Using RDF Knowledge Graphs in the Oracle Public Cloud

1 Overview

This document provides detailed steps to setup and use RDF Semantic Graph as part of an Oracle Database Cloud Service. The steps are based on Oracle Database version 18.0.0.0 and may vary if a different database version is used. In addition, these steps assume a Windows 10 desktop as the client computer.

This how-to covers the following tasks:

- Provisioning a Database Cloud Service (DBCS) instance
- Using Putty to login to the newly created DBCS instance
- Connecting SQL Developer to the DBCS instance
- Using the RDF Semantic Graph plugin for SQL Developer to setup and configure RDF Semantic Graph
- Using the RDF Semantic Graph plugin to bulk load and query a publicly-available RDF dataset
- Setting up a Fuseki SPARQL endpoint on the DBCS instance to provide a W3C-standard REST interface

2 Before You Get Started

The following tasks should be completed before starting this how-to:

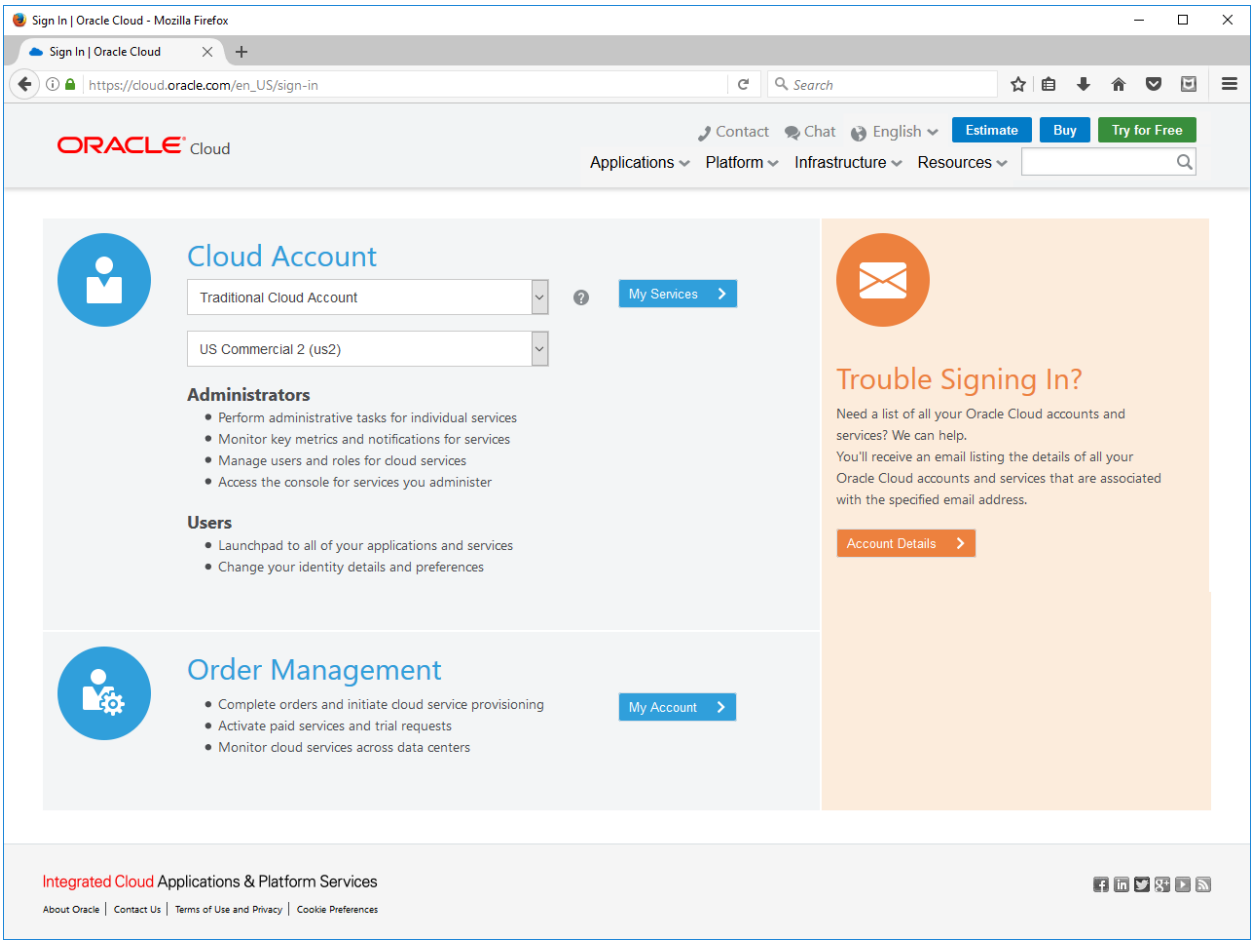
- Become familiar with DBCS, Oracle Spatial and Graph, and RDF Semantic Graph
    - DBCS Help Center: https://docs.oracle.com/en/cloud/paas/database-dbaas-cloud/index.html
    - Oracle Spatial and Graph: http://www.oracle.com/technetwork/database/options/spatialandgraph/overview/index.html
    - RDF Semantic Graph: https://docs.oracle.com/en/database/oracle/oracle-database/18/rdfrm/index.html
- Obtain an Oracle Cloud subscription or trial account including DBCS and Java Cloud Service (JCS)
    - Available at: http://cloud.oracle.com
- Download and install Puttygen and Putty onto your client computer
    - Available at: http://www.putty.org
- Download and install Cygwin onto your client computer
    - Available at: https://www.cygwin.com/
- Download and install WinSCP onto your client computer
    - Available at: https://winscp.net/eng/index.php

- Download and install Oracle SQL Developer onto your client computer
    - Available at: http://www.oracle.com/technetwork/developer-tools/sql-developer/overview/index.html

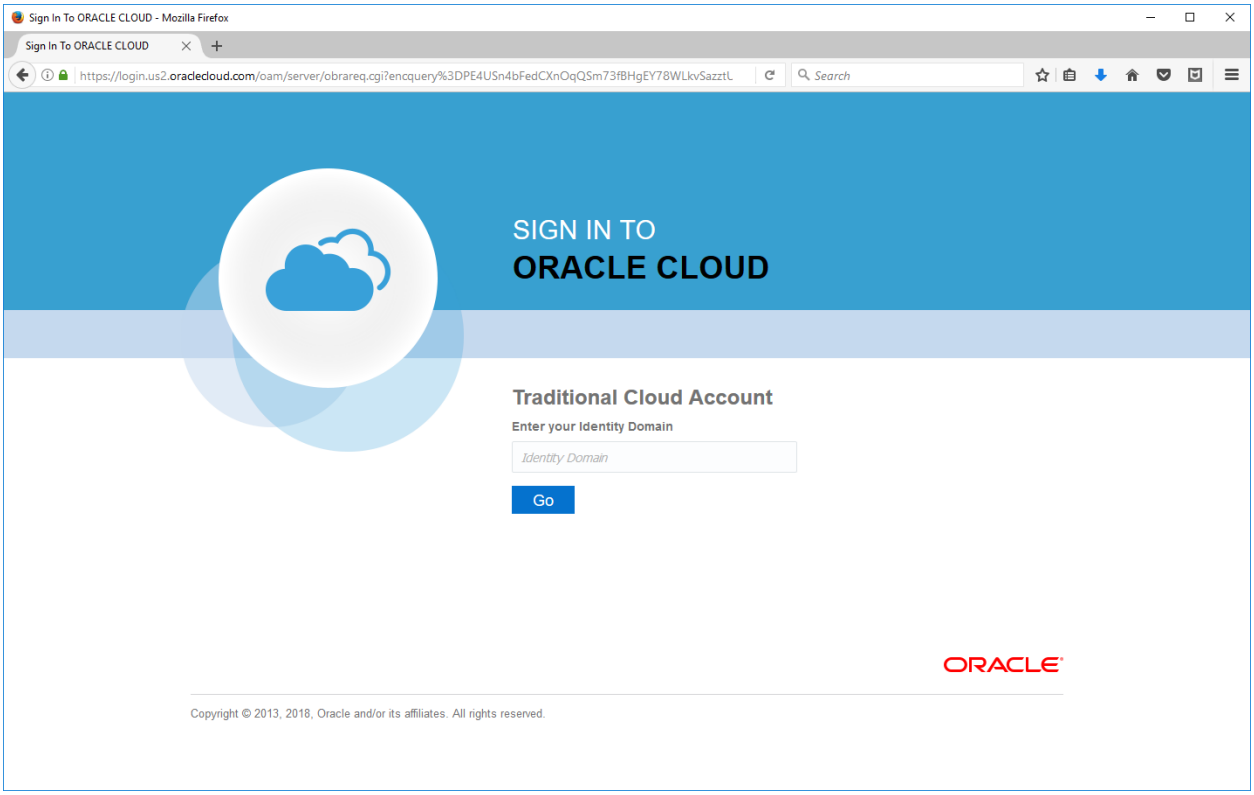
3 PROVISIONING AN ORACLE CLOUD DATABASE SERVICE (DBCS) INSTANCE

The steps below illustrate how to provision an Oracle DBCS instance.

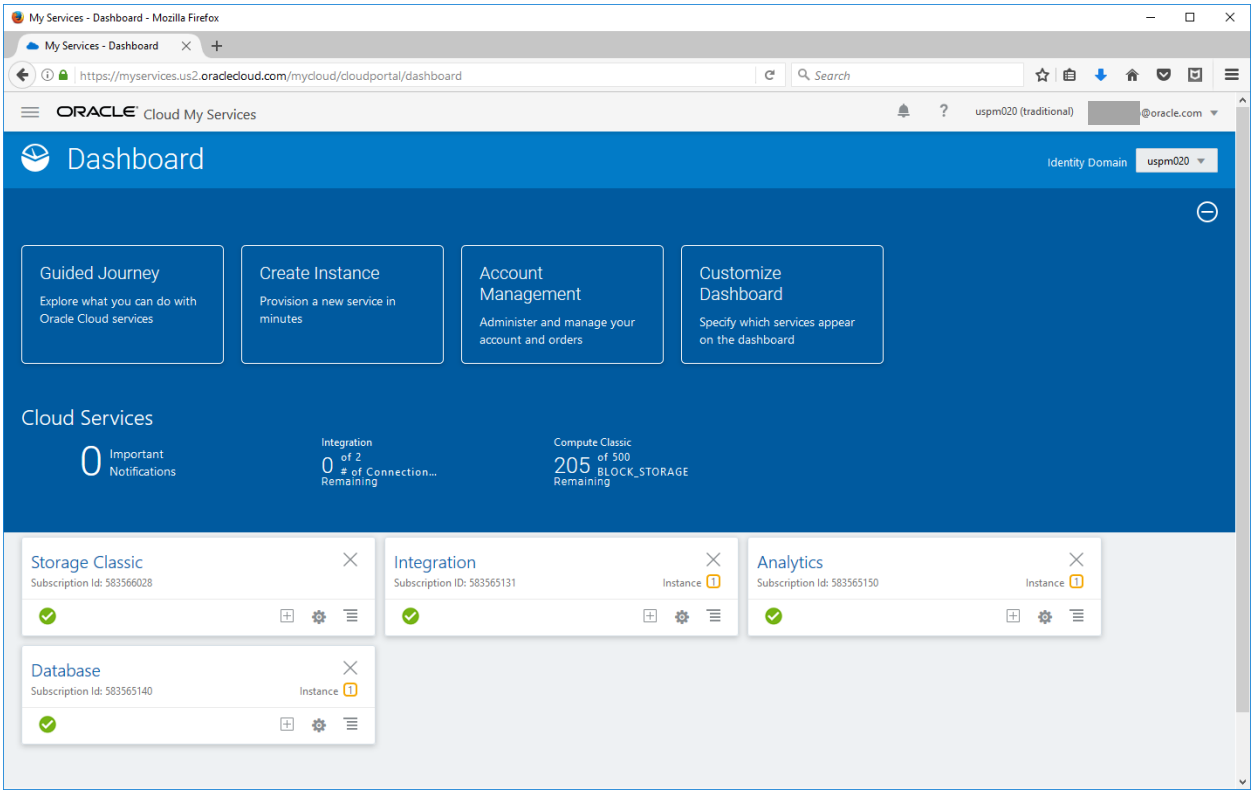
Log into your Oracle Cloud account at cloud.oracle.com. We will be using a Traditional Cloud Account for this exercise.



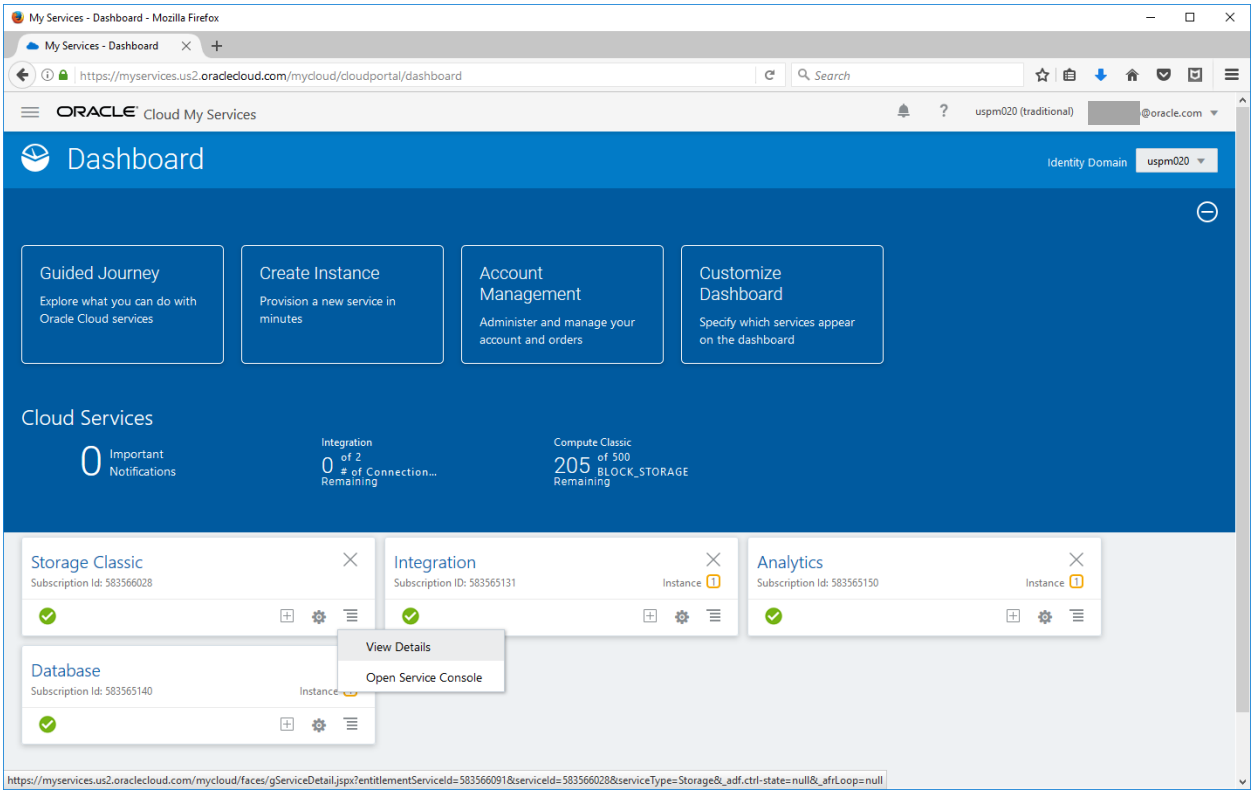
Enter your identity domain and then enter your login and password on the next screen.



After logging in, you will see your Dashboard for managing cloud services. You can click on Customize Dashboard to choose which services will appear.

