLE\_HOME>/oracle\_common/common/bin folder. The Configuration Type window displays.



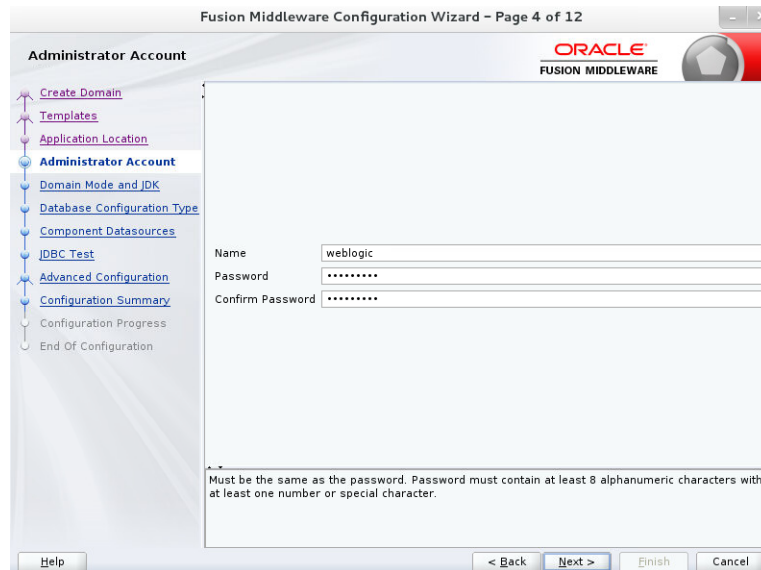
2. Select Create a new domain, and provide a domain location, then click Next. The Templates window displays. By default, the Basic WebLogic Server Domain - 12.2.1.0 [wlserver] check box is selected.

Select the Oracle JRF - 12.2.1.4.0 [oracle\_common], Oracle Enterprise Manager - 12.2.1.4.0 [em], Oracle WSM Policy Manager - 12.2.1.4.0 [oracle\_common], and WebLogic Coherence Cluster Extension - 12.2.1.4.0[wlserver] check boxes.

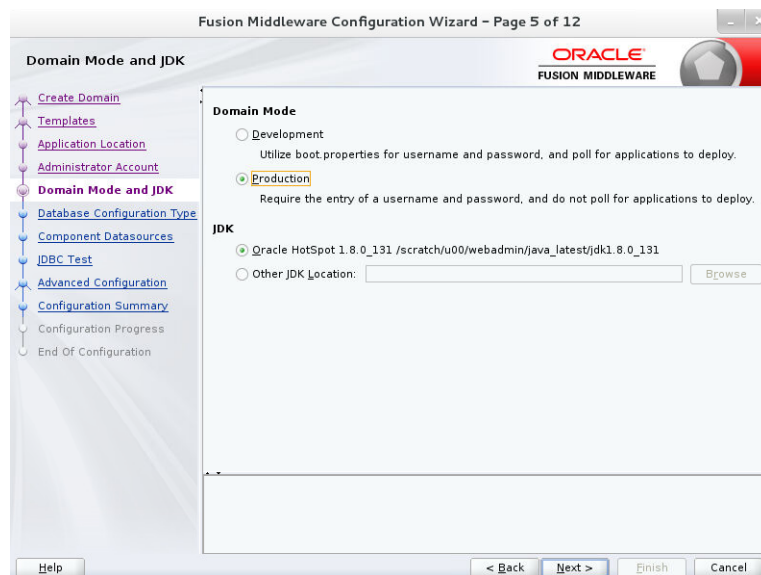


3. Click Next. The Administrator Account window displays. Enter the user credentials that will be used to log into the WebLogic Administration Console.



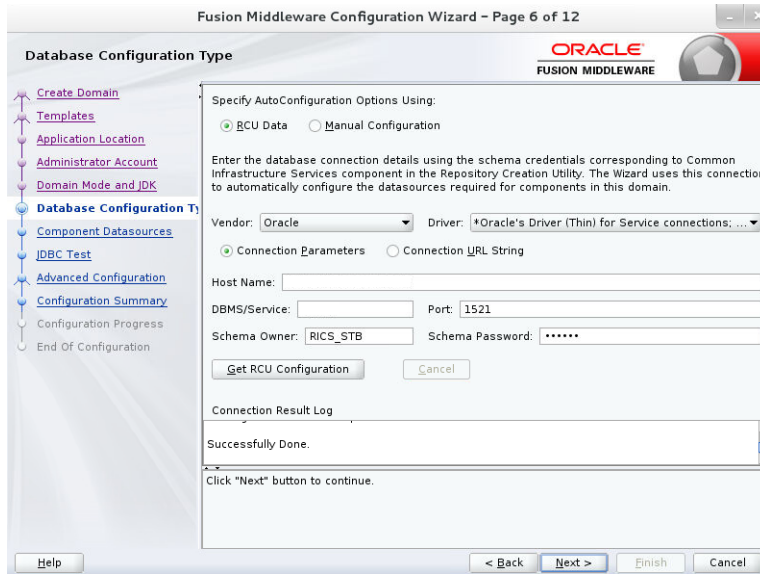


4. Click **Next**. The Domain Mode and JDK window displays. Set the Domain Mode as Production and select the JDK version (JDK 1.8 with the latest security updates is recommended) you want to use.

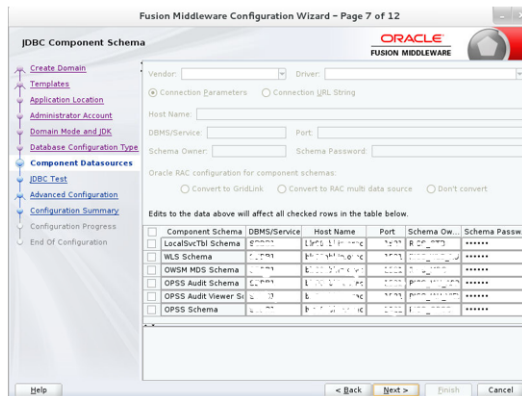


5. Click **Next**. The Database Configuration Type window displays.
  - a. Select the RCU Data radio button.
  - b. Select Oracle as the Vendor.
  - c. Select Oracle's Driver (Thin) for Service connections; Version 9.0.1 and later.
  - d. Enter the Service, Host Name, Port, Schema Owner, and Schema Password for the \*\_STB schema created using the RCU.
  - e. Click Get RCU Configuration.

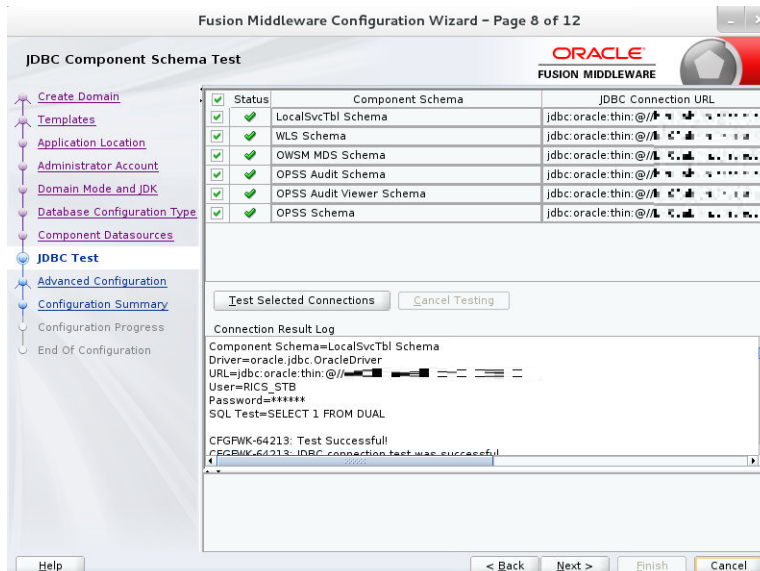
The Connection Result Log displays the connection status.



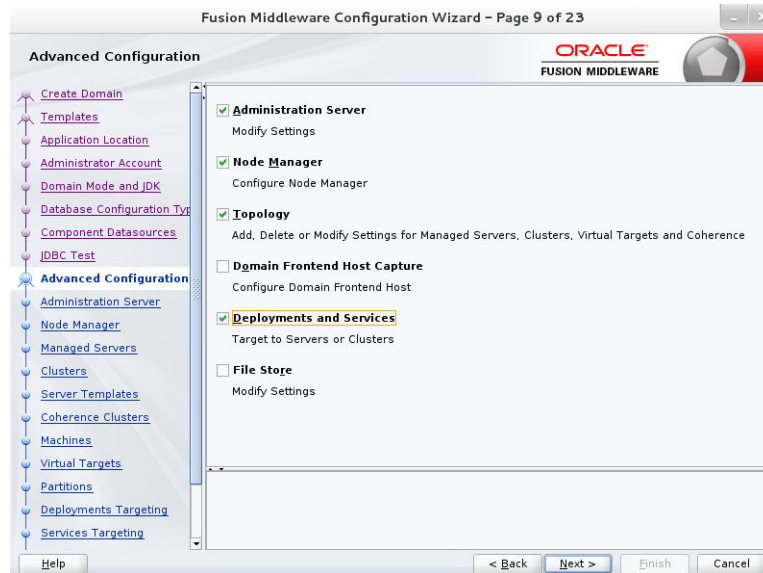
6. Click Next. The JDBC Component Schema window displays.



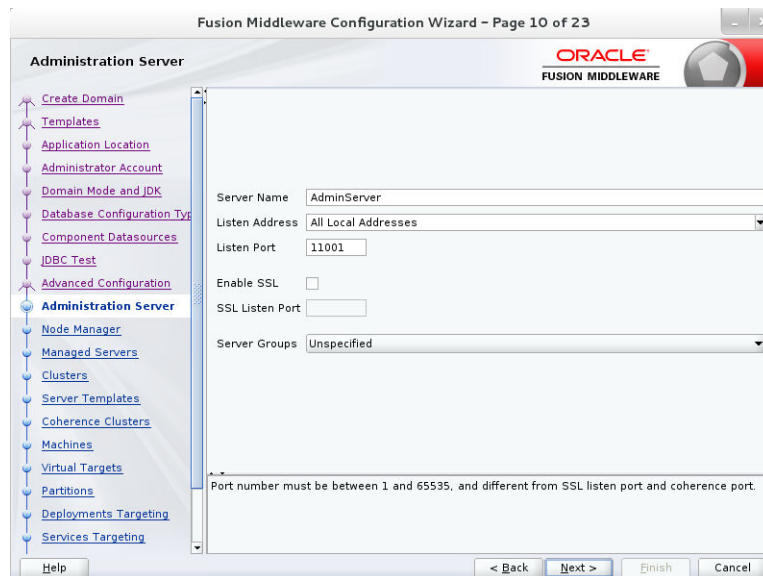
7. Click Next. The JDBC Component Schema Test window displays a status indicating the JDBC tests on the schemas were successful.