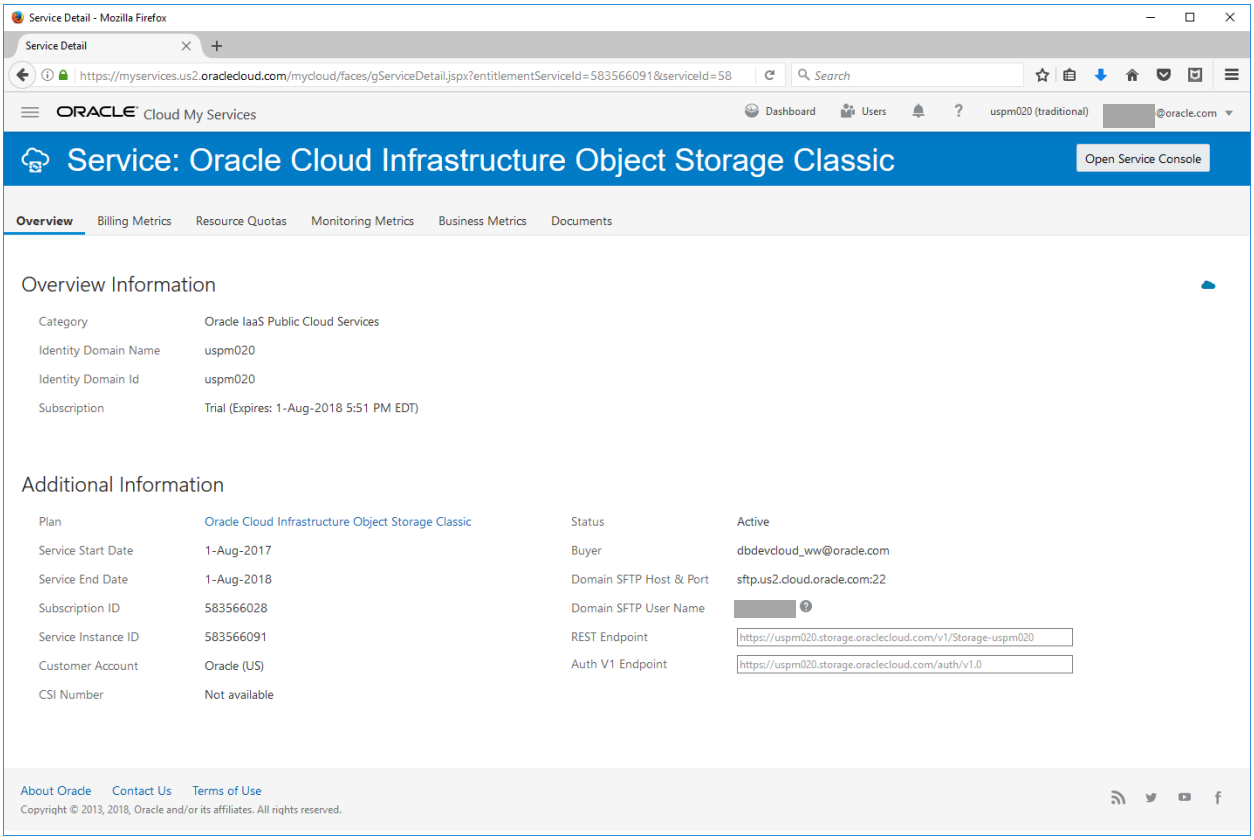


Before creating a DBCS instance, we will create a storage container for backups. Note that this step is optional, as there are several other backup mechanisms available for DBCS. Click the action button on the Storage Classic widget and select View Details.

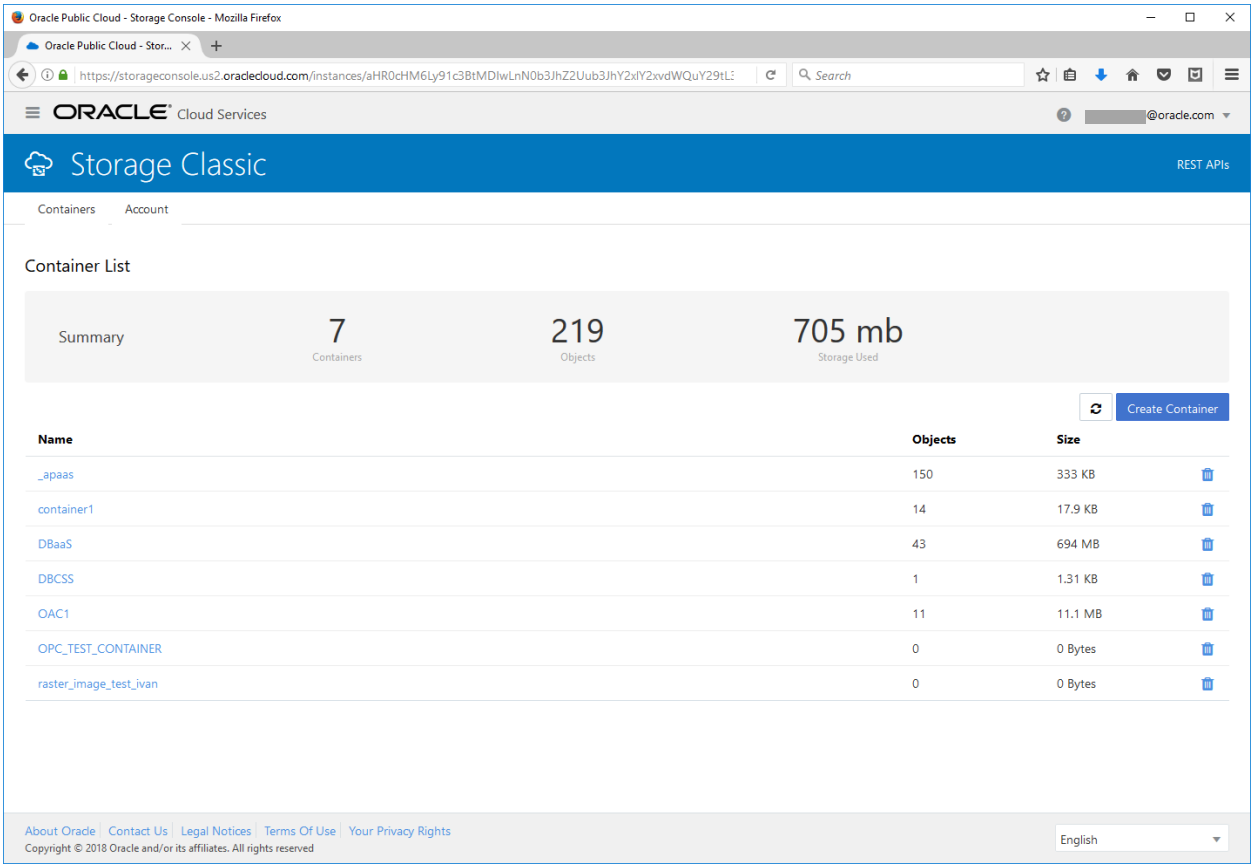


The Details page for your Storage Classic services will appear. Note the REST Endpoint URL, which will be needed later to setup your DBCS backup.

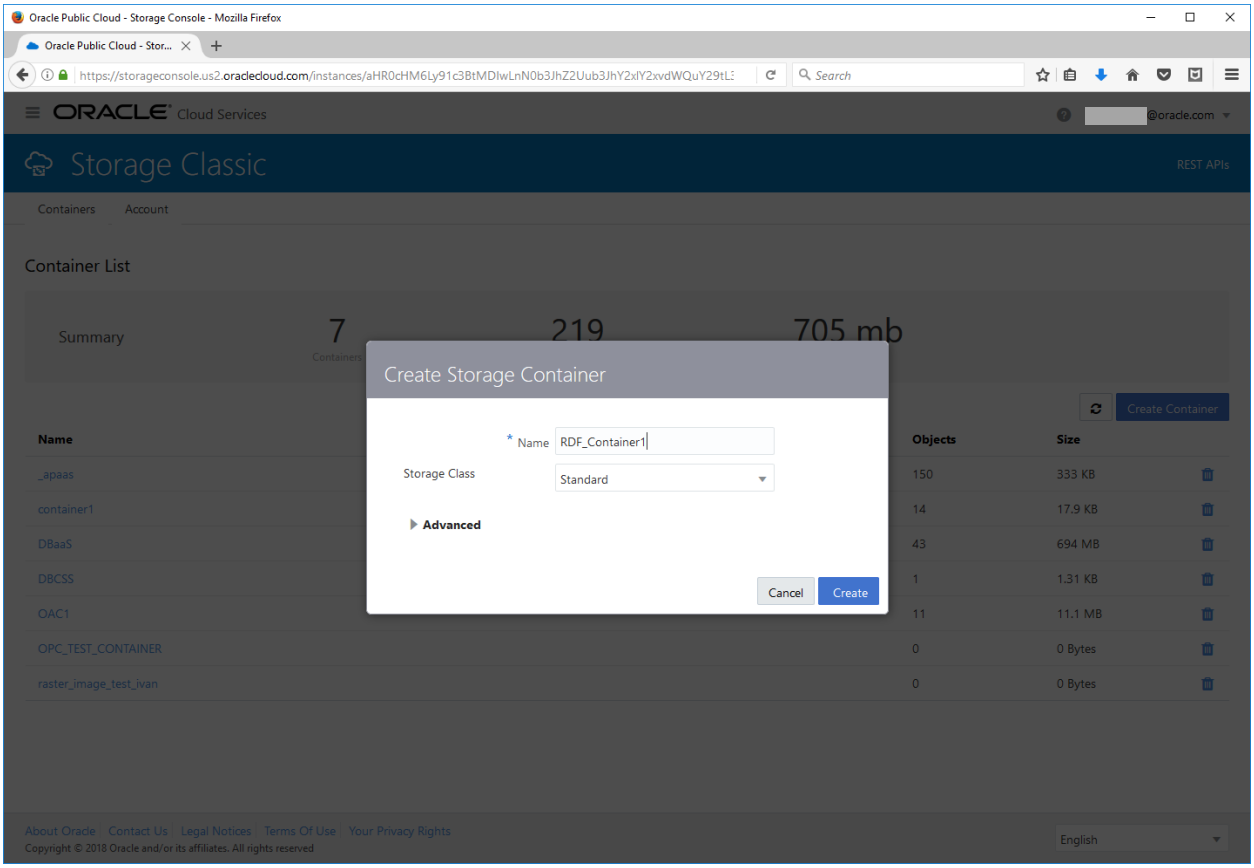
Click Open Service Console to bring up the Storage Containers page.

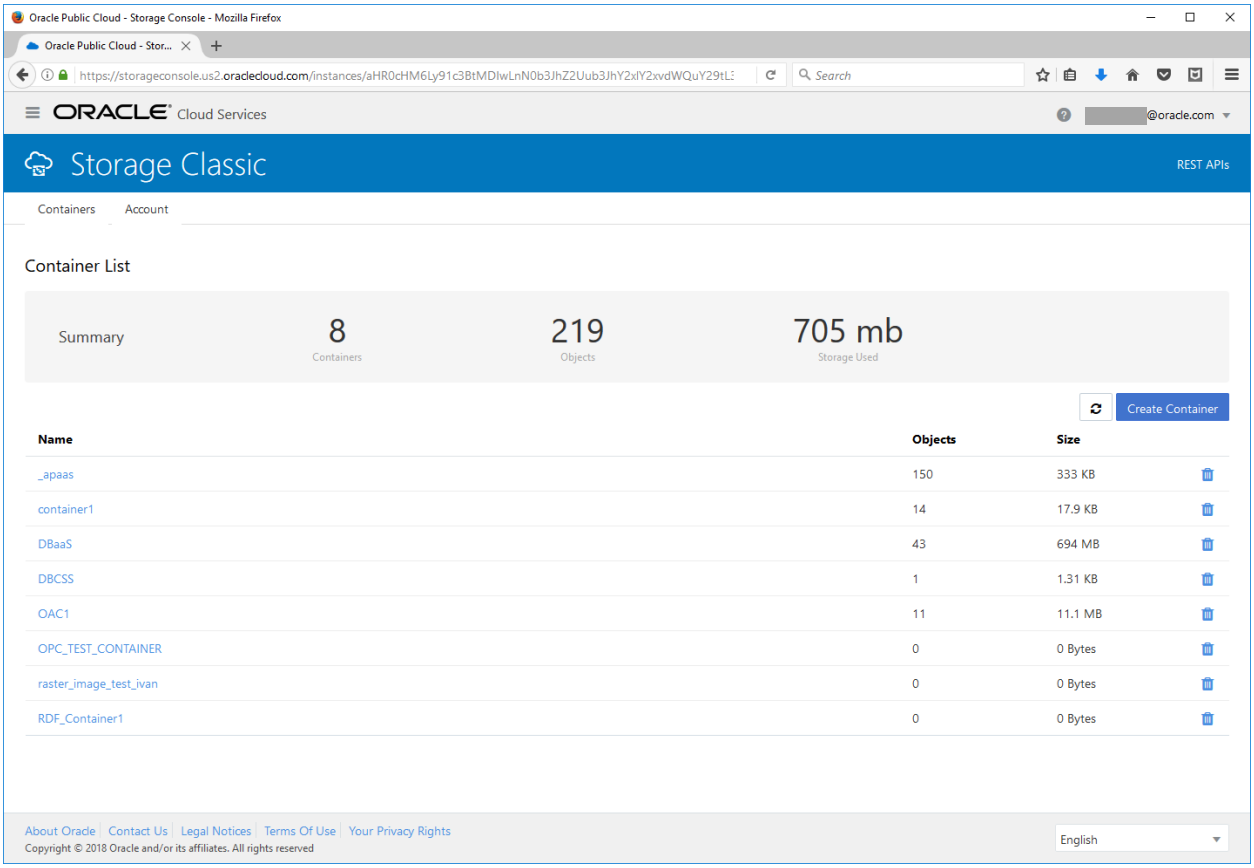


Click Create Container to create a new storage container.

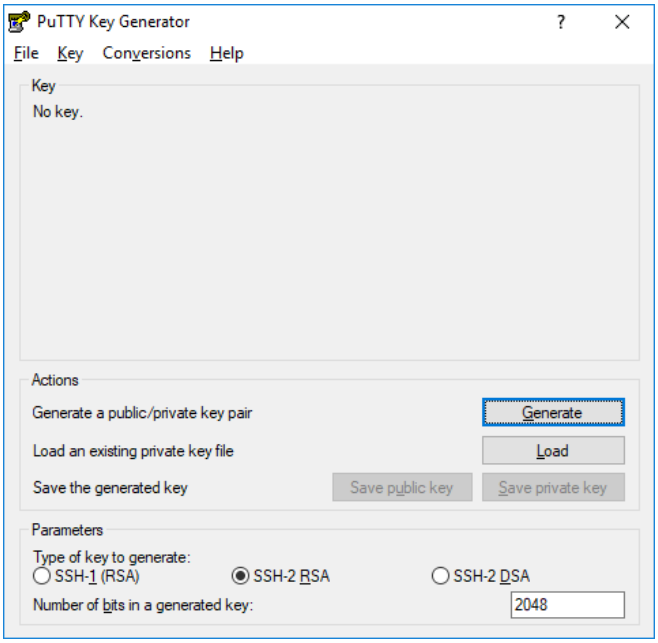


Enter a name for your storage container and click Create. We will name the container RDF_Container1 in this case. RDF_Container1 should be visible in your Container List afterwards.