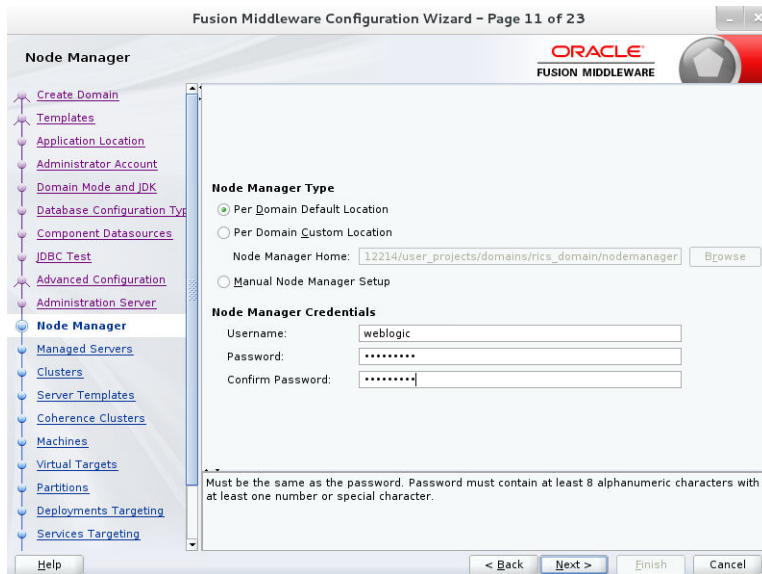
8. Click **Next**. The Advanced Configuration window displays. Select all the checkboxes, except Domain Frontend Host Capture and JMS File Store options, in this window.



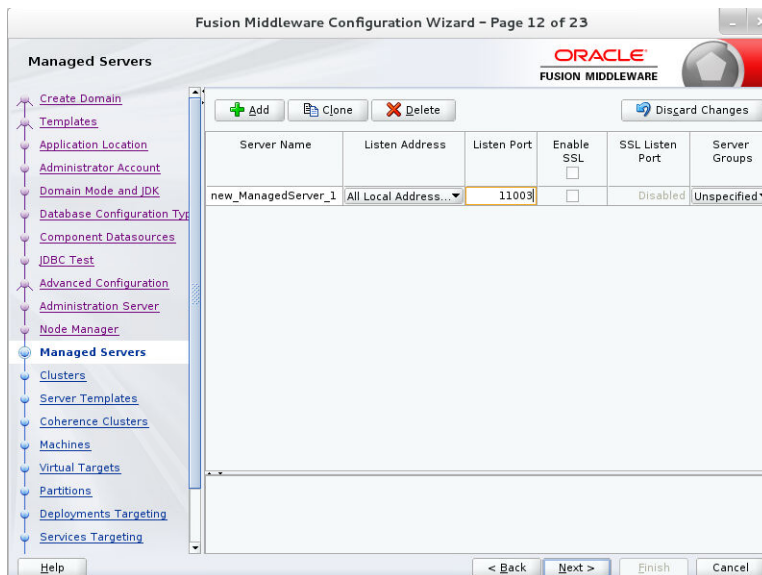
9. Click **Next**. The Administration Server window displays. Enter the Listen Address and the Listen Port details.



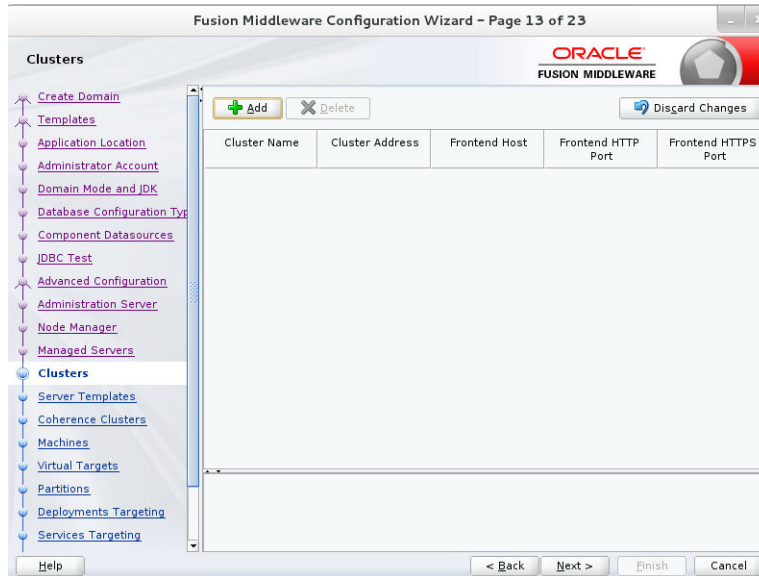
10. Click **Next**. The Node Manager window displays. Select the Node Manager Type and enter the Node Manager Credentials.



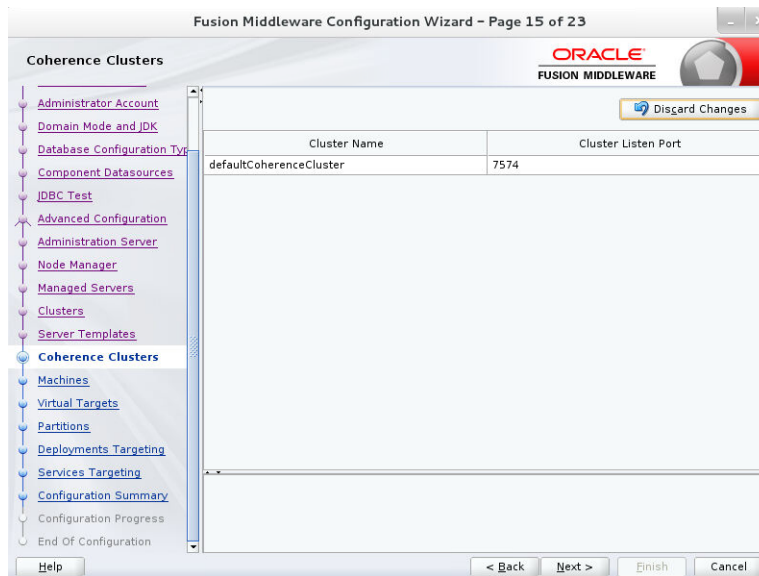
11. Click **Next**. The Managed Servers window displays.
  - a. Click **Add** to add a managed server to deploy the DDS Web Application.
  - b. Enter the Server Name, Listen Address, and Listen Port for the managed server.
  - c. Set the Server Groups to `JRF_MAN_SRV`.



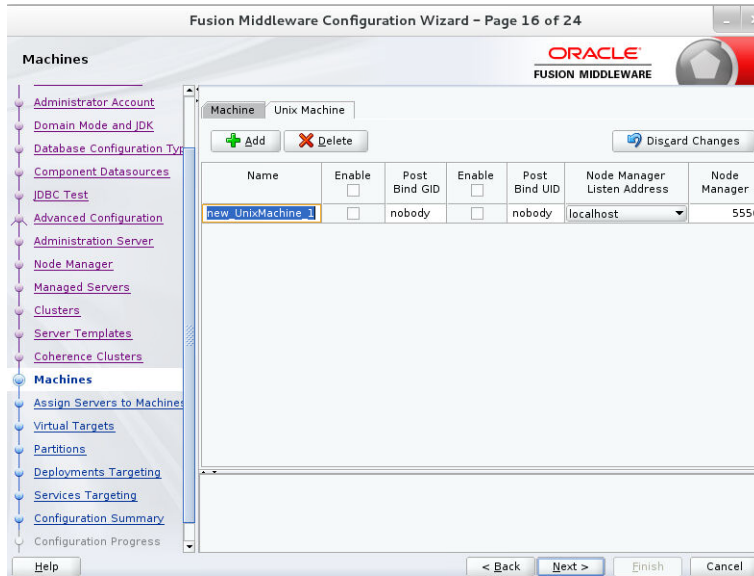
12. Click **Next**. The Clusters window displays.
  - a. Click **Add** to add a cluster. This is an optional step in the procedure.



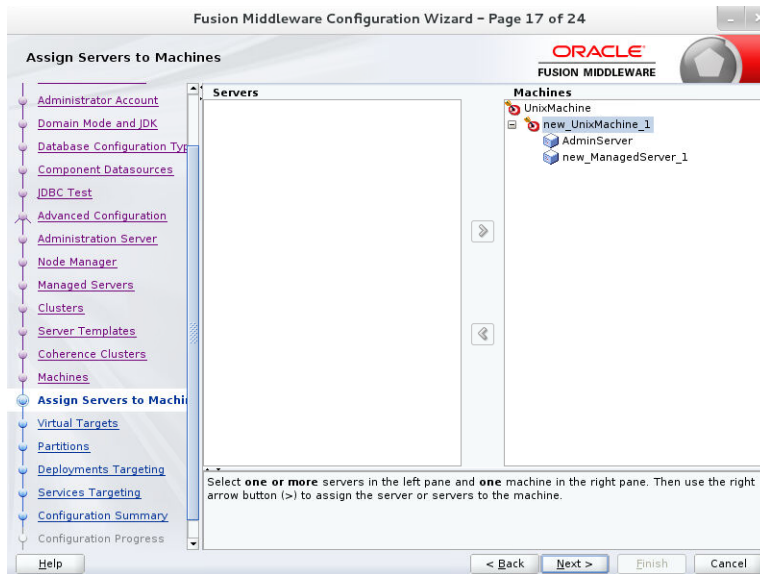
13. Click **Next**. The Coherence Clusters window displays.
  - a. Add a coherence cluster. This is an optional step in the procedure.



14. Click **Next**. The Machines window displays.
  - a. Click **Add**.
  - b. Enter the Name and the Node Manager Listen Address for the managed server.

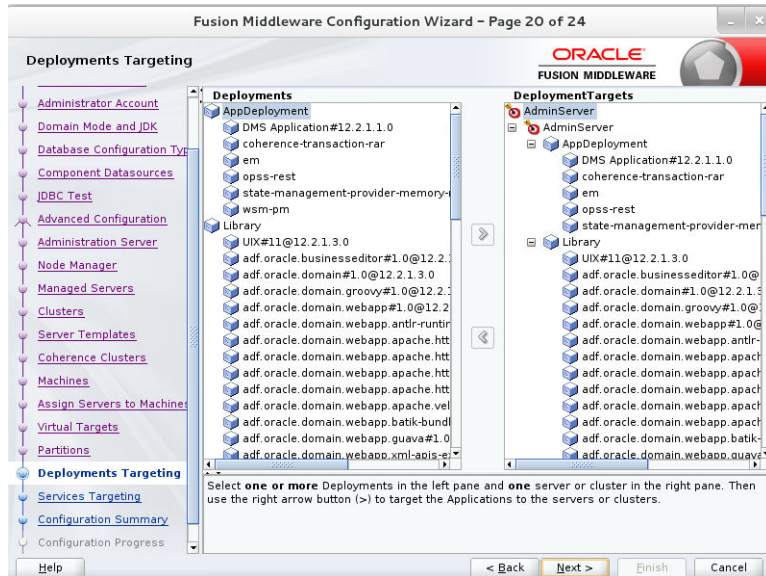


15. Click **Next**. The Assign Servers to Machines window displays. Add the AdminServer and the Managed Server to the machine.