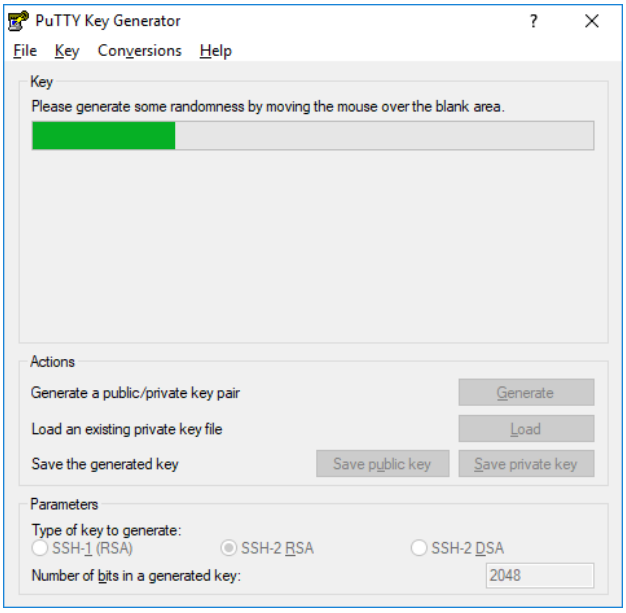

Oracle Public Cloud - Storage Console - Mozilla Firefox
Oracle Public Cloud - Stor...
https://storageconsole.us2.oraclecloud.com/instances/aHR0cHM6Ly91c3BtMDIwLnN0b3JhZ2Uub3JhY2xlY2xvdWQuY29tL
Search
ORACLE Cloud Services
@oracle.com
Storage Classic
REST APIs
Containers
Account
Container List
Summary
7
Containers
219
705 mb
Create Container
Name
Objects
Size
_apaas
150
333 KB
container1
14
17.9 KB
DBaaS
43
694 MB
DBCSS
1
1.31 KB
OAC1
11
11.1 MB
OPC_TEST_CONTAINER
0
0 Bytes
raster_image_test_ivan
0
0 Bytes
Create Storage Container
* Name
RDF_Container1
Storage Class
Standard
Advanced
Cancel
Create
About Oracle
Contact Us
Legal Notices
Terms Of Use
Your Privacy Rights
Copyright © 2018 Oracle and/or its affiliates. All rights reserved
English



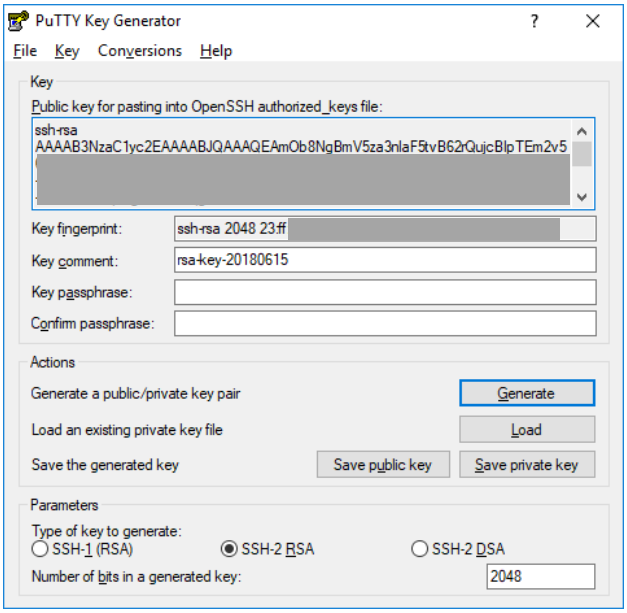
In the next step, we will create a DBCS instance. As a prerequisite, we will need to generate an SSH public/private key pair. You can use PuTTYgen for this on Windows. Open PuTTYgen and click Generate.



Follow the instructions to generate randomness.

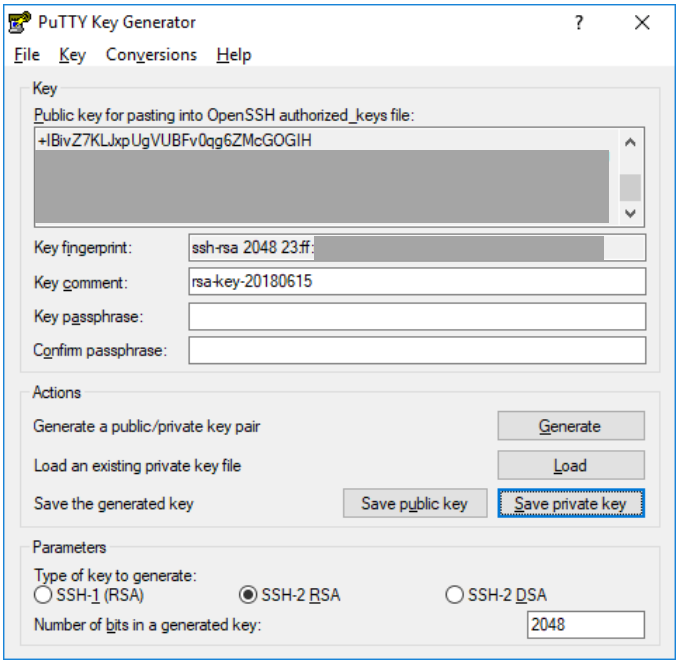


After generation, create a <key_name>.pub file and use Notepad or a similar text editor to copy and paste the entire contents of the Public key text into the myPubKey.pub file (be sure to scroll down to get everything). Save the <key_name>.pub file.



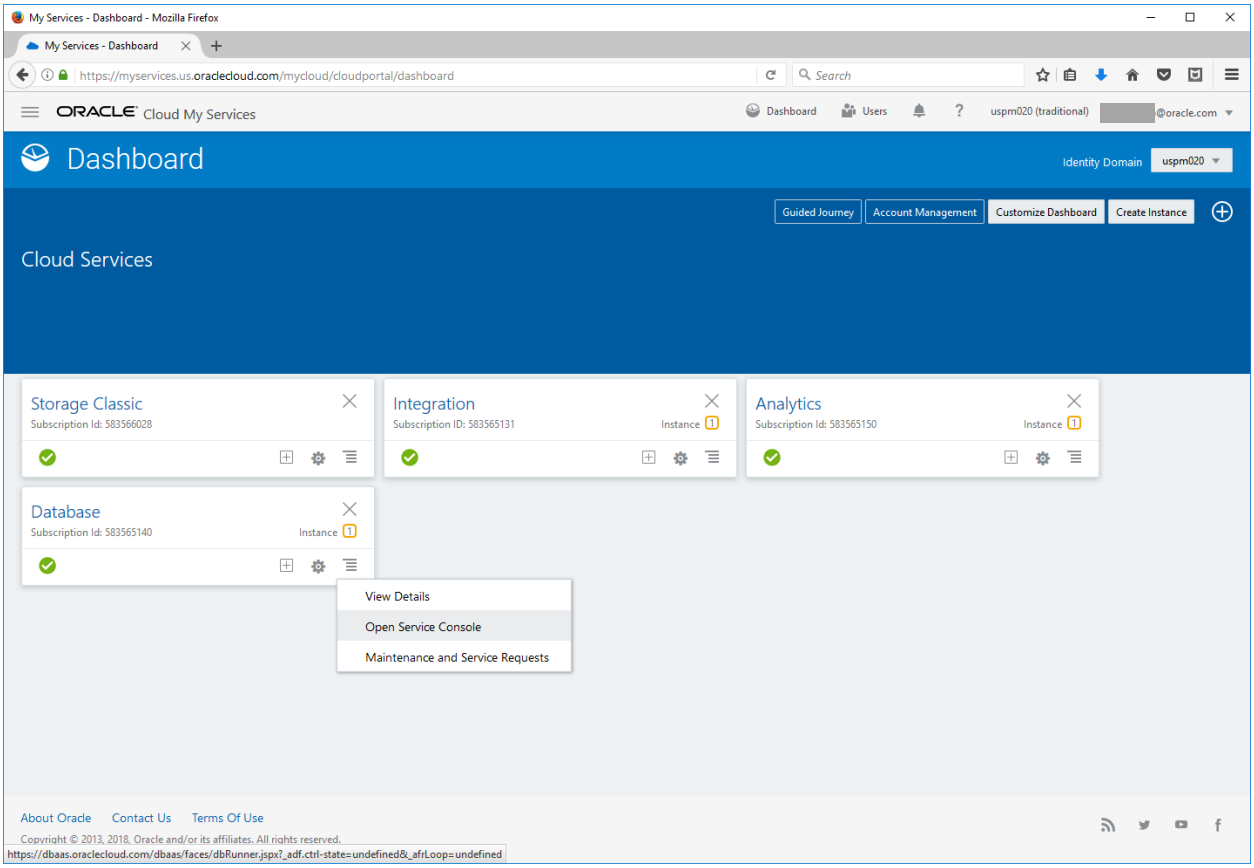
Click Save private key. You can optionally enter a passphrase that will need to be entered whenever the private key is used. This will save your private key in .ppk format.

In addition, click Conversions and select Export OpenSSH Key to also save the key in OpenSSH format.

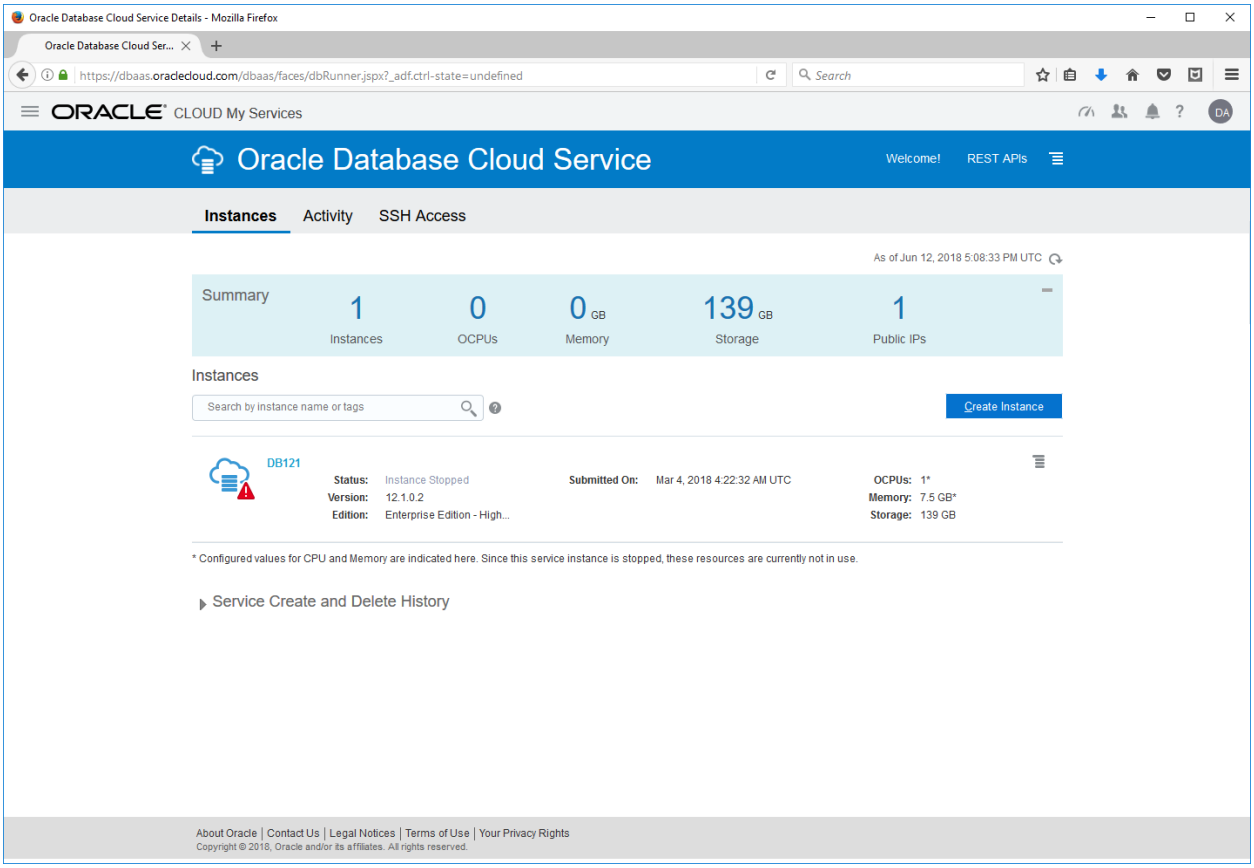


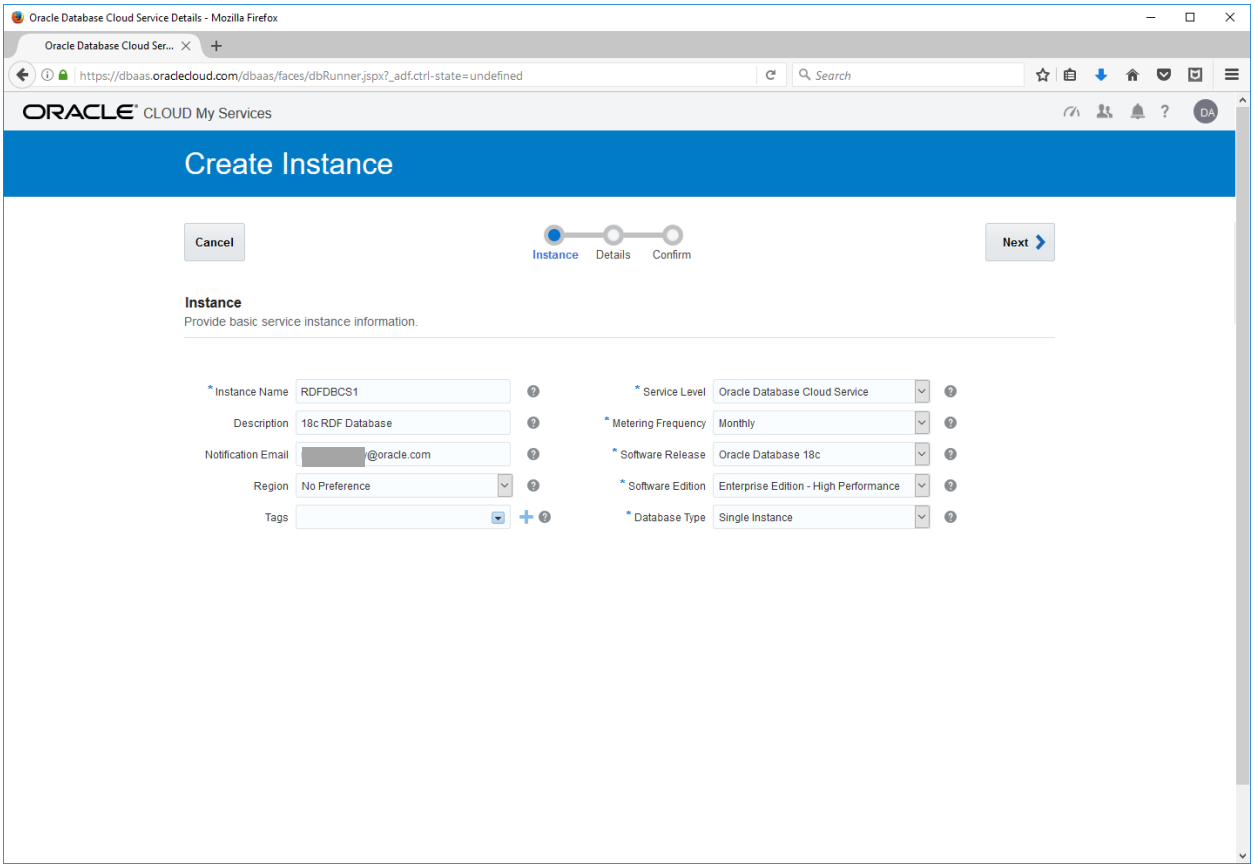
After this, you should have created three files: <key_name>.pub, <key_name>.ppk and <key_name>.openssh.

We will create the DBCS instance in the next step. First, open your cloud dashboard. In the Database widget, click Action and select Open Service Console.

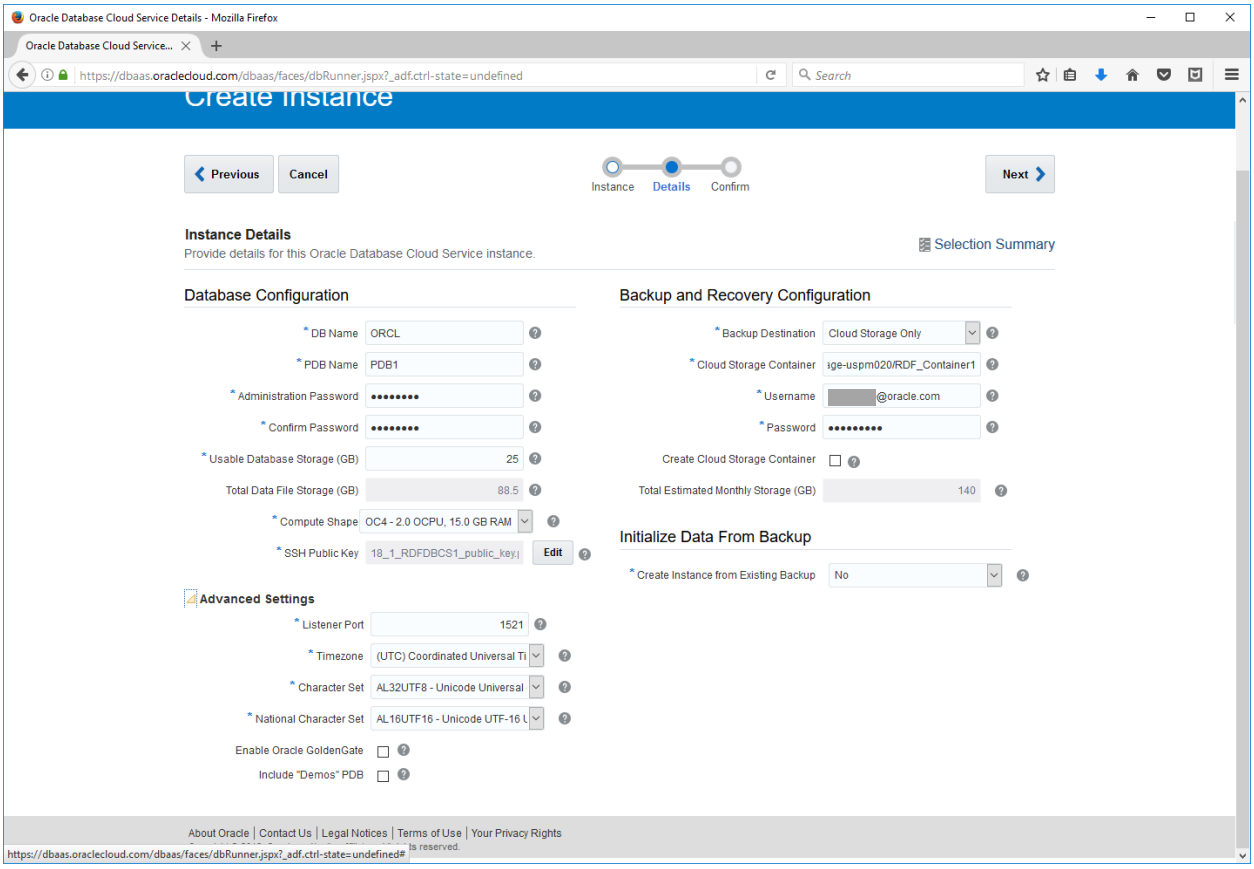


You will see any existing DBCS instances that you have. Click Create Instance to create a new DBCS instance.

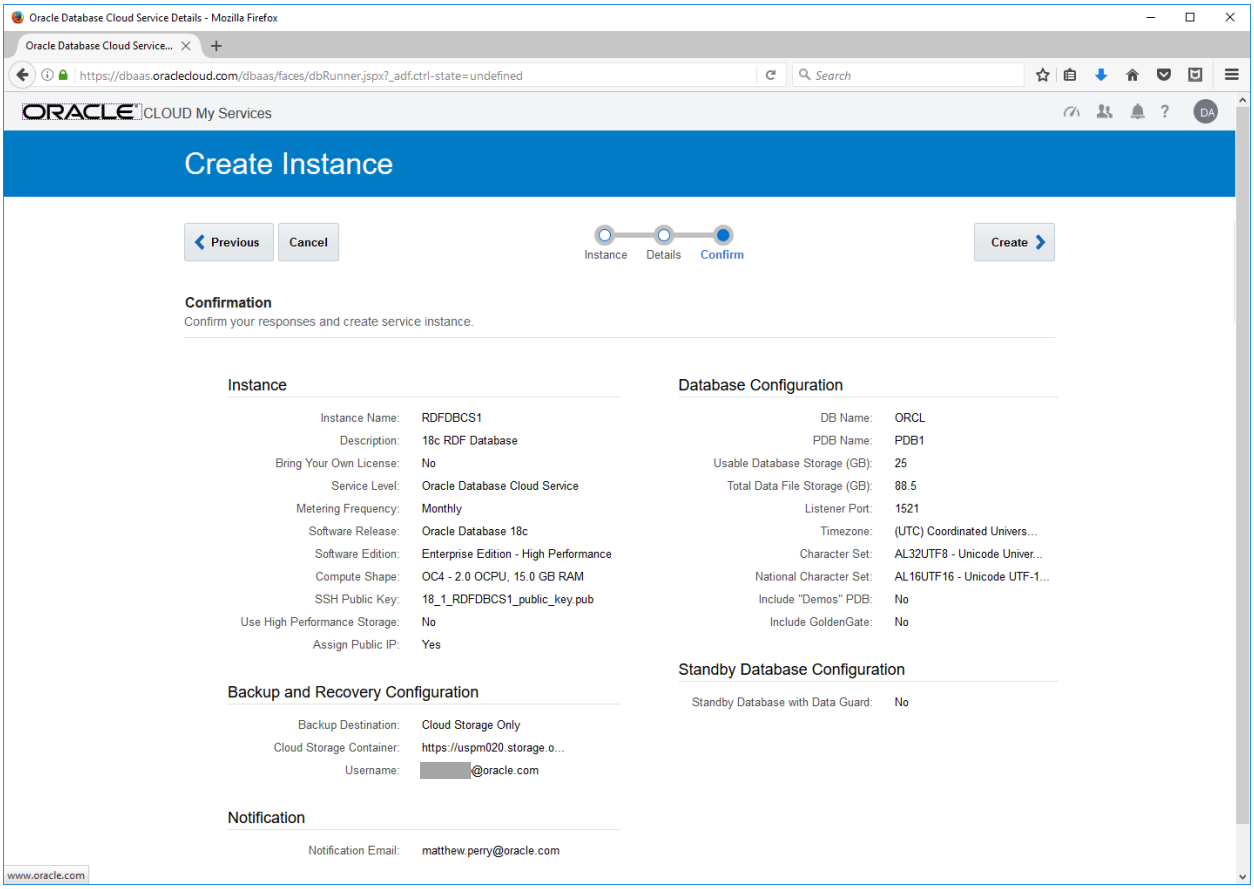

Oracle Database Cloud Service Details - Mozilla Firefox
Oracle Database Cloud Ser...
https://dbaas.oraclecloud.com/dbaas/faces/dbRunner.jspx?_adf.ctrl-state=undefined
Search
ORACLE CLOUD My Services
Oracle Database Cloud Service
Welcome!
REST APIs
Instances
Activity
SSH Access
As of Jun 12, 2018 5:08:33 PM UTC
Summary
1
Instances
0
OCPUs
0 GB
Memory
139 GB
Storage
1
Public IPs
Instances
Search by instance name or tags
Create Instance
DB121
Status: Instance Stopped
Version: 12.1.0.2
Edition: Enterprise Edition - High...
Submitted On: Mar 4, 2018 4:22:32 AM UTC
OCPUs: 1*
Memory: 7.5 GB*
Storage: 139 GB
* Configured values for CPU and Memory are indicated here. Since this service instance is stopped, these resources are currently not in use.
Service Create and Delete History
About Oracle | Contact Us | Legal Notices | Terms of Use | Your Privacy Rights
Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



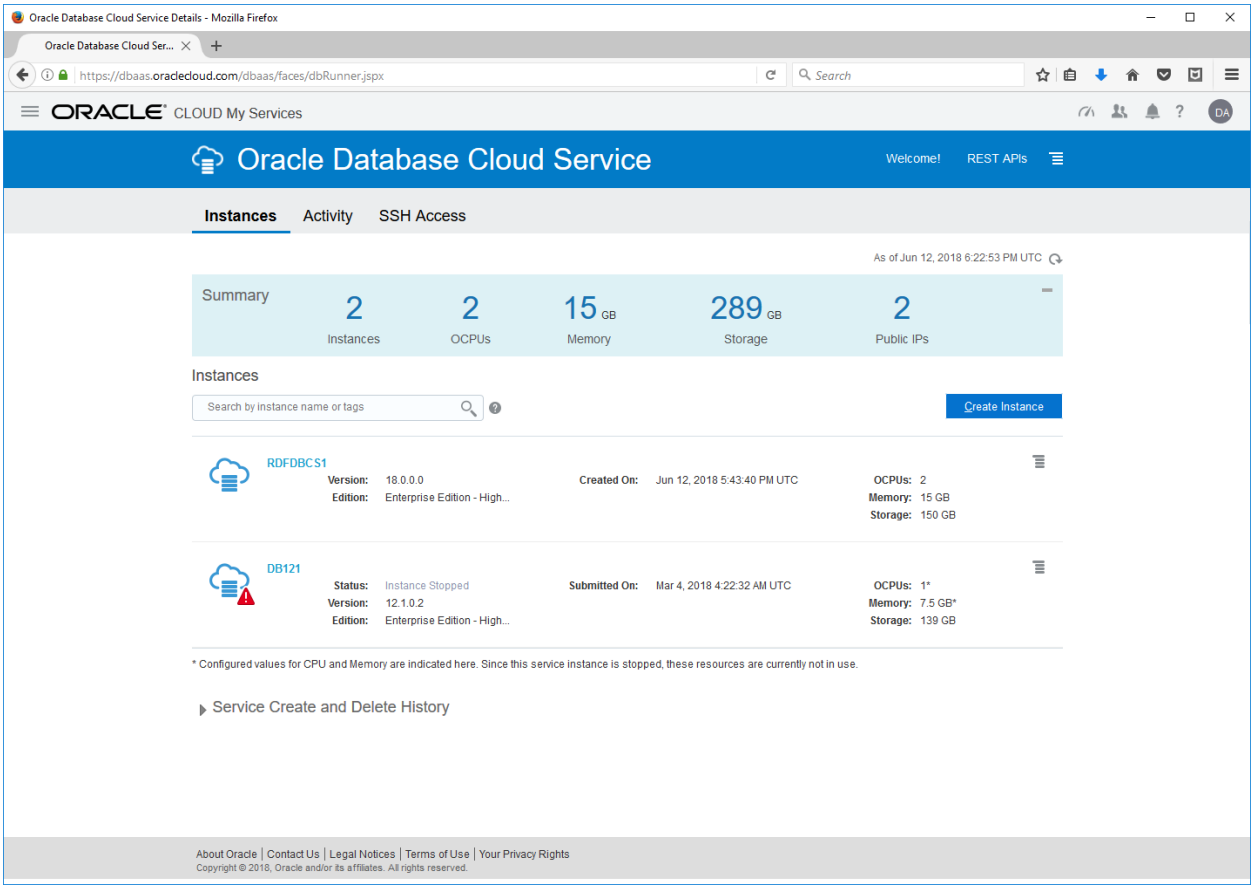
Fill in the form with your desired values. We will be using Oracle Database version 18c. You will need either Enterprise Edition - High Performance or Enterprise Edition - Extreme Performance to get the Spatial and Graph option, which includes RDF Semantic Graph.



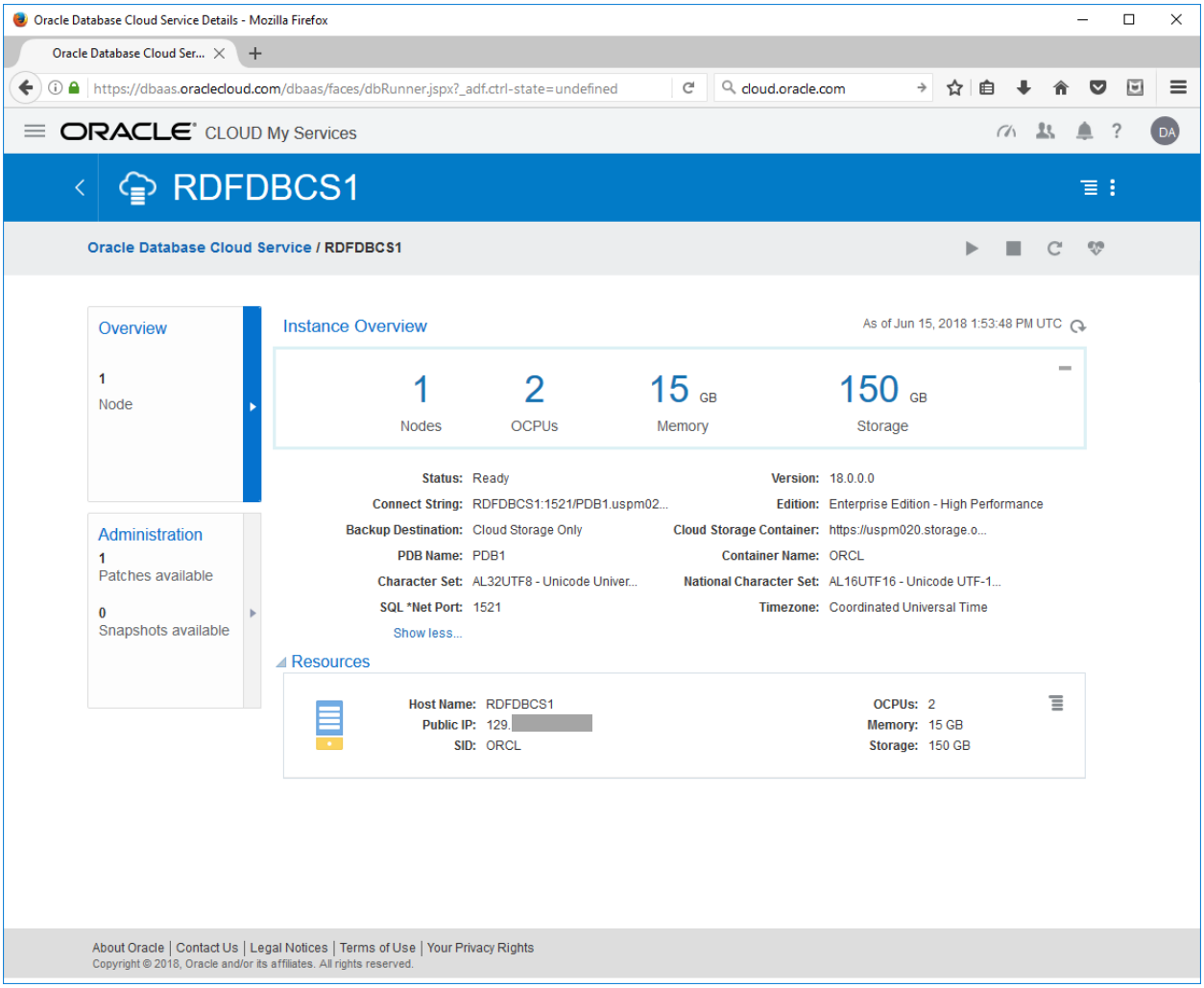
Fill in the Instance Details. Choose the <key name>.pub file to enter the SSH public key you created earlier, and enter <REST endpoint URL>/<Container Name> for the Cloud Storage Container you created earlier. Click Next to bring up the confirmation page.