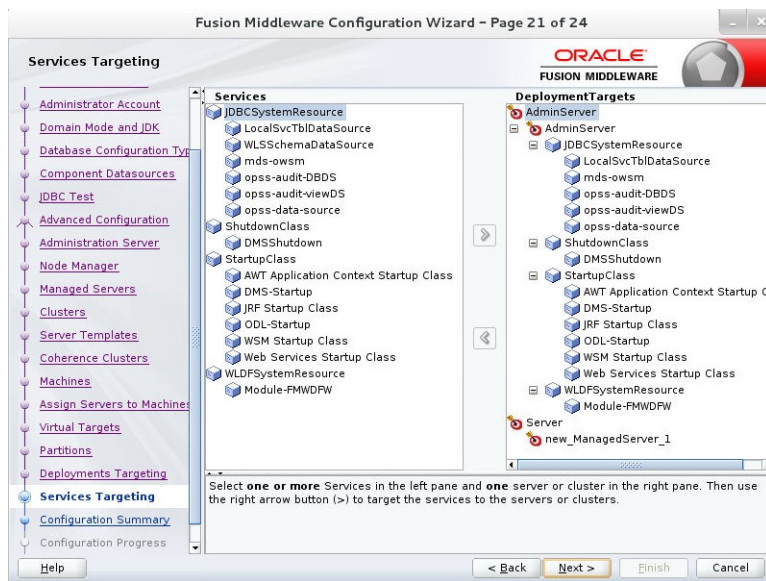


16. Click **Next**. The Deployments Targeting window displays. Select wsm-pm from the Deployments section on the left and add it to the AdminServer in the Deployment Targets section on the right.

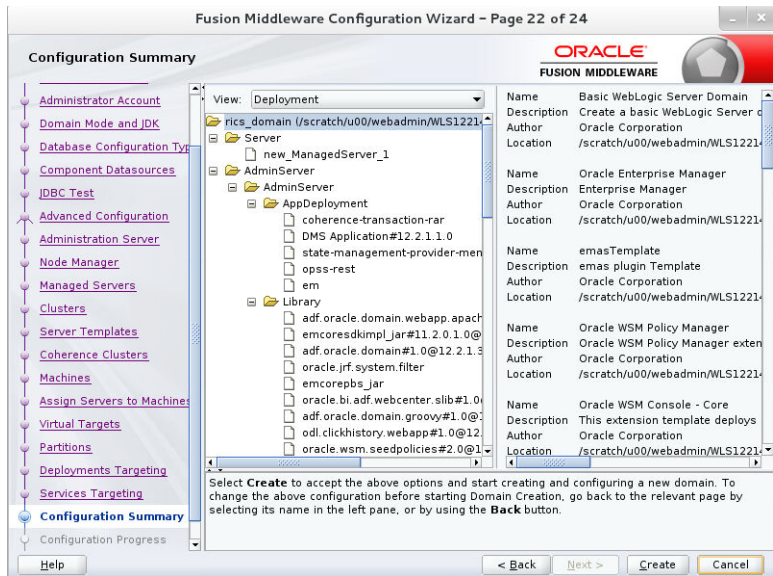




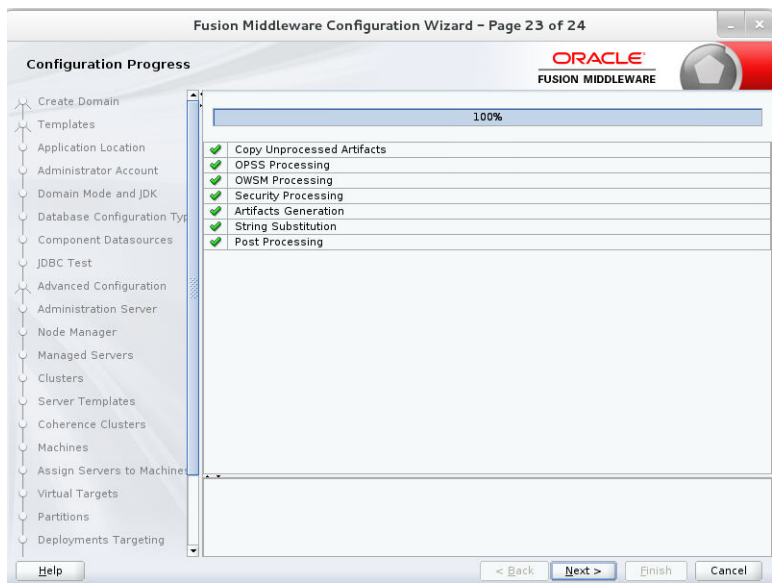
17. Click **Next**. The Services Targeting window displays. Target JDBC services to the Admin and Managed server.



18. Click **Next**. The Configuration Summary window displays. Verify that all information displayed in this window is accurate.

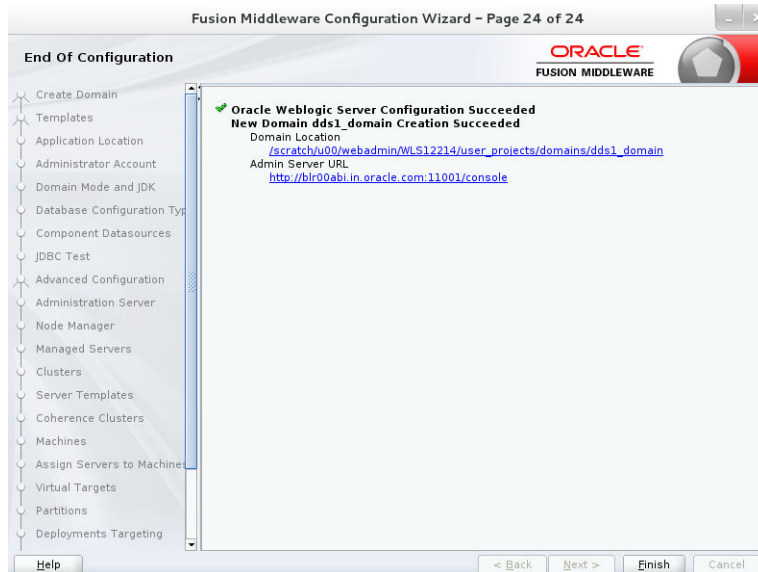


19. Click **Create**. The Configuration Progress window will display a message when the domain is created successfully.



20. Click **Next**. The Configuration Success window displays that describes the Domain Location and Admin Server URL once the configuration is complete.





21. Click **Finish** to complete creating the WebLogic domain and managed servers with ADF runtime.
22. Add the following security policy to \$ORACLE\_HOME/wlserver/server/lib/weblogic.policy file.
 

```
grant codeBase "file:/<DOMAIN_HOME>/-" {
  permission java.security.AllPermission;
  permission oracle.security.jps.service.credstore.CredentialAccessPermission
    "credstoressp.credstore", "read,write,update,delete";
  permission oracle.security.jps.service.credstore.CredentialAccessPermission
    "credstoressp.credstore.*", "read,write,update,delete";
};
```
23. Start the WebLogic Admin and Managed Server.



---



---

## Deploying DDS Web Application

This chapter describes the steps you should take to deploy DDS Web Application.

### Preparing the Database for DDS Installation

Before you begin installing the DDS Web Application, make sure you have a valid DDS Data Source available. DDS assumes that the tables created, in the DDS data schema configured, are valid.

### Deploying DDS Web Application on the WebLogic Servers

To deploy the DDS Web Application ear, take the following steps:

1. Download DynamicDataService19.1.000ForAll19.x.xApps\_eng\_ga.zip
2. Unpack the folder. A folder dds\_home will be available.
3. Edit the dds-deployment-env-info.json as follows:

```
cd dds_home/conf/
vi dds-deployment-env-info.json
```

4. Modify the DataSourceDef and MiddlewareServerDef with information that is specific to your environment.

By default, the JSON files should have placeholders for the DDS, RIB Error Hospital, OCDS and RFI connections, namely DDSDataSource, RibErrorHospitalDataSource, OCDSDataSource and RFIDDataSource. If you plan on using either of these connections, delete the other connections that are not in use.

The table below summarizes the values that need to be changed specific to the environment

**Table 3-1** *DataSourceDef and MiddlewareServerDef Values*

| Value                                 | Description  |
|---------------------------------------|--|
| DDSDDataSource -> jdbcUrl             | Database details of the server where the DDS default data source schema is hosted.                   |
| RibErrorHospitalDataSource -> jdbcUrl | Database details of the server where the database schema for the RIB Error hospital table is hosted. |
| OCDSDataSource -> jdbcUrl             | Database details of the server where the database schema for the OCDS application is hosted.         |
| RFIDDataSource -> jdbcUrl             | Database details of the server where the database schema for the RFI application is hosted.          |

**Table 3–1 (Cont.) DataSourceDef and MiddlewareServerDef Values**

| Value  | Description   |
|--|---|
| DDSAppServer -><br>weblogicDomainName                        | Name of the domain where the DDS application will be deployed.                                    |
| DDSAppServer -><br>weblogicDomainHome                        | Absolute path to the domain. (starts from the root directory)                                     |
| DDSAppServer -><br>weblogicDomainAdminServerUrl              | Admin server URL link of the domain.  |
| DDSAppServer -><br>weblogicDomainAdminServerProtocol         | Web Protocol to be used in the domain. (Can be t3, unsecure or t3s, secure)                       |
| DDSAppServer -><br>weblogicDomainAdminServerHost             | Admin server host name.<br>(domain.example.name.com)  |
| DDSAppServer -><br>weblogicDomainAdminServerPort             | Admin server host port number   |
| DDSAppServer -><br>weblogicDomainTargetManagedServer<br>Name | Name of the managed server where DDS will be deployed.  |
| DDSAppServer -> DDSAdminUiUrl                                | The complete URL link to the deployed DDS application. (http://<host_name>:<managed_sever_port>/) |

---



---

**Note:** The alias names in the configuration files should not be changed.

---



---

The following is an example configuration:

```
"DataSourceDef": {
  "DDSDataSource": {
    "dataSourceName": "DDSDataSource",
    "dataSourceClass": "oracle.jdbc.pool.OracleDataSource",
    "dataSourceJndiName": "jdbc/DDSDataSource",
    "jdbcUrl": "jdbc:oracle:thin:@//dbhost.example.com:1522/pdborcl",
    "jdbcUserAlias": "DDSAdminDataSourceUserAlias",
    "jdbcUser": "GET_FROM_WALLET",
    "jdbcPassword": "GET_FROM_WALLET",
  },

  "RibErrorHospitalDataSource": {
    "dataSourceName": "RibErrorHospitalDataSource",
    "dataSourceClass": "oracle.jdbc.pool.OracleDataSource",
    "dataSourceJndiName": "jdbc/RibErrorHospitalDataSource",
    "jdbcUrl": "jdbc:oracle:thin:@//dbhost.example.com:1522/pdborcl",
    "jdbcUserAlias": "RibErrorHospitalDataSourceUserAlias",
    "jdbcUser": "GET_FROM_WALLET",
    "jdbcPassword": "GET_FROM_WALLET",
  },

  "OCDSDataSource": {
    "dataSourceName": "OCDSDataSource",
    "dataSourceClass": "oracle.jdbc.pool.OracleDataSource",
    "dataSourceJndiName": "jdbc/OCDSDataSource",
    "jdbcUrl": "jdbc:oracle:thin:@//dbhost.example.com:1522/pdborcl",
    "jdbcUserAlias": "OCDSDataSourceUserAlias",
```

```

"jdbcUser": "GET_FROM_WALLET",
"jdbcPassword": "GET_FROM_WALLET",
},

"RFIDDataSource": {
"dataSourceName": "RFIDDataSource",
"dataSourceClass": "oracle.jdbc.pool.OracleDataSource",
"dataSourceJndiName": "jdbc/RFIDDataSource",
"jdbcUrl": "jdbc:oracle:thin:@//dbhost.example.com:1522/pdborcl",
"jdbcUserAlias": "RFIDDataSourceUserAlias",
"jdbcUser": "GET_FROM_WALLET",
"jdbcPassword": "GET_FROM_WALLET",
}
}
"MiddlewareServerDef": {
"DDSAppServer": {
"weblogicDomainName": "dds_domain",
"weblogicDomainHome": "home/oracle/middleware_1221/user_projects/domains/dds_
domain",
"weblogicDomainAdminServerUrl": "t3://ddsdomainhost.example.com:7001",
"weblogicDomainAdminServerProtocol": "t3",
"weblogicDomainAdminServerHost": " ddsdomainhost.example.com ",
"weblogicDomainAdminServerPort": "7001",
"weblogicDomainAdminServerUserAlias": "DDSAdminServerUserAlias",
"weblogicDomainTargetManagedServerName": "dds_managed_server",

"DDSAdminUiUrl": "http://ddsdomainhost.example.com:7003",
"DDSAdminUiUserGroup": "DdsAdminGroup",
"DDSAdminUiUserAlias": "DDSAdminUiUserAlias",
"DDSAdminUiUser": "GET_FROM_WALLET",
"DDSAdminUiPassword": "GET_FROM_WALLET",

"DDSOperatorUiUserGroup": "DdsOperatorGroup",
"DDSOperatorUiUserAlias": "DDSOperatorUiUserAlias",
"DDSOperatorUiUser": "GET_FROM_WALLET",
"DDSOperatorUiPassword": "GET_FROM_WALLET",

"DDSMonitorUiUserGroup": "DdsMonitorGroup",
"DDSMonitorUiUserAlias": "DDSMonitorUiUserAlias",
"DDSMonitorUiUser": "GET_FROM_WALLET",
"DDSMonitorUiPassword": "GET_FROM_WALLET"

}
} ,
"DDSAdminApplication": {
"DDSAdminAppUses": [
"DDSAppServer"
]
}
}

```

5. Run the deployer script to create the datasource and deploy the DDS Web Application.

```

$cd dds_home/bin/
$ssh dds-deployer.sh -setup-credentials -deploy-dds-app

```

6. Enter the parameter value that is prompted by the script.
7. Bounce the WebLogic Server hosting the DDS Web Application.
8. Restrict Access to the DDS home folder:

```

$cd ..

```

```
$chmod -R 700 dds-home
```

---

---

**Note:** To deploy the DDS application on a clustered environment, change the following in the config JSON file:

1. `weblogicDomainTargetManagedServerName` - give the cluster name as the name of the managed server here.
  2. `DDSAdminUiUrl` - give the port number of the managed server which have been linked to the cluster on which DDS is to be deployed.
- 
- 

## Redeploying DDS Web Application

If you have already configured the credentials and can use the same credentials (typically when redeploying the app), you can run the deployer with the `-use-existing-credentials` option as follows, and you will not be prompted for the credentials again for the deployment.

```
sh dds-deployer.sh -use-existing-credentials -deploy-job-admin-app
```

## Testing the Deployment

After you deploy the server successfully, the DDS Web Application can be accessed using the following URL:

<http://<host-server>:<managed-server-port>/dynamic-data-service-ui>

Dynamic Data Service provides table level, row level, and column level security. All the end points are protected with basic authentication and role based authorization.

There are three security groups that provide role based authorization.

There are three Roles and three Groups.

#### Roles:

- **AdminRole** - Users with this role have access to all the functions of the DDS app. They can also setup the security permissions for other users.
- **OperatorRole** - Users with this role have the ability to read, write and modify content in the schemas and tables. However they will not have access to the admin functions and cannot setup security permissions.
- **MonitorRole** - Users with this role can only read the data from schemas and tables. They also will not have access to security setup functions.

#### Groups:

- **DdsAdminGroup** - Users that belong to this group can perform all operations
- **DdsOperatorGroup** - Users that belong to this group can perform all operations except security setup
- **DdsMonitorGroup** - Users that belong to this group can only perform read only operations

The following table lists all the functions which can be performed by the roles and groups mentioned above.

**Table 4–1 Functions Performed by Roles and Groups**

| Role Name                   | Admin Role    | Operator Role    | Monitor Role    |
|-----------------------------|---------------|------------------|-----------------|
| Group Name                  | DdsAdminGroup | DdsOperatorGroup | DdsMonitorGroup |
| Create Access Level         | Yes           | No               | No              |
| Delete Access Level         | Yes           | No               | No              |
| Create Security Group       | Yes           | No               | No              |
| Delete Security Group       | Yes           | No               | No              |
| Create Table Level Security | Yes           | No               | No              |
| Delete Table Level Security | Yes           | No               | No              |

**Table 4-1 (Cont.) Functions Performed by Roles and Groups**

| <b>Role Name</b>       | <b>Admin Role</b>    | <b>Operator Role</b>    | <b>Monitor Role</b>    |
|------------------------|----------------------|-------------------------|------------------------|
| <b>Group Name</b>      | <b>DdsAdminGroup</b> | <b>DdsOperatorGroup</b> | <b>DdsMonitorGroup</b> |
| View Table Definition  | Yes                  | Yes                     | Yes                    |
| Create Queries         | Yes                  | Yes                     | Yes                    |
| Run Queries            | Yes                  | Yes                     | Yes                    |
| View Table Data        | Yes                  | Yes                     | Yes                    |
| Modify Table data      | Yes                  | Yes                     | No                     |
| Delete Table Data      | Yes                  | Yes                     | No                     |
| Bulk Update Table Data | Yes                  | Yes                     | No                     |

## Access Levels

Access Level defines security permissions.

Here are the permissions that can be associated with an access level.

- DataReadPermission
- DataCreatePermission
- DataUpdatePermission
- DataDeletePermission
- RowAccessPermission
- ColumnAccessPermission
- DataSecuritySetupPermission

The following access levels are created when default security is setup using the end point (/resources/admin/security/setup/{schemaName}).

- **DdsAdminAccessLevel** - Provides all permissions
- **DdsOperatorAccessLevel** - Provides all permissions except DataSecuritySetupPermission
- **DdsMonitorAccessLevel** - Provides DataReadPermission, RowAccessPermission and ColumnAccessPermission

## Table Level Security

Table level security defines who can access a table with set of allowed permissions. A table cannot be accessed if table level security is not setup.

Table level security is associated with the following information.

- Schema Name
- Table Name
- Security Group - Provides who can access the table
- Access Level - Provides permissions

Table level securities are set up for all tables when default security is setup using the end point. Every table is setup to be accessed by users in all security groups with default permissions.



Example

Table - DdsAdminGroup, DdsAdminAccessLevel

Table - DdsOperatorGroup, DdsOperatorAccessLevel

Table - DdsMonitorGroup, DdsMonitorAccessLevel

## Column Level Security

Column level security defines who can access a column as well as permission to access the column. By default, Dynamic Data Service allows access to a column if the user has access to the table.

Column level security is defined using the following information.

- Column Name
- Column Permission Type - Valid values are ALLOW, DONT\_ALLOW, MASK
- Table Level Security

## Row Level Security

Row level security defines who can access row(s) in a table. Row level security can be enabled if the table has user and group columns.

Row level security is defined using the following information.

- EnableRowAccess
- UserColumnName
- GroupColumnName
- TableLevelSecurity

## Security Setup

Follow the steps to setup security. Users will not be able access any data without the security setup. Only users that belong to the DdsAdminGroup can set up security.

1. Run the end point (/resources/admin/security/config) to create security configuration for the schema.

```
{
  "schemaName": "schema",
  "enableSecurity": "true",
  "tableLevelSecurityType": "WHITELIST",
  "columnLevelSecurityType": "BLACKLIST"
}
```

2. Run default security setup end point (/resources/admin/security/setup/{schemaName}) to set up default security for all tables.

If the default security setup is not sufficient, use Dynamic Data Service UI to setup the access levels, security groups, and table level securities.

## System Options

Dynamic Data Service finds data sources with the JNDI name that starts with "jdbc" by default. The following system option can be used to provide a different JNDI Name pattern.

`dataSourcePattern`

The system option can be setup using the end point (`/resources/system-setting/system-options`).

---

---

## RESTful Services

This chapter provides information about the RESTful Services used by DDS.

### Discover

This end point returns a list of Dynamic Data Service end points.

HTTP Operation: GET

Path: **/resources/discover**

### Get Data

This end point returns data from the provided schema and table(s). The query parameters "fromTables" allows single or multiple tables.

The query parameter "rowFilter" can be used to provide a join between tables.

Tables and columns need to be qualified if multiple tables are provided in "fromTables". The data is returned in JSON/XML format and one page of data is returned. The response contains a link to the next page if there are additional pages.

HTTP Operation: GET

Path: **/resources/dds/{schemaName}/data?fromTables=<tables>**

The following query parameters can be provided to filter the data.

- columnFilter - Columns to be included in the response
- rowFilter - Predicate can be provided in the rowFilter to filter the data
- sortBy - Valid values are ascending(asc) or descending(desc)
- Page - Page number
- pageLimit - Number of records to be included in the page. The default page size is 25
- format - Valid values are short or long. Short format provides data without column names. Long format provides data with column names.

### Get Data from a Table

This end point returns data from a table using a primary key.

HTTP Operation: GET

Path: **/resources/dds/{schemaName}/{table}?idFilter=<Primary Key>**

## Insert Data

This end point inserts data in a table. Input can be a single record or multiple records provided in JSON format. Users that belong to the admin or operator group can only perform this operation. Users also need to have DataCreatePermission to insert data.

HTTP Operation: PUT

Path: `/resources/dds/{schemaName}/data/{table}`

Sample input for inserting data in the BDI\_RECEIVER\_OPTIONS table

`/dds/bdi_rms_schema/data/BDI_RECEIVER_OPTIONS`

```
{
  "items":
  [
    {
      "ID":1,
      "BASE_FOLDER":"base",
      "FOLDER_TEMPLATE":"baseTemplate",
      "INTERFACE_MODULE":"interfaceModule",
      "INTERFACE_SHORT_NAME":"interfaceShortName",
      "MERGE_STRATEGY":"mergeStrategy"
    }
  ]
}
```

## Update Data

This end point updates data in a table using the rowFilter query parameter and input is provided in JSON format. Users that belong to the admin or operator group can only perform this operation. Users also need to have DataUpdatePermission to update data.

HTTP Operation: POST

Path: `/resources/dds/{schemaName}/data/{table}?rowFilter=<predicate>`

Sample input for updating data in the BDI\_SYSTEM\_OPTIONS table

`/dds/bdi_rms_schema/data/BDI_SYSTEM_OPTIONS?rowFilter=VARIABLE_NAME='LOADJOBDEF'`

```
{
  "VARIABLE_VALUE": "TRUE"
}
```

## Delete Data

This end point deletes data from a table based on the rowFilter query parameter. Users that belong to the admin or operator group can only perform this operation. Users also need to have DataDeletePermission to delete data.