Click Create to create the instance.



After about 30 minutes, the DBCS instance will be ready to use. You can click the instance name (RDFDBCS1 in this case) to see details.

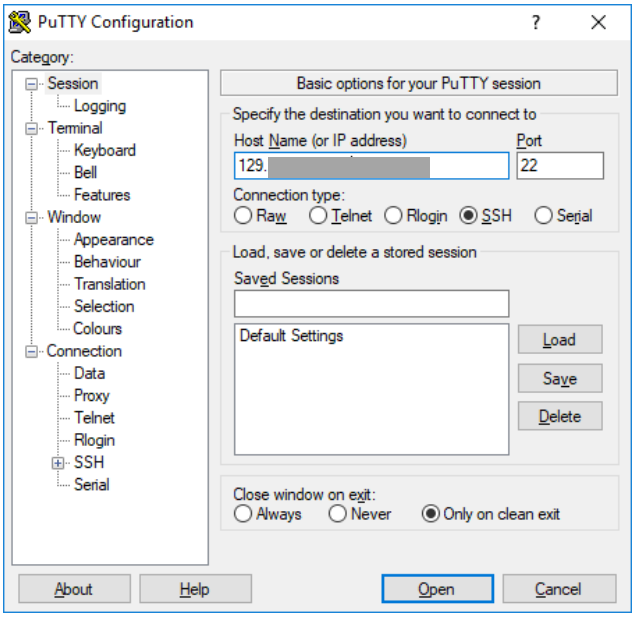


Note the Public IP, Connect String and SQL *Net port. You will need this information to make a connection to the DBCS instance.

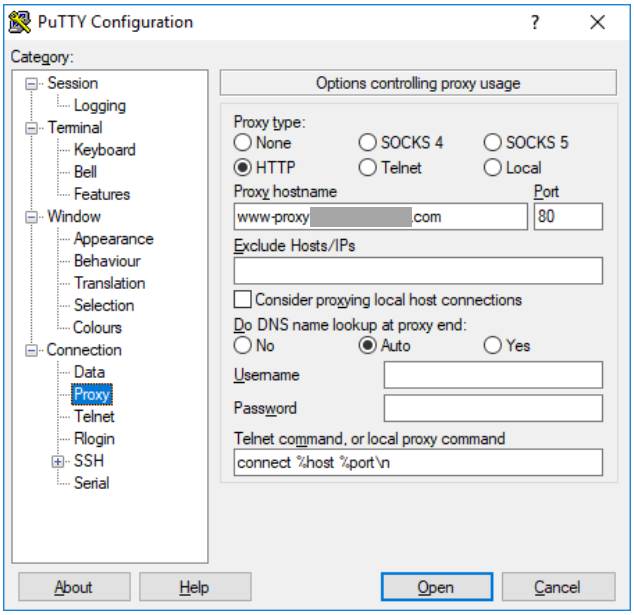
4 CONNECTING TO THE DBCS INSTANCE WITH PUTTY

Next, we will use PuTTY to make an SSH connection to our DBCS instance, start SQL *Plus, and check the RDF Semantic Graph installation in our database.

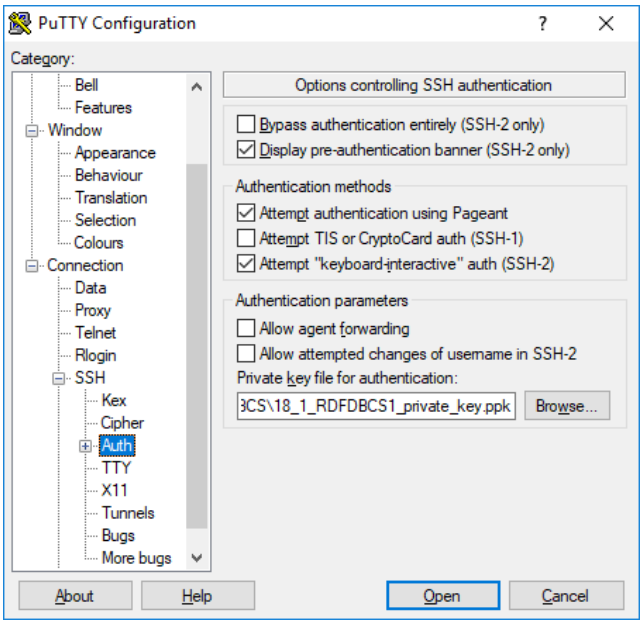
Open PuTTY and click Session in the tree on the left-hand side. Enter the public IP for your DBCS instance for Host Name and enter 22 for Port.



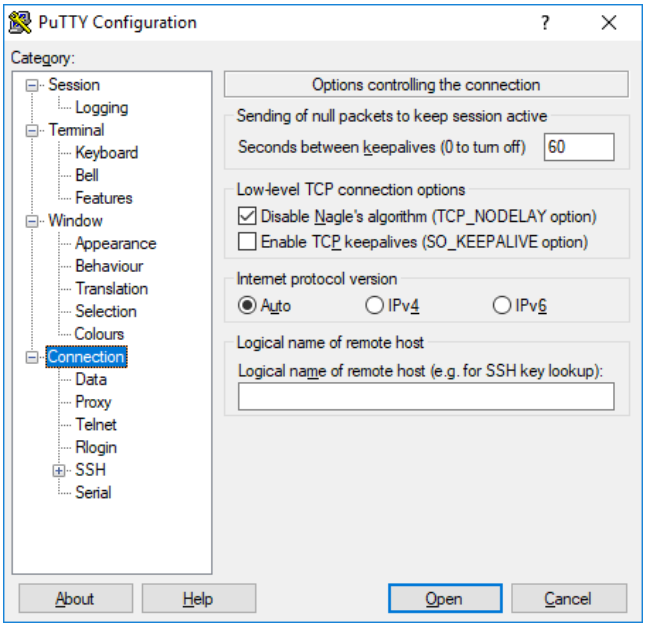
If necessary, expand Proxy and configure any proxy settings for your network.



Expand SSH and click Auth. Click Browse and select your .ppk private key file for Private key file for authentication.

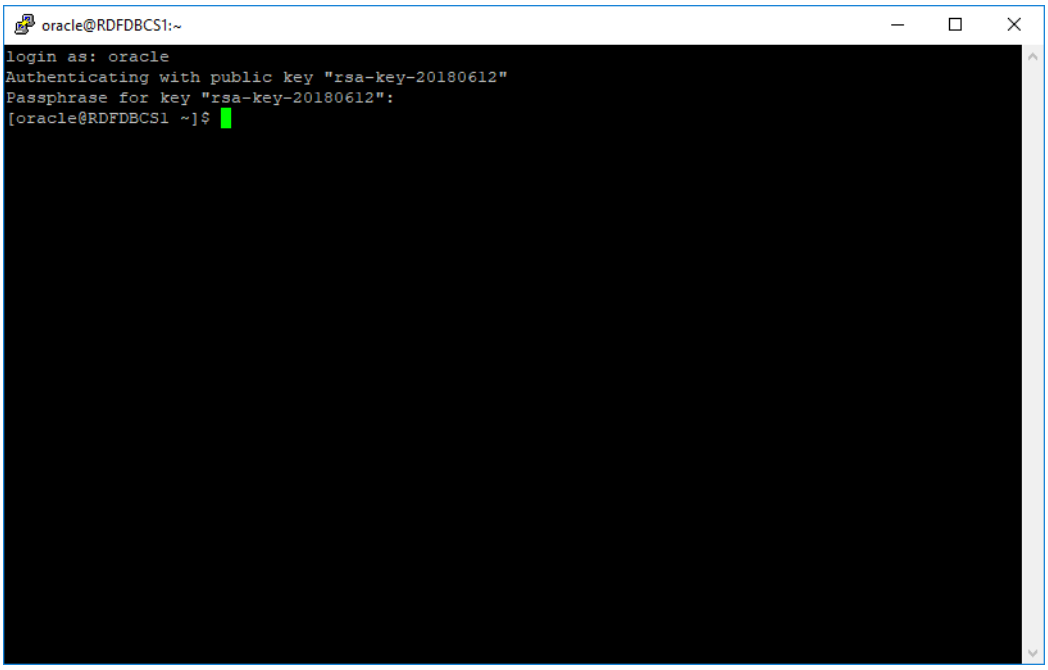


You can also click Connection and specify a non-zero keepalive value to prevent the SSH session from ending due to inactivity.



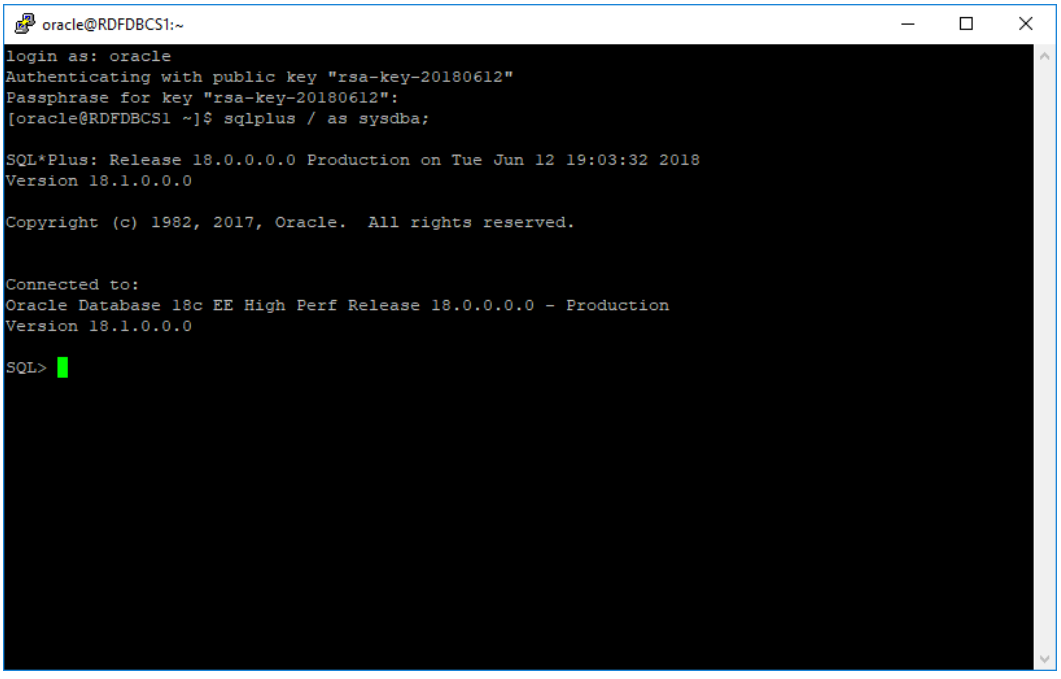
Tip: it is a good idea to click Session and save this PuTTY configuration for future use.

Now, click Open to start the SSH session. Enter oracle for the login name. Then enter your passphrase if you used one when creating your public/private key pair.

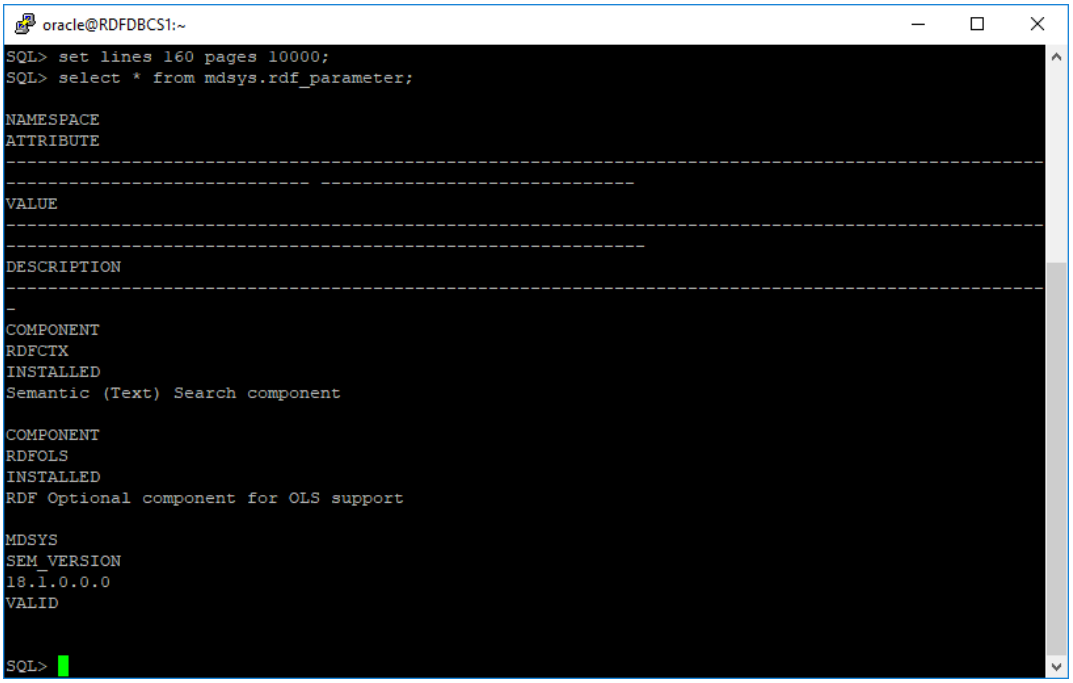


Start SQL*Plus with the following command:

sqlplus / as sysdba



Execute the query `SELECT * FROM MDSYS.RDF_PARAMETER` to verify the RDF Semantic Graph installation. You should see a valid 18.1.0.0.0 installation.

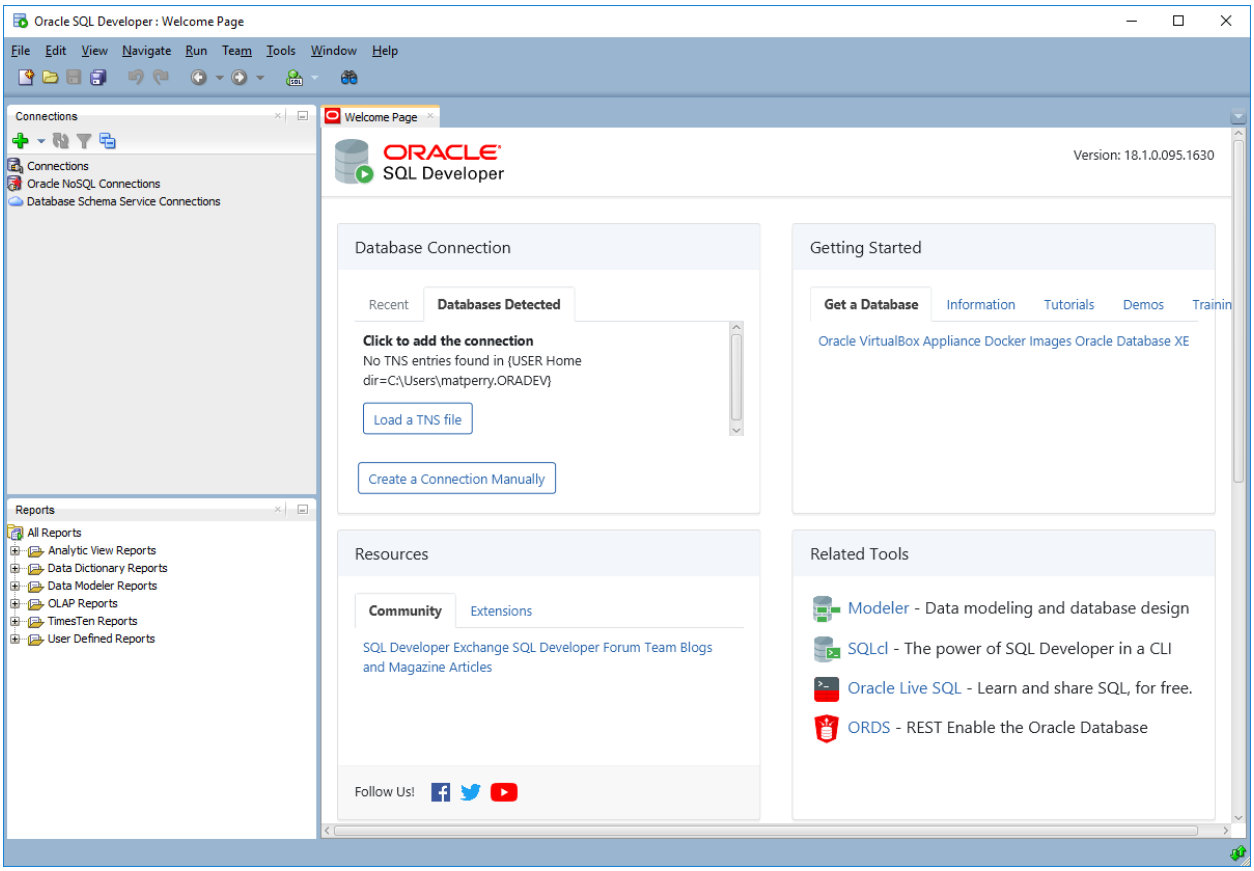


We have now verified that RDF Semantic Graph is properly installed on our DBCS instance.

5 USING SQL DEVELOPER FOR RDF SEMANTIC GRAPH ON THE DBCS INSTANCE

We will now use SQL Developer to setup and configure our RDF Semantic Graph installation.

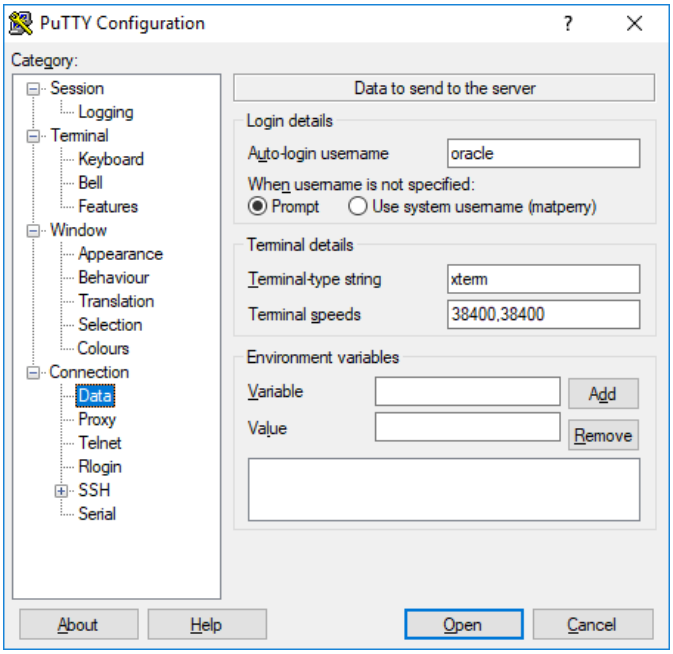
Open SQL Developer. We are using version 18.1.0.095 in this example.



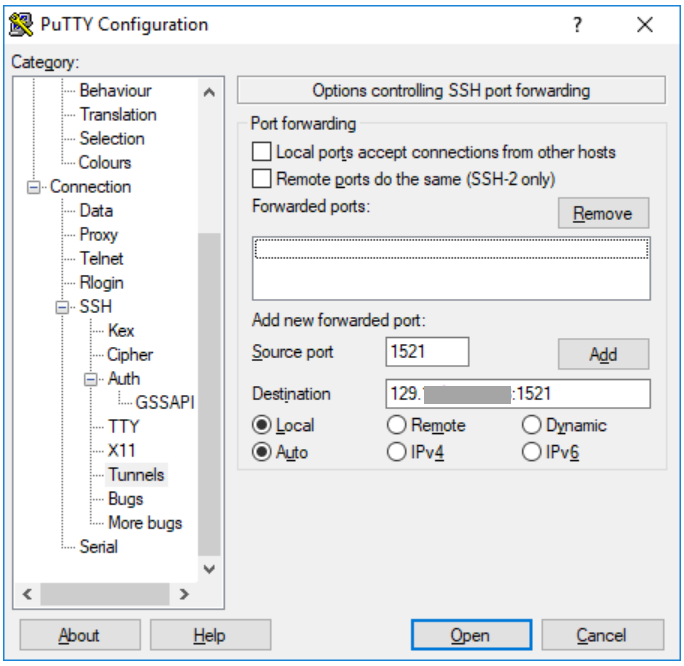
5.1 Connecting to the DBCS Instance

First, we need to create a connection to our DBCS instance. There are a few ways to create a connection. In this tutorial, we will create an SSH tunnel and configure port forwarding for our SQL*Net port. See the following Oracle Help Center article for more details on other ways to connect SQL Developer to a DBCS instance.

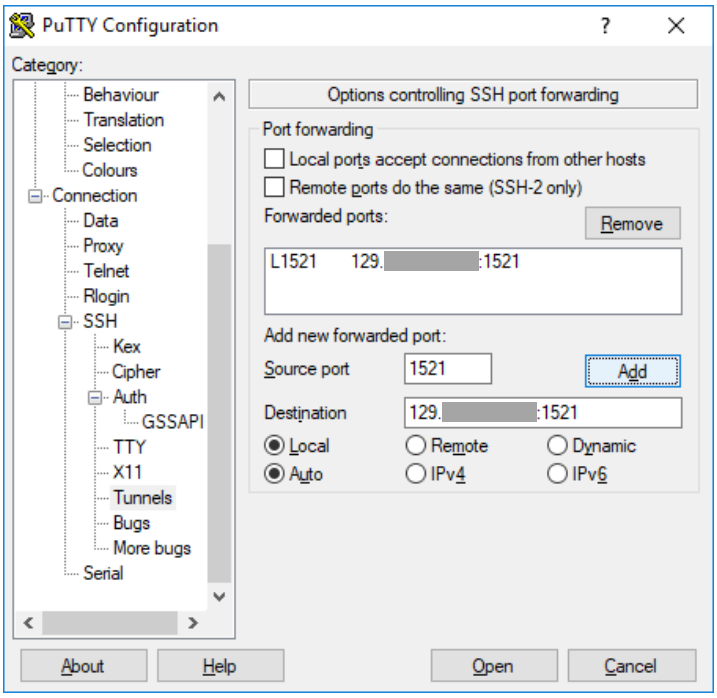
We will use PuTTY to create an SSH tunnel for the SQL*Net port (1521 in this example). Open PuTTY and repeat all the steps from before to configure an SSH connection to your DBCS instance. Next, click Data under Connection and enter oracle for Auto-login username.



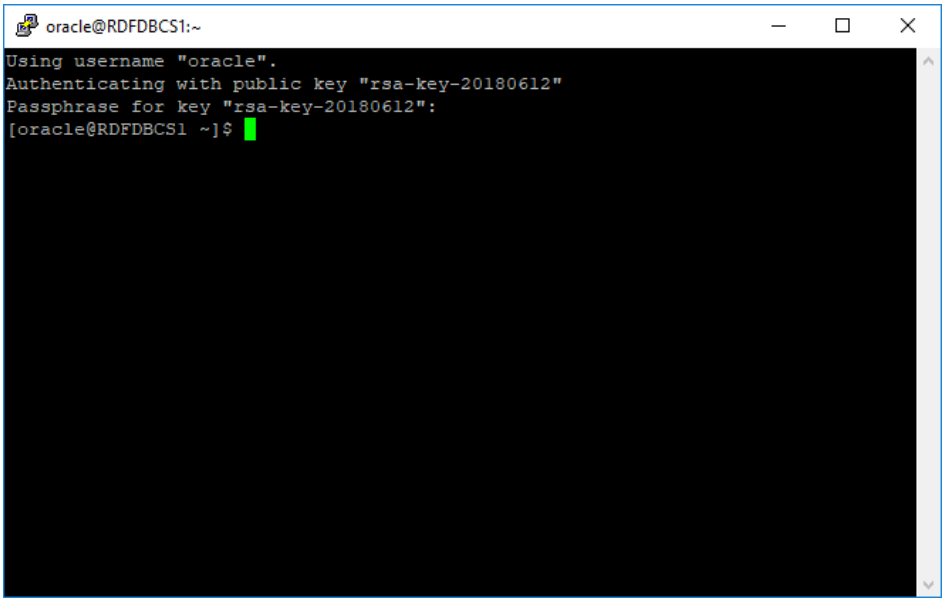
Click Tunnels and enter a Source port to tunnel and use the public IP and SQL*Net port for your DBCS instance for Destination. Click Add.



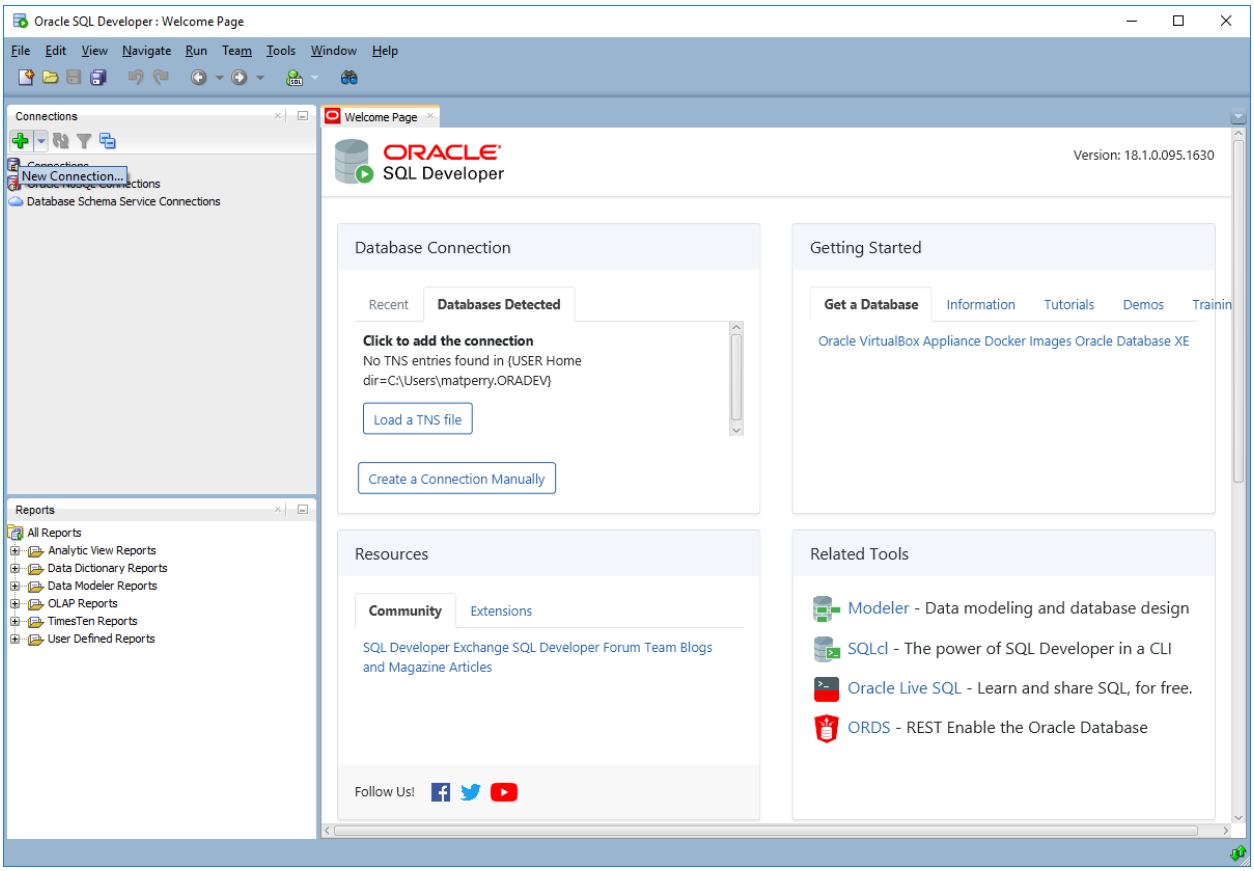
After the forwarded port has been added, click Open.



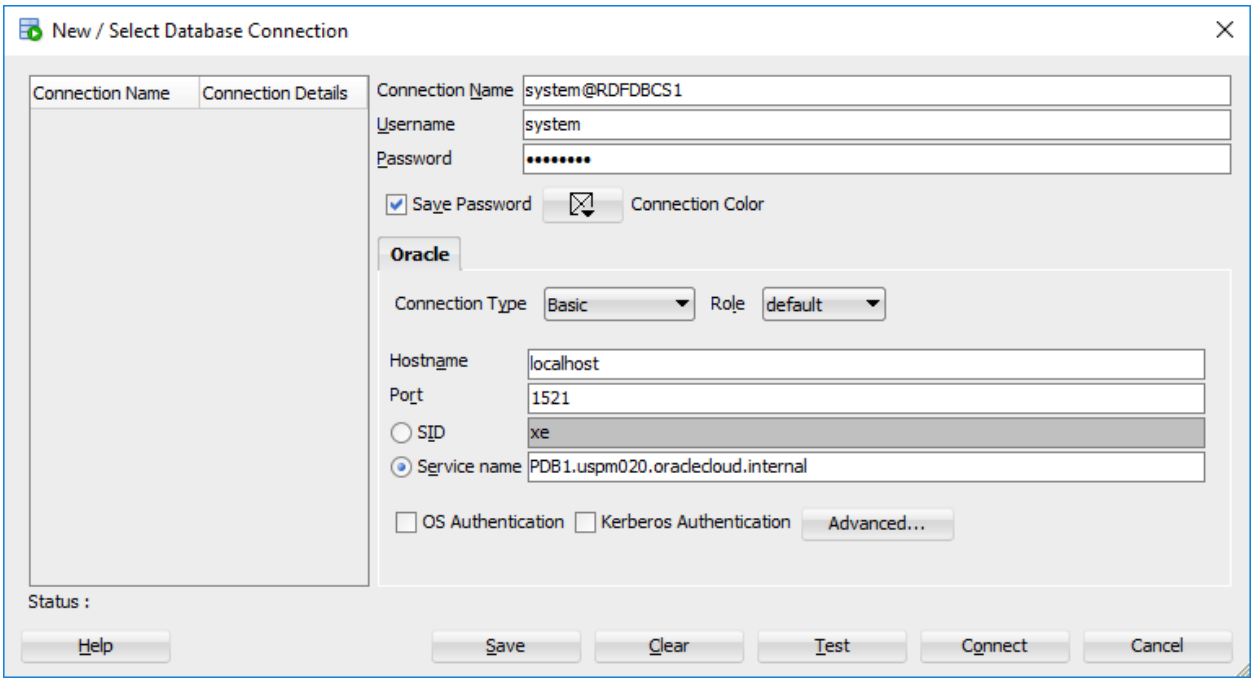
If applicable, enter the passphrase for your private key. Leave this window open.