HTTP Operation: DELETE

Path: `/resources/dds/{schemaName}/data/{table}?rowFilter=<predicate>`

Sample End Point

`/dds/bdi_rms_schema/data/BDI_SYSTEM_OPTIONS?rowFilter=VARIABLE NAME='LOADJOBDEF'`

## Get Schemas

This end point returns all configured database schemas.

HTTP Operation: GET

Path: `/resources/dds/schemas`

## Get Tables

This end point returns all tables from a database schema.

HTTP Operation: GET

Path: `/resources/dds/{schemaName}/tables`

## Get Columns for a Table

This end point returns columns for a table.

HTTP Operation: GET

Path: `/resources/dds/{schemaName}/{table}`

## Get System Options

This end point returns all system options from the DDS\_SYSTEM\_OPTIONS table.

HTTP Operation: GET

Path: `/resources/system-setting/system-options`

## Create System Options

This end point creates a system option in the DDS\_SYSTEM\_OPTIONS table.

HTTP Operation: PUT

Path: `/resources/system-setting/system-options`

Sample Input

```
{
  "key": "testKey",
  "value": "testValue"
}
```

## Update System Options

This end point updates a system option in the DDS\_SYSTEM\_OPTIONS table.

HTTP Operation: POST

Path: `/resources/system-setting/system-options`

Sample Input

```
{
  "key": "testKey",
  "value": "testValue1"
}
```

## Delete System Options

This end point deletes a system option in the DDS\_SYSTEM\_OPTIONS table.

HTTP Operation: DELETE

Path: **`/resources/system-setting/system-options/{key}`**

## Reset Cache

This end point resets the system option cache.

HTTP Operation: POST

Path: **`/resources/system-setting/reset-cache`**

---

---

## Security RESTful Services

The security end points are only accessible by users that belong to the admin group.

### Get Security Groups

This end point returns configured security groups.

HTTP Operation: GET

Path: `/resources/admin/security/groups`

### Get Security Group Permissions

This end point returns all security permissions.

HTTP Operation: GET

Path: `/resources/admin/security/permissions`

### Get Access Levels

This end point returns access levels.

HTTP Operation: GET

Path: `/resources/admin/security/accessLevels`

### Get Security Config for a Schema

This end point returns the security configuration for a schema.

HTTP Operation: GET

Path: `/resources/admin/security/config/{schemaName}`

### Get All Security Config

This end point returns all security configurations.

HTTP Operation: GET

Path: `/resources/admin/security/config`

### Get All Table Level Securities

This end point returns all table level securities for a schema.

HTTP Operation: GET

Path: `/resources/admin/security/table/{schemaName}`

## Get Table Level Security for Table

This end point returns table level security for a table in a schema.

HTTP Operation: GET

Path: `/resources/admin/security/table/{schemaName}/{tableName}`

## Get Table Level Security for Table and Group

This end point returns table level security for a table and group.

HTTP Operation: GET

Path: `/resources/admin/security/table/{schemaName}/{tableName}/{groupName}`

## Get Column Level Security for Table and Group

This end point returns column level security for a table and group.

HTTP Operation: GET

Path: `/resources/admin/security/column/{schemaName}/{tableName}/{groupName}`

## Get Row Level Security for Table and Group

This end point returns row level security for a table, group, and access level.

HTTP Operation: GET

Path:

`/resources/admin/security/row/{schemaName}/{tableName}/{groupName}/{accessLevel}`

## Create Access Level

This end point creates an access level.

HTTP Operation: PUT

Path: `/resources/admin/security/accessLevel`

Sample Input

```
{
  "accessLevelName": "testAccessLevel",
  "accessLevelPermissionVoList": [
    "securityPermission": "DataCreatePermission"
  ]
}
```

## Update Access Level

This end point updates an access level.

HTTP Operation: POST



Path: `/resources/admin/security/accessLevel`

Sample Input

```
{
  "accessLevelName": "testAccessLevel",
  "accessLevelPermissionVoList": [
    "securityPermission": "DataCreatePermission"
    "securityPermission": "DataDeletePermission"
  ]
}
```

## Delete Access Level

This end point deletes an access level.

HTTP Operation: DELETE

Path: `/resources/admin/security/accessLevel/{accessLevelName}`

## Create Table Level Security

This end point creates table level security.

HTTP Operation: PUT

Path: `/resources/admin/security/table`

Sample Input

```
{
  "schemaName": "testSchema",
  "tableName": "TEST",
  "securityGroup": "DDSMonitorGroup",
  "accessLevelVo": {
    "accessLevelName": "testAccessLevel"
  }
}
```

## Create Table Level Securities

This end point creates table level securities.

HTTP Operation: PUT

Path: `/resources/admin/security/tables`

Sample Input

```
{
  "tableLevelSecurityVoList": [
    {
      "schemaName": "testSchema",
      "tableName": "TEST",
      "securityGroup": "DDSMonitorGroup",
      "accessLevelVo": {
        "accessLevelName": "testAccessLevel"
      }
    }
  ]
}
```

## Update Table Level Security

This end point updates table level security.

HTTP Operation: POST

Path: `/resources/admin/security/table`

Sample Input

```
{
  "schemaName": "testSchema",
  "tableName": "TEST",
  "securityGroup": "DDSMonitorGroup",
  "accessLevelVo": {
    "accessLevelName": "monitorAccessLevel"
  }
}
```

## Update Table Level Securities

This end point updates table level securities.

HTTP Operation: POST

Path: `/resources/admin/security/tables`

Sample Input

```
{
  "tableLevelSecurityVoList": [
    {
      "schemaName": "testSchema",
      "tableName": "TEST",
      "securityGroup": "DDSMonitorGroup",
      "accessLevelVo": {
        "accessLevelName": "monitorAccessLevel"
      }
    }
  ]
}
```

## Delete Table Level Security

This end point deletes table level security.

HTTP Operation: DELETE

Path: `/resources/admin/security/table`

Sample Input

```
{
  "schemaName": "testSchema",
  "tableName": "TEST",
  "securityGroup": "DDSMonitorGroup",
  "accessLevelVo": {
    "accessLevelName": "testAccessLevel"
  }
}
```

## Delete Table Level Security by Id

This end point deletes table level security by id.

HTTP Operation: DELETE

Path: `/resources/admin/security/table/{id}`

## Delete Table Level Securities by Type

This end point deletes table level securities.

HTTP Operation: DELETE

Path:

`/resources/admin/security/table/bulk/{schemaName}?keys=<keys>&keyType=<key Type>`

Valid KeyType values: table, securityGroup, accessLevel

If keyType is table, then a list of comma separated table names needs to be provided in the keys query parameter.

If keyType is securityGroup, then a list of comma separated security groups needs to be provided in the keys query parameter.

If keyType is accessLevel, then a list of comma separated access levels need to be provided in the keys query parameter.

## Create Column Level Security

This end point creates column level security.

HTTP Operation: PUT

Path: `/resources/admin/security/column`

Valid Column Permission Type values: ALLOW, DONT\_ALLOW, MASK

Sample Input

```
{
  "columnName": "testColumn",
  "columnPermissionType": "ALLOW",
  "securityGroup": "DDSEOperatorGroup",
  "tableLevelSecurityVo": {
    "schemaName": "testSchema",
    "tableName": "TEST",
    "securityGroup": "DDSEOperatorGroup",
    "accessLevelVo": {
      "accessLevelName": "testAccessLevel"
    }
  }
}
```

## Update Column Level Security

This end point updates column level security.

HTTP Operation: POST

Path: `/resources/admin/security/column`

Valid Column Permission Type values: ALLOW, DONT\_ALLOW, MASK

Sample Input

```
{
  "columnName": "testColumn",
  "columnPermissionType": "MASK",
  "securityGroup": "DDSOperatorGroup",
  "tableLevelSecurityVo": {
    "schemaName": "testSchema",
    "tableName": "TEST",
    "securityGroup": "DDSOperatorGroup",
    "accessLevelVo": {
      "accessLevelName": "testAccessLevel"
    }
  }
}
```

## Delete Column Level Security

This end point deletes column level security.

HTTP Operation: DELETE

Path: **/resources/admin/security/column**

Sample Input

```
{
  "columnName": "testColumn",
  "tableLevelSecurityVo": {
    "schemaName": "testSchema",
    "tableName": "TEST",
    "securityGroup": "DDSOperatorGroup",
    "accessLevelVo": {
      "accessLevelName": "testAccessLevel"
    }
  }
}
```

## Create Data Security Configuration

This end point creates data security configuration.

HTTP Operation: PUT

Path: **/resources/admin/security/config**

Valid values for security type: **WHITELIST, BLACKLIST**

**WHITELIST** allows access only if security is setup (Table level security)

**BLACKLIST** denies access to entities (such as column level security) if it is set up and allows others.

Sample Input

```
{
  "schemaName": "testSchema",
  "enableSecurity": "true",
  "tableLevelSecurityType": "WHITELIST",
  "columnLevelSecurityType": "BLACKLIST"
}
```

## Update Data Security Configuration

This end point updates the data security configuration.

HTTP Operation: POST

Path: `/resources/admin/security/config`

Valid values for security type: **WHITELIST**, **BLACKLIST**

**WHITELIST** allows access only if security is setup (such as Table level security)

**BLACKLIST** denies access to entities (such as column level security) if it is set up and allows others.

Sample Input

```
{
  "schemaName": "testSchema",
  "enableSecurity": "false",
  "tableLevelSecurityType": "WHITELIST",
  "columnLevelSecurityType": "BLACKLIST"
}
```

## Delete Data Security Configuration

This end point deletes the data security configuration for a schema.

HTTP Operation: DELETE

Path: `/resources/admin/security/config/{schemaName}`

## Create Security Group

This end point creates a security group.

HTTP Operation: PUT

Path: `/resources/admin/security/groups`

Sample Input

```
{
  "groupName": "DDSAdminGroup",
}
```

## Delete Security Group

This end point deletes a security group.

HTTP Operation: DELETE

Path: `/resources/admin/security/groups/{groupName}`

## Create Row Level Security

This end point creates row level security.

Row level security can be enforced if there are user and group columns in the table.

HTTP Operation: PUT

Path: `/resources/admin/security/row`

Sample Input

```
{
  "enableRowAccess": "true",
  "userColumnName": "USER",
  "groupColumnName": "GROUP",
  "tableLevelSecurityVo": {
    "schemaName": "testSchema",
    "tableName": "TEST",
    "securityGroup": "DDSOperatorGroup",
    "accessLevelVo": {
      "accessLevelName": "testAccessLevel"
    }
  }
}
```

## Update Row Level Security

This end point updates row level security for a table.

HTTP Operation: POST

Path: `/resources/admin/security/row`

Sample Input

```
{
  "enableRowAccess": "false",
  "tableLevelSecurityVo": {
    "schemaName": "testSchema",
    "tableName": "TEST",
    "securityGroup": "DDSOperatorGroup",
    "accessLevelVo": {
      "accessLevelName": "testAccessLevel"
    }
  }
}
```

## Delete Row Level Security

This end point deletes row level security.