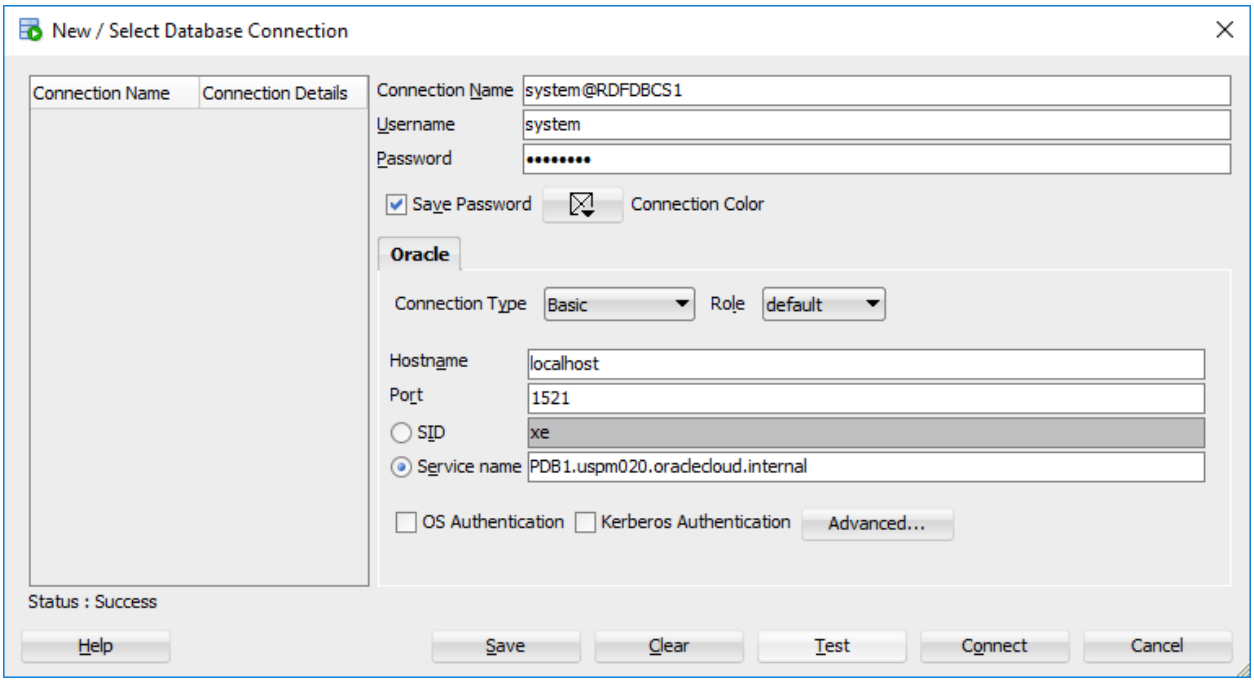
Now we will create a connection in SQL Developer that uses the SSH tunnel we created for port 1521. Click the green plus under Connections to create a new connection.



Choose a name for the connection. Use system as the Username and use the admin password for your DBCS instance. Use localhost for Hostname, the SQL*Net port from your DBCS instance for Port (1521 in this example), and the service name from your DBCS connection string (the portion of the connect string after <port>/).



Click Test to test the connection.

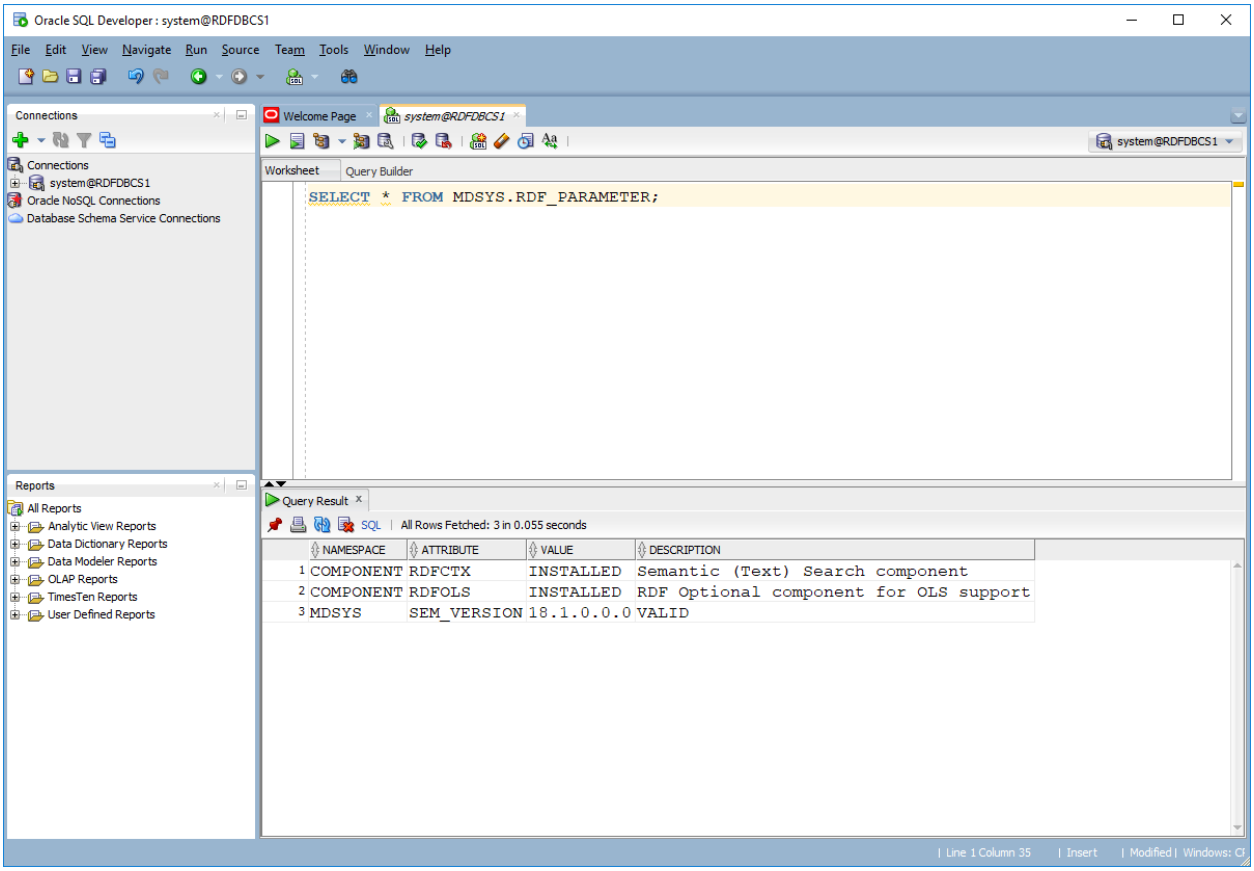


Save the connection if the test was successful, and click Connect to open the connection. This will open up a SQL Worksheet for the system connection.

Now we will verify the RDF Semantic Graph installation. Type the following query into your SQL Worksheet and click the green triangle to run the query.

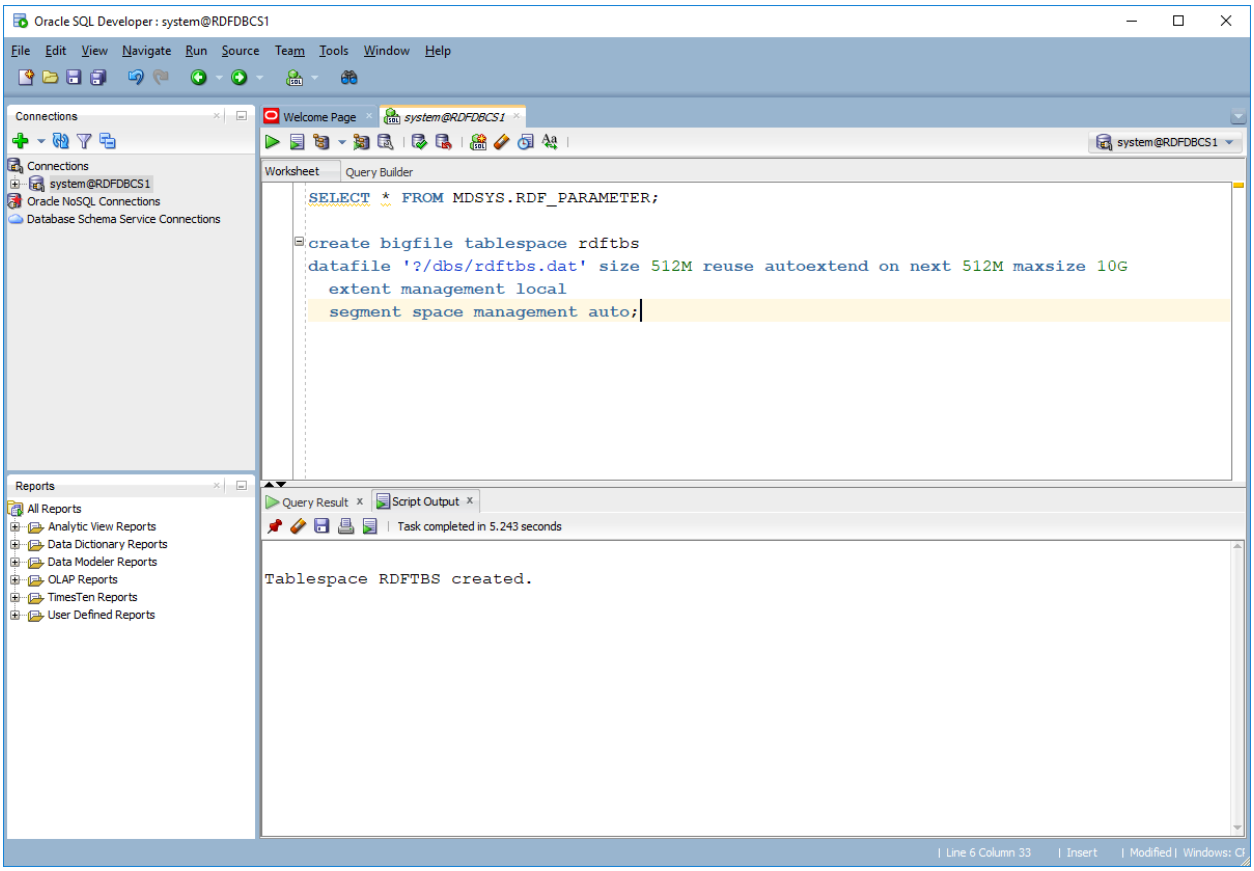
```
SELECT * FROM MDSYS.RDF_PARAMETER;
```

The query result should show a valid 18.1.0.0.0 installation of RDF Semantic Graph.



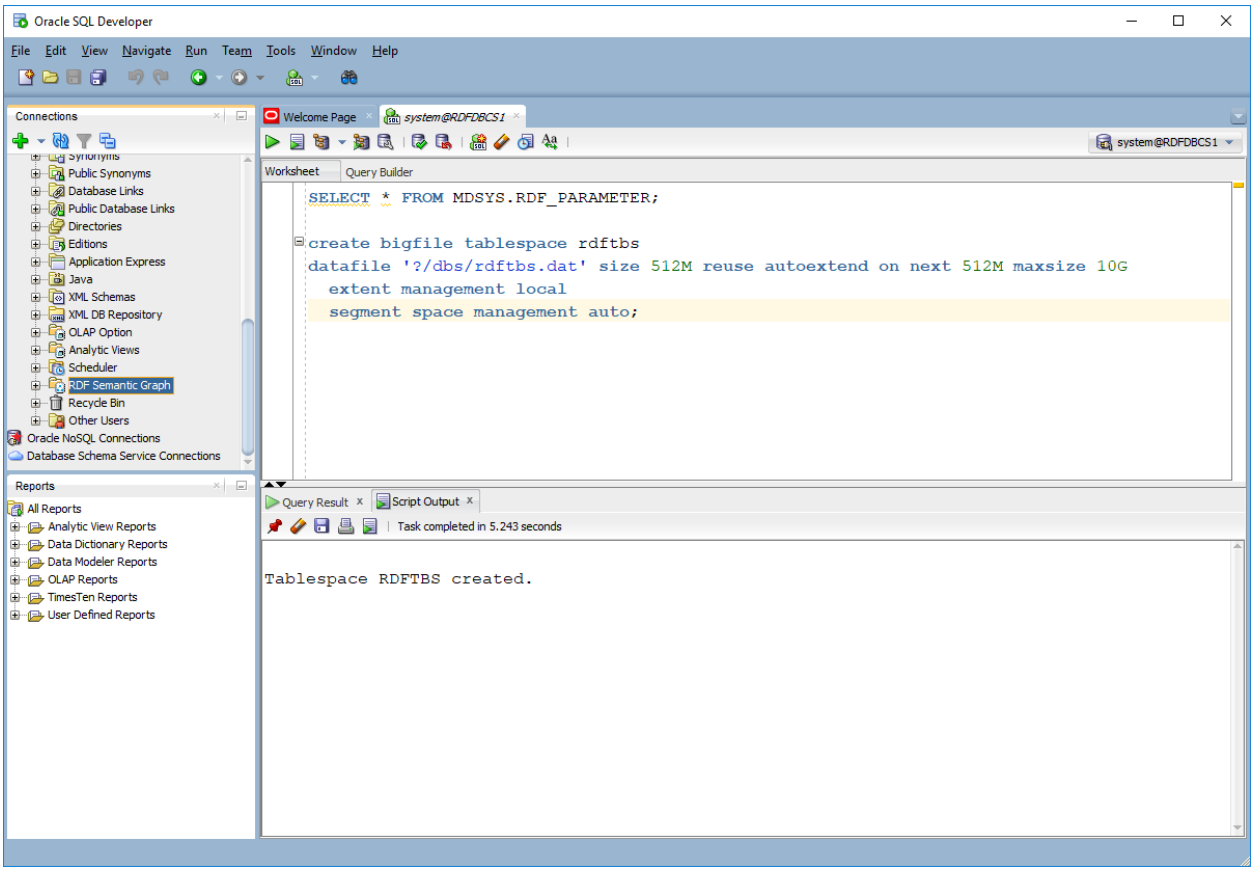
Next, we will create the Semantic Network to prepare the database for storing RDF data. As a prerequisite, we need to create a tablespace for the Semantic Network. We can do this by running the following SQL statement as system:

```
create bigfile tablespace rdftbs
datafile '?/dbs/rdftbs.dat' size 512M reuse autoextend on next 512M maxsize
10G
  extent management local
  segment space management auto;
```