HTTP Operation: DELETE

Path:

`/resources/admin/security/row/{schemaName}/{tableName}/{securityGroup}/{accessLevelName}`

## Create Default Security Setup

This end point creates default security setup for all tables in a schema. It creates the following.

**Security Groups** - DdsAdminGroup, DdsOperatorGroup, DdsMonitorGroup

**Access Levels** - DdsAdminAccessLevel, DdsOperatorAccessLevel, DdsMonitorAccessLevel

**DdsAdminAccessLevel** has permissions to all operations.

**DDSOperatorAccessLevel** has permissions to all operations except DataSecuritySetupPermission

**DDSMonitorAccessLevel** has permissions to read only operations

HTTP Operation: PUT

Path: **`/resources/admin/security/setup/{schemaName}`**





---

---

## User Interface

The DDS UI is divided into two sections; Admin tab, and Designer tab. The Admin tab lets admin users control access and perform security operations. The Designer tab lets admin, operators and monitor users perform authorized functions.

This chapter explains the main UI functionality available to the user.

### Admin Tab Guide

This tab has all the functions for configuring security related permissions and user access. Only the admin user has access to this tab. The admin can enable read, write and modify access permissions and also limit functionality of a user.

The Admin Tab has three main sections:

- Manage Access Levels
- Manage Security Groups
- Manage Security permissions

### Initial Permissions Setup

In order to access the data on tables being viewed in the Designer Tab, first the Table level securities have to be setup in the Admin tab of the DDS Application. The Admin can select the desired database schema from the associated database schema dropdown.

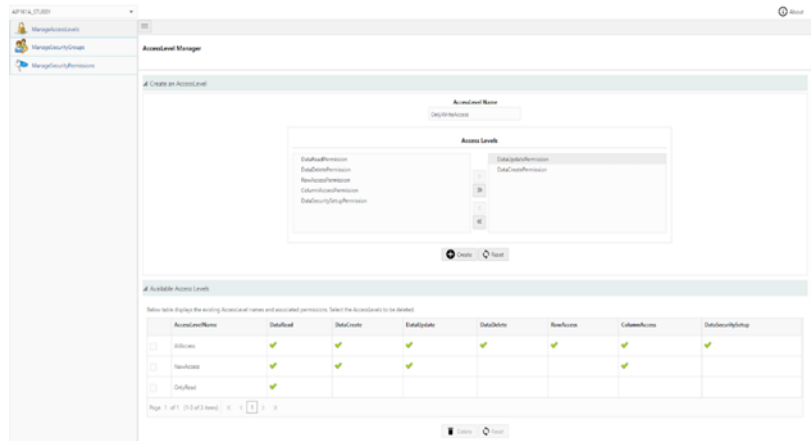
Access to this tab is only allowed to the Admin User.

### Manage Access Levels

This tab allows the creation and deletion of Access Levels for data and functionality. These can be assigned to groups for particular tables in a schema.

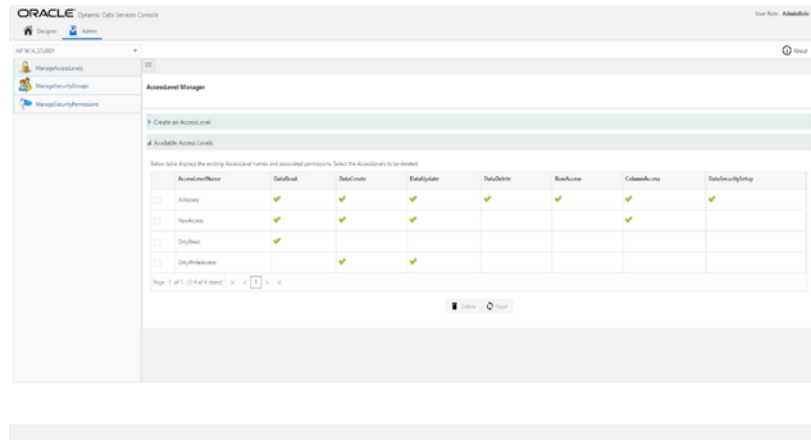
#### **To create an access level**

1. Input a name in the Name text box.
2. Select the necessary access permissions.
3. Click **Create**.



### To delete an access level

1. Select the Access Level to be deleted from the table which displays the available access levels.
2. Click **Delete**.

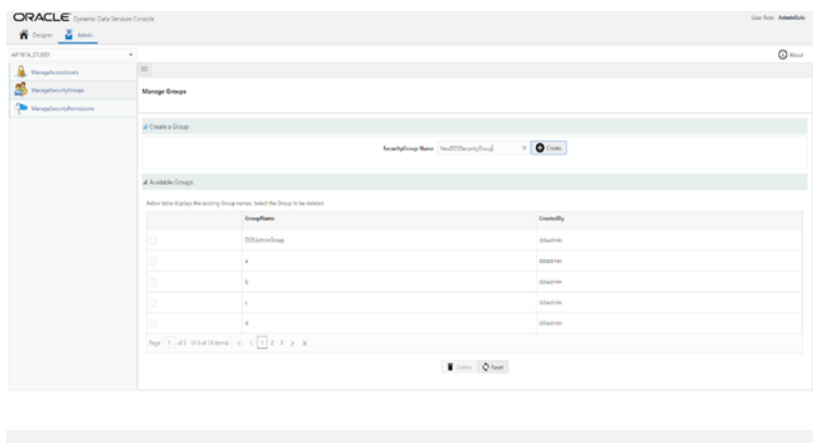


## Manage Security Groups

This tab includes functionality for the creation and deletion of Security Groups. Security Groups are groups of users that have a certain level of data access. These, along with one or more access levels, can be assigned to a table to limit access to data and functionality in the Designer tab.

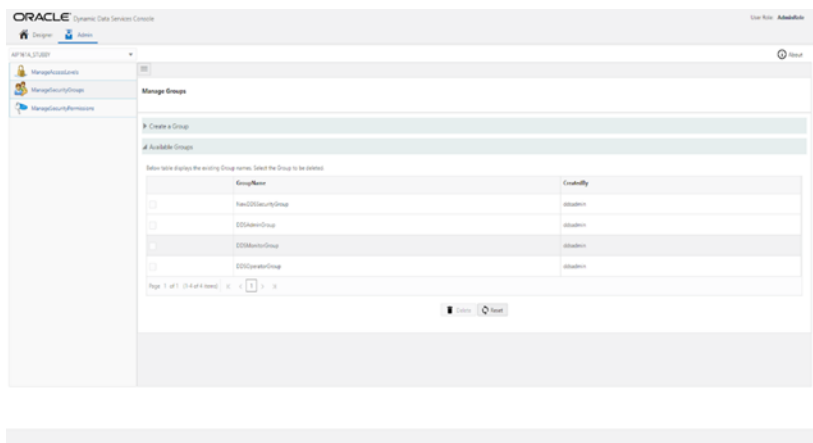
### To create a Security Group

1. Input a name in the Name text box.
2. Click **Create**.



### To delete a Security Group

1. Select the Security Group to be deleted from the table that displays the available Security Groups.
2. Click **Delete**.

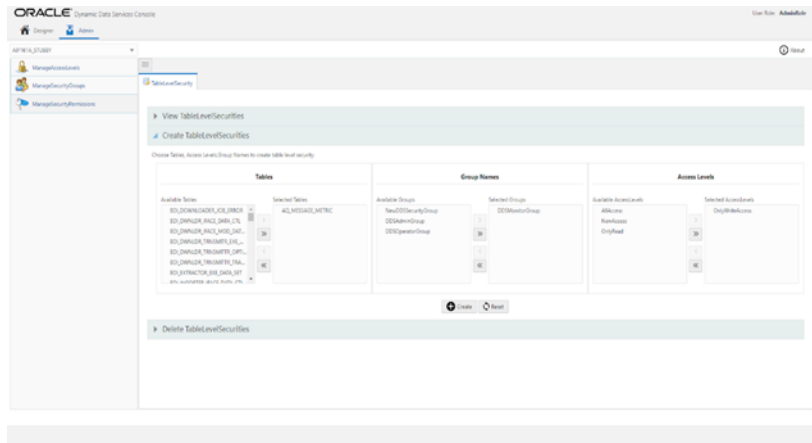


## Manage Security Permissions

This tab allows the user to create and delete Security Permissions. Security Permissions are a combination of one or more Access levels in a Security Group mapped to a table to restrict data access and also limit functionality, like removing the ability to create, modify or delete data.

### To create a Security Permission for a table

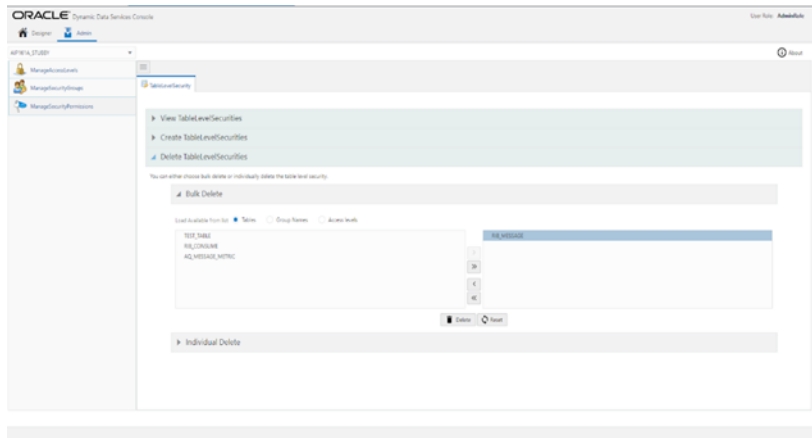
1. In the Create TableLevelSecurities collapsible pane, select the desired table.
2. Then select the required security group.
3. Finally select the required access level(s).
4. Click **Create**.



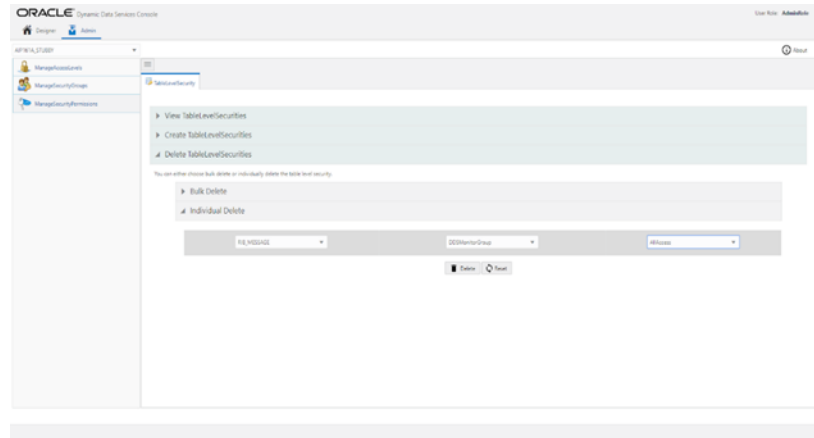
### To delete Security Permission(s)

Functionality is provided to either perform bulk delete or individual deletion of security.

1. To bulk delete security permissions:
  - a. Select the required table name(s) or security group(s) or access level(s).
  - b. Click **Delete**.



2. To individually delete security permissions:
  - a. Select the required table name.
  - b. Select the particular security group associated.
  - c. Select the associated access level.
  - d. Click **Delete**.



## Designer Tab Guide

This tab contains all the CRUD operations to be performed by user on tables in the database schema. The user can select the desired database schema available in the database schema dropdown. It has two sections:

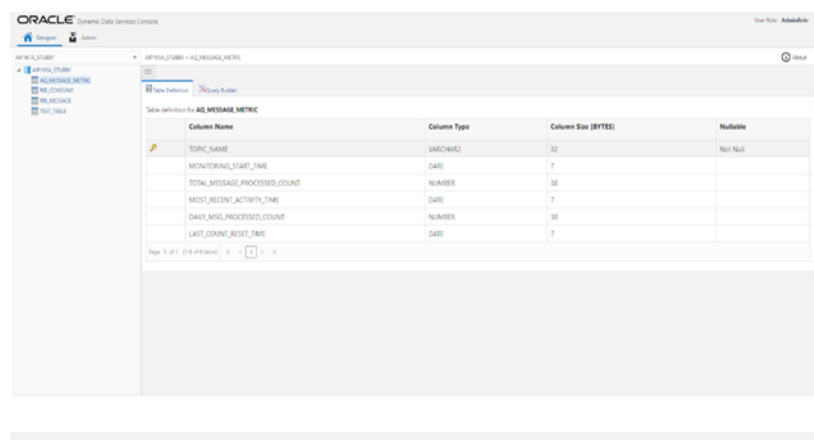
- Table Definition
- Query Builder

Make sure the user has sufficient permissions before performing any action in the designer tab.

### Table Definition

This tab allows the user to see the basic definition of the Database table in a simplified format presented in a tabular form.

To view a table's definition, select the required table from the list of tables from the left pane of the screen. Clicking on the table name will load the definition on the display screen under the Table Definition tab.



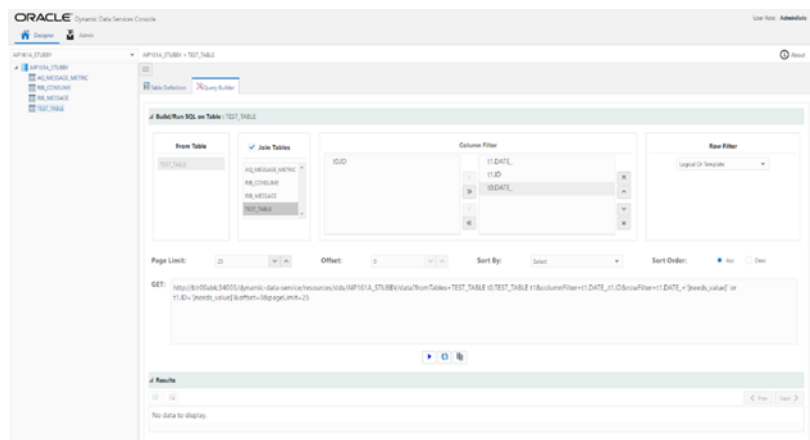
### Query Builder

This tab holds the functionality for data manipulation. Functions include Querying records to creation, modification and deletion of records.

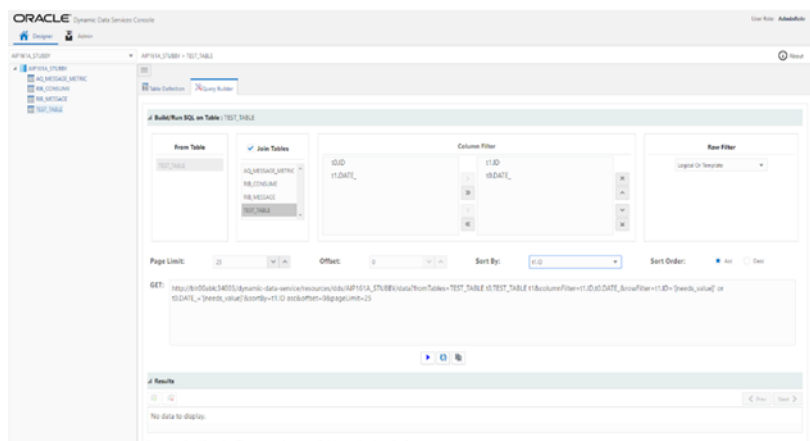





5. Select the row filter.

**Note:** The row filter is applied only to the first two columns in the table, so reorder the columns such that the ones that have to be used as row filter are in the two positions.



6. Set the page limit, this is the number of records that are to be displayed on one page, the remaining records will be paginated and displayed upon request.
7. The Sort by drop down list works if there are columns selected for a column filter. Select the desired column, and then select the sort order, ascending or descending.




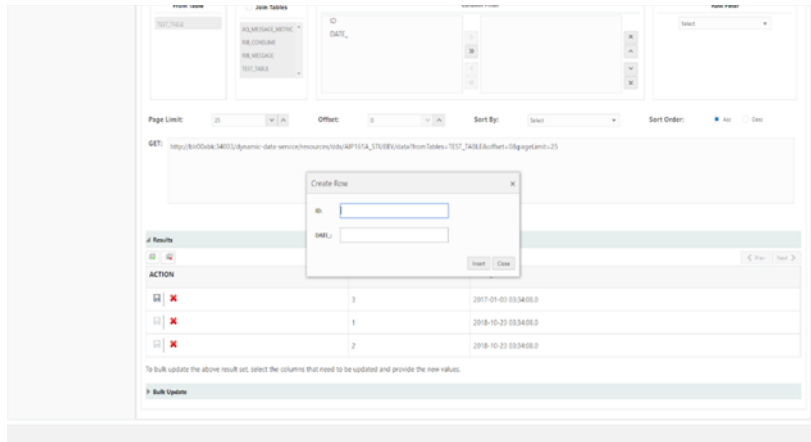
8. Click on the run button  to get the Query results using the settings configured earlier.
9. The reset button  sets the settings back to default.
10. The copy query button  is used to copy the GET query generated in the "GET" box. Click on the button and the link is copied. Paste the link in a browser and press enter/return to execute the query.

### Data Insertion

To insert the data in the table:


1. The query results from the GET call are displayed here in an editable tabulated format.

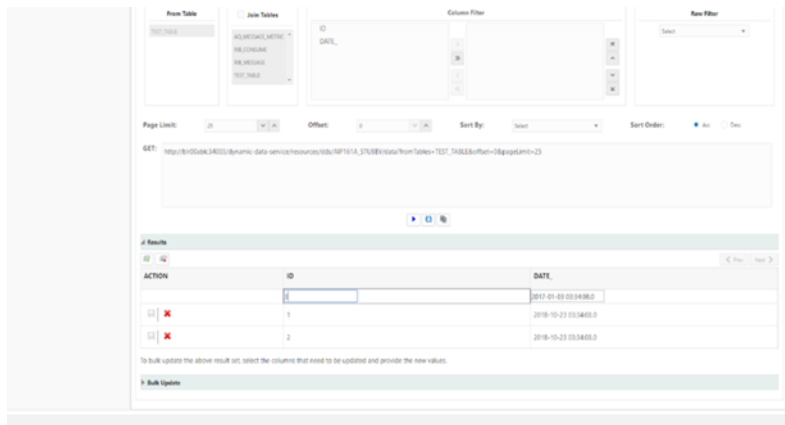
2. To insert a record click on the insert record button .
3. In the pop-up, enter the data. Make sure to enter all values for not null columns.
4. Click **Insert**.
5. The record will be successfully inserted after validation.



### Data Modification

To modify existing data, select the data from desired table.

1. Double click on the record to be modified
2. Edit the contents
3. Press escape on the keyboard to exit edit mode and click on the save button  to save the changes.
4. Message - Record updated successfully will be displayed after validation

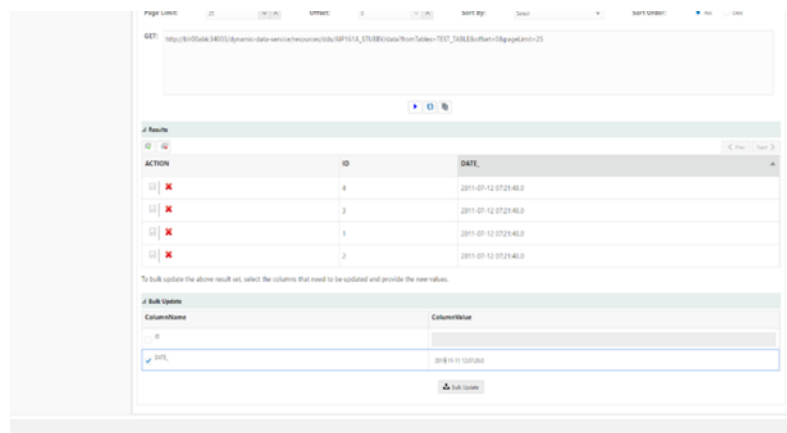



### Bulk Updates of Data

The option is available to update bulk records on selected data. To bulk update column(s):

1. In the bulk update collapsible pane one or more columns for all records in the queried table can be updated with a single value, if the column is not unique.
2. Select the checkbox of the required column to enable edit mode.
3. Enter the data as necessary.






4. Click the Bulk Update option  to update the selected columns for all the records in the table.
5. Records will be updated after successful validation of data.

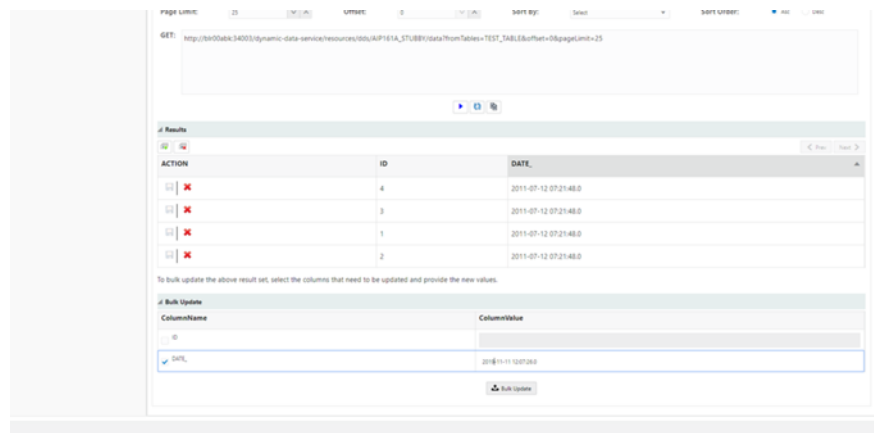
### Data Deletion

This option is available to perform individual and bulk deletions of records.

Individual Delete - To delete a single record, click the delete button  on the corresponding record.

Bulk Delete -To delete all records, click the delete all button  and confirm once the confirmation dialog pops up.

The number of records deleted message will be displayed upon successful deletion of records.





## Advanced Backend Features

The DDS backend service supports many SQL query commands. Some of these commands can be added to a query using the components available in the DDS UI, while others must be added by manually typing them into the query itself. Once that is done, the query can then be run through the DDS UI, or a browser, or a cURL commands to get the results.

Follow the instructions in the "Query Builder" section within the "Designer Tab Guide" section of the "User Interface" chapter for instructions on setting up a basic query in the DDS UI. Once those steps have been followed, a query is generated in the GET box. The query can be edited to add the following statements individually or in a combination as required.

---

**Note:** Please maintain the formatting of the GET request, making sure all the special characters (like the separating ampersand &) are in place, a lack of which may cause unexpected errors.

---

### LIKE Statement

The LIKE statement can be added separately to the query request to view only the results that are similar to the LIKE value. A simple GET request and result with LIKE statement can be like the following:

```
http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_schema>/data?fromTables=<selected_table>&rowFilter=<COULMN_NAME> LIKE '<data_to_search>'&offset=0&pageLimit=25
```

A curl equivalent of the above example:

```
curl "http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_schema>/data?fromTables=<selected_table>&rowFilter=<COULMN_NAME>%20LIKE%20'<data_to_search>'&offset=0&pageLimit=25" -u <dds_user>:<dds_pass> -o filename.txt
```

```
GET: http://rgbu-mum-33.anbomprshared1.gbucdsint02bom.oraclevcn.com:7005/dynamic-data-service-web/resources/dds/RFI_SHWE_19/data?fromTables=EXT_FIN_FILE_CONTENT_DETAIL&rowFilter=RFI_SEQ_NO LIKE '93' and FILE_ID BETWEEN '2504' AND '2505'&offset=0&pageLimit=25
```

| Results |            |         |                       |                       |
|---------|------------|---------|-----------------------|-----------------------|
| ACTION  | RFI_SEQ_NO | FILE_ID | CREATION_DATE         | LAST_UPDATE_DATE      |
|         | 93         | 2504    | 2021-05-27 18:23:02.0 | 2021-05-27 18:23:02.0 |
|         | 93         | 2505    | 2021-05-27 18:23:02.0 | 2021-05-27 18:23:02.0 |

## IN Statement

The IN statement can be added separately to the query request to only view results containing one of a set of possible values. A simple GET request and result with IN statement can be like the following:

```
http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_
schema>/data?fromTables=<selected_table>&rowFilter=COULMN_NAME LIKE '<data_to_
search>'&offset=0&pageLimit=25
```

A curl equivalent of the above example:

```
curl "http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_
schema>/data?fromTables=<selected_table>&rowFilter=COULMN_
NAME%20IN%20('<value1>', '<value2>', '<value3>', ...)&offset=0&pageLimit=25" -u <dds_
user>:<dds_pass> -o filename.txt
```

---

**Note:** The IN Statement supports searching using comma-separated values in parentheses. Sub-queries are **NOT** supported by the DDS design.

---

GET: `http://rgbu-mum-33.snbomprshared1.gbucdsint02bom.oraclevcn.com:7005/dynamic-data-service-web/resources/dds/DOS_RIBEH/data?fromTables=TEST_TABLE_2&rowFilter=COLUMN2 IN ('Iden2','Iden3','Iden4')&offset=0&pageLimit=25`

| ACTION | COLUMN1 | COLUMN2 | COLUMN3 | COLUMN4         |
|--------|---------|---------|---------|-----------------|
| ✘      | Data2   | Iden4   | Val2    | NO MESSAGE DATA |
| ✘      | Data3   | Iden3   | "Val3"  | Invalid value   |
| ✘      | Data4   | Iden4   | Val4    | Not Null        |

## Comparative Operators

Comparative operators (like < and >) can be used to filter data for a particular range. A GET request and result with these operators can be like the following:

```
http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_
schema>/data?fromTables=<selected_table>&rowFilter=COULMN_NAME<'data_to_
search'&offset=0&pageLimit=25
```

A curl equivalent of the above example:

```
curl "http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_
schema>/data?fromTables=<selected_table>&rowFilter=COULMN_NAME<'data_to_
search'&offset=0&pageLimit=25" -u <dds_user>:<dds_pass> -o filename.txt
```

GET: `http://rgbj-mum-33.snbomprshared1.gbucdsint02bom.oraclevcn.com:7005/dynamic-data-service-web/resources/dds/RFI_SHWE_19/data?fromTables=EXT_FIN_FILE_CONTENT_DETAIL&rowFilter=FILE_ID=4788 and RFI_SEQ_NO=2&offset=0&pageLimit=25`

| ACTION | RFI_SEQ_NO | FILE_ID | CREATION_DATE         | LAST_UPDATE_DATE      |
|--------|------------|---------|-----------------------|-----------------------|
|        | 1          | 4789    | 2021-07-14 10:20:29.0 | 2021-07-14 10:20:29.0 |
|        | 1          | 4791    | 2021-07-14 10:23:57.0 | 2021-07-14 10:23:57.0 |
|        | 1          | 4792    | 2021-07-14 10:23:57.0 | 2021-07-14 10:23:57.0 |

## Filtering DATE Column

Date columns can be filtered using the 'DD-MMM-YYYY' date format. Additionally, `SYSDATE±N` can also be used to query date columns.

GET: `http://rgbj-mum-33.snbomprshared1.gbucdsint02bom.oraclevcn.com:7005/dynamic-data-service-web/resources/dds/RFI_SHWE_19/data?fromTables=EXT_FIN_FILE_CONTENT_DETAIL&rowFilter=CREATION_DATE>'31-MAY-2021' and LAST_UPDATE_DATE<SYSDATE&offset=0&pageLimit=25`

| ACTION | RFI_SEQ_NO | FILE_ID | CREATION_DATE         | LAST_UPDATE_DATE      |
|--------|------------|---------|-----------------------|-----------------------|
|        | 93         | 2777    | 2021-05-31 05:52:50.0 | 2021-05-31 05:52:50.0 |
|        | 93         | 2778    | 2021-05-31 05:52:50.0 | 2021-05-31 05:52:50.0 |
|        | 1          | 4789    | 2021-07-14 10:20:29.0 | 2021-07-14 10:20:29.0 |

Example with the `SYSDATE` keyword:

```
http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_schema>/data?fromTables=<selected_table>&rowFilter=CREATION_DATE<SYSDATE-1&offset=0&pageLimit=25
```

A curl equivalent of the above example:

```
curl "http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_schema>/data?fromTables=<selected_table>&rowFilter=CREATION_DATE<SYSDATE-1&offset=0&pageLimit=25" -u <dds_user>:<dds_pass> -o file-name.txt
```

Example with 'DD-MMM-YYYY' date format:

```
http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_schema>/data?fromTables=<selected_table>&rowFilter=CREATION_DATE>'01-JAN-1976'&offset=0&pageLimit=25
```

A curl equivalent of the above example:

```
curl "http://<dds_host>:<port>/dynamic-data-service-web/resources/dds/<selected_schema>/data?fromTables=<selected_table>&rowFilter=CREATION_DATE>'01-JAN-1976'&offset=0&pageLimit=25" -u <dds_user>:<dds_pass> -o file-name.txt
```

## Combinations of OR, AND & BETWEEN Statements

The conditional statements OR, AND & BETWEEN can be used in combination with each other using parentheses to further filter results based on required conditions. The Filter DDS row in the UI can be used to add a single conditional statement, upon



## Viewing Query Response as JSON Data

The DDS backend supports sending data as either JSON or XML.

To view the data as XML, run the query in a curl command by adding the -H switch set like the following:

```
curl "http://<dds_host>:<dds_
port>/dynamic-data-service-web/resources/dds/<selected_
schema>/data?fromTables=<selected_table>&offset=0&pageLimit=25" -H
"Accept:application/xml" -u <dds_user>:<dds_pass> -o file_name.txt
```

JSON is the default response format. To return a JSON object, do not include the -H switch in the cURL command.

Or, you can set the cURL command with the -H switch setting Accept to application/json in the header. The cURL with output is as follows:

```
curl "http://<dds_host>:<dds_
port>/dynamic-data-service-web/resources/dds/<selected_
schema>/data?fromTables=<selected_table>&offset=0&pageLimit=25" -H
"Accept:application/json" -u <dds_user>:<dds_pass> -o file_name.txt
```

---

**Note:** Ensure that the required DDS application username/password is supplied.

---

## JOINing Two or More Tables

DDS has the ability to join two or more tables. This is done using the JOIN element in the Query Builder Tab under the Designer section.

Use the check box to enable table joins. Once enabled, click inside the visible text field to view a list of tables. Select one or more tables as required.

---

**Note:** The row filter must be added to this, which can either be done by using the Row Filter in the UI or adding your own rowFilter by typing the required conditions in the GET query box. You must also ensure that all required columns are selected in the column filter, otherwise an error will display when you try to run the query.

---

GET: `http://rgbu-mum-33.snbomprshred1.gbucdsint02bom.oraclevcn.com:7005/dynamic-data-service-web/resources/dds/DDS_RIBEH/data?fromTables=RIB_MESSAGE,t0,RIB_MESSAGE_FAILURE,t1,RIB_MESSAGE_HOSPITAL_REF,t2&columnFilter=t0.MESSAGE_NUM,t0.FAMILY,t2.MESSAGE_FAMILY,t1.SEQ_NUMBER,t1.DESCRPTION&rowFilter=t0.MESSAGE_NUM=t1.MESSAGE_NUM and t1.MESSAGE_NUM=t2.MESSAGE_NUM and t0.FAMILY=t2.MESSAGE_FAMILY and t1.SEQ_NUMBER LIKE '1'&offset=0&pageLimit=25`

| ACTION | MESSAGE_NUM | FAMILY    | MESSAGE_FAMILY | SEQ_NUMBER | DESCRIPTION   |
|--------|-------------|-----------|----------------|------------|---|
|        | 1           | Receiving | Receiving      | 1          | javax.ejb.EJBException: Error while calling Injector Service.: Exception calling psql inject, package owner is null or empty string f |

A sample GET query with result is as follows:

```
http://<dds_host>:<dds_port>/dynamic-data-service-web/resources/dds/<SELECTED_
TABLE>/data?fromTables=TABLE1 t0, TABLE2 t1, TABLE3
t2&columnFilter=t0.COLUMN1,t1.COLUMN1,t2.COLUMN1,t0.COLUMN2,t1.COLUMN2,t2.COLUMN2,
t0.COLUMN3,t1.COLUMN3,t2.COLUMN3&rowFilter=t0.COLUMN1=t1.COLUMN1 and
t1.COLUMN1=t2.COLUMN1 and t1.COLUMN2=t2.COLUMN3 and t1.COLUMN3 LIKE
'1'&offset=0&pageLimit=25
```

A curl equivalent of the above example:

```
curl "http://<dds_host>:<dds_
port>/dynamic-data-service-web/resources/dds/<SELECTED_
TABLE>/data?fromTables=TABLE1%20t0, TABLE2%20t1, TABLE3%20t2&columnFilter=t0.COLUMN1
,t1.COLUMN1,t2.COLUMN1,t0.COLUMN2,t1.COLUMN2,t2.COLUMN2,t0.COLUMN3,t1.COLUMN3,t2.C
OLUMN3&rowFilter=t0.COLUMN1=t1.COLUMN1%20and%20t1.COLUMN1=t2.COLUMN1%20and%20t1.CO
LUMN2=t2.COLUMN3%20and%20t1.COLUMN3%20LIKE%20'1'&offset=0&pageLimit=25" -u <dds_
user>:<dds_pass> -o filename.txt
```