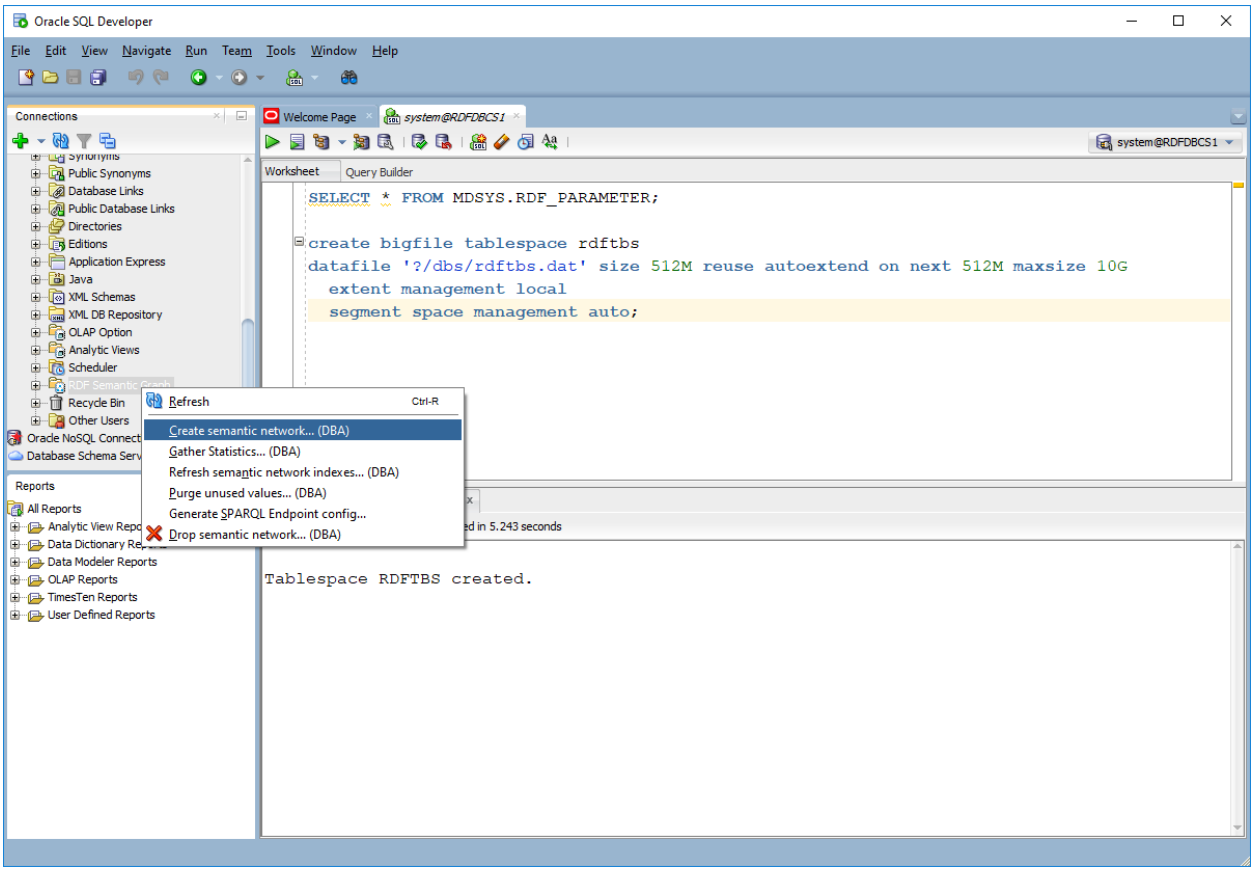


5.2 USING THE RDF SEMANTIC GRAPH PLUGIN

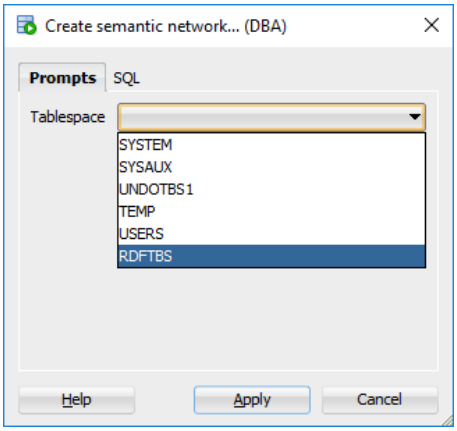
Now we can use the RDF Semantic Graph component of SQL Developer to create the Semantic Network. Expand the system connection by clicking the plus sign next to the connection name, and then scroll down to the RDF Semantic Graph component.

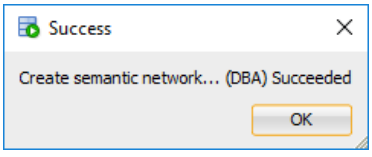


Right-click on RDF Semantic Graph and select Create Semantic Network.



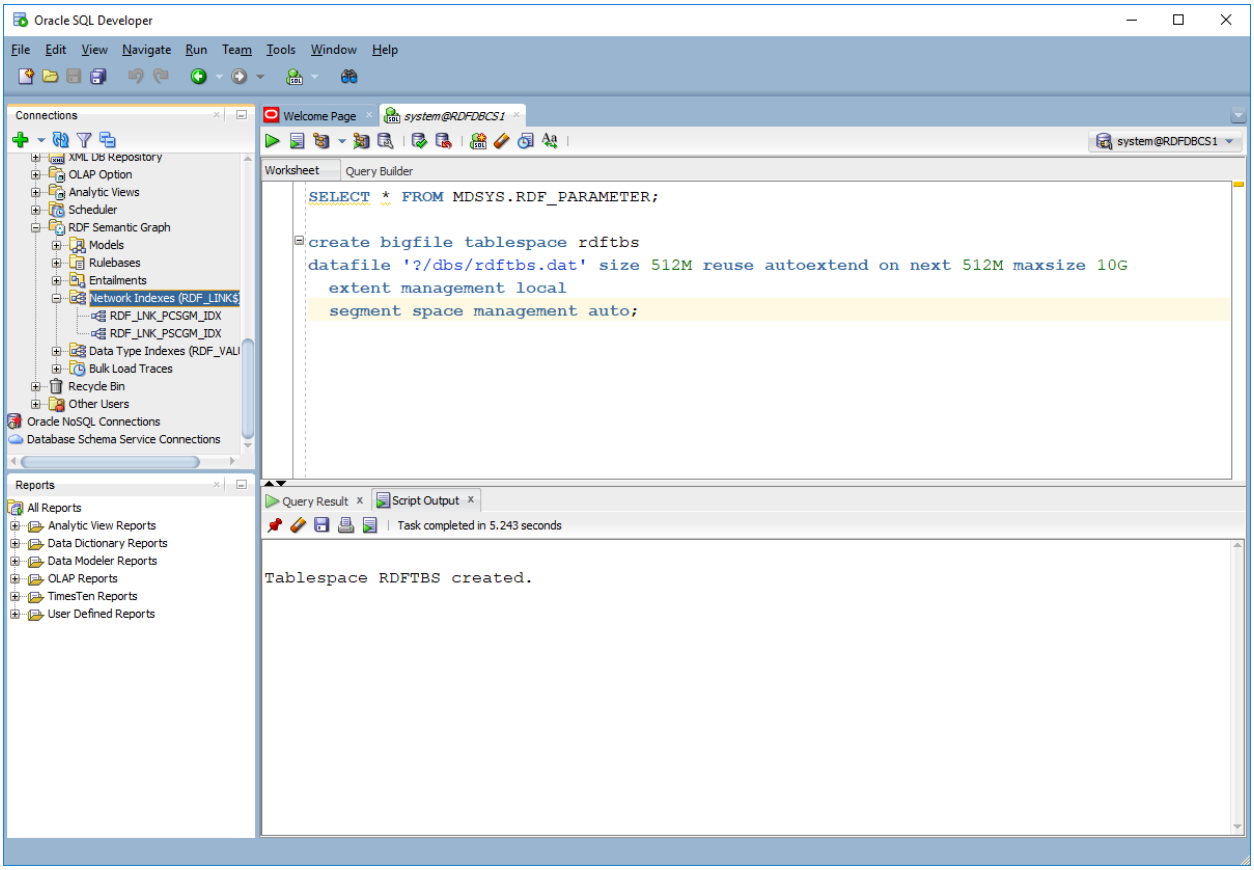
Use the drop down menu to select the tablespace that we created earlier and click Apply.





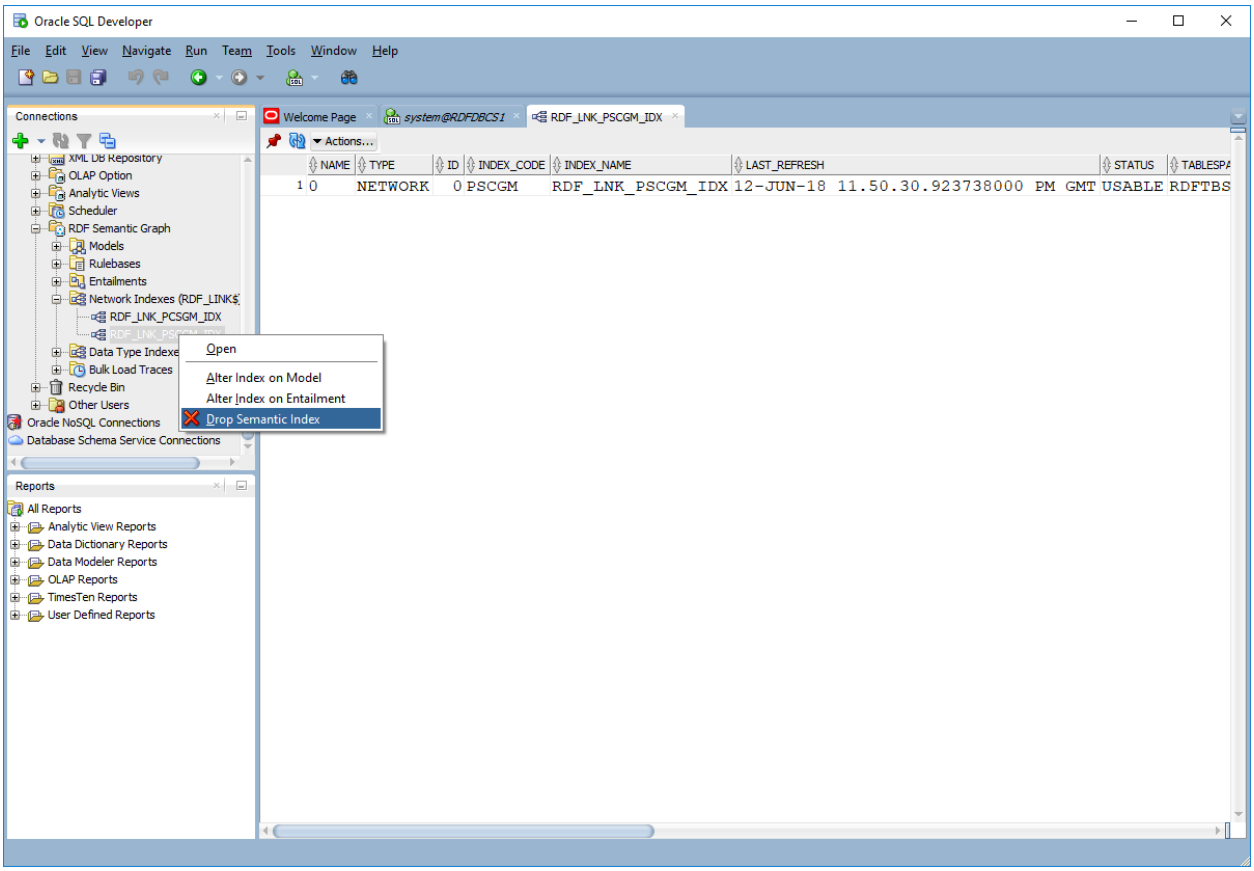
Next, we will make some changes to the default index configuration for better general-purpose query performance. Expand Network Indexes under RDF Semantic Graph to see index codes for the current network indexes. Each letter in an index code corresponds to a component of an RDF quad: S – subject, P – predicate, C – canonical object, G – graph, M – model. By default, two indexes are created: a

mandatory PCSGM unique index and a PSCGM index. This indexing scheme works very well when SPARQL triple patterns have constants in the predicate position, but this scheme may encounter performance problems if variables appear in the predicate position.

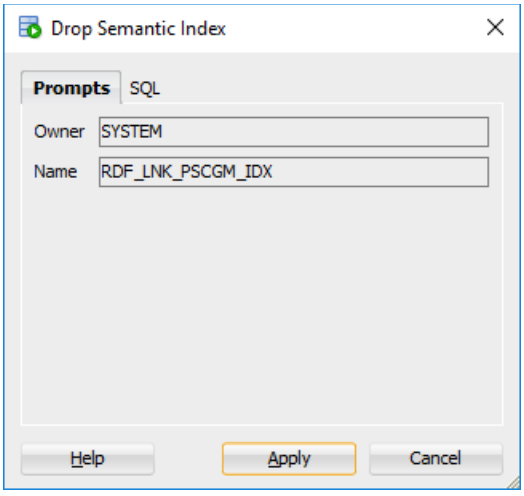


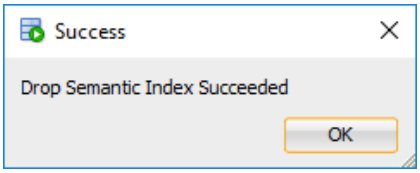
For a more general scheme, a three-index combination of PCSGM, SPCGM and CPSGM works well, so we will drop the PSCGM index and add SPCGM and CPSGM indexes.

Right click on RDF_LNK_PSCGM_IDX and select Drop Semantic Index.

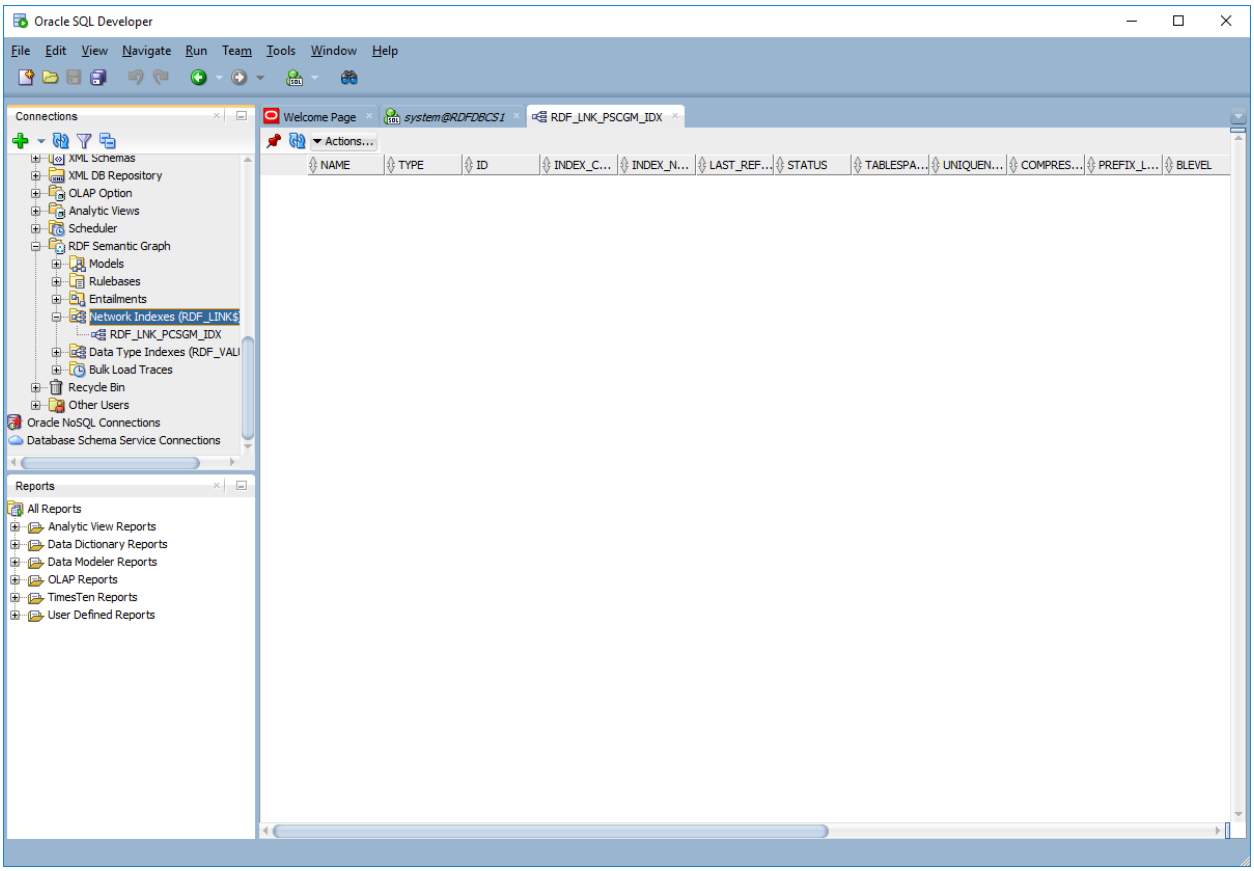


Click Apply.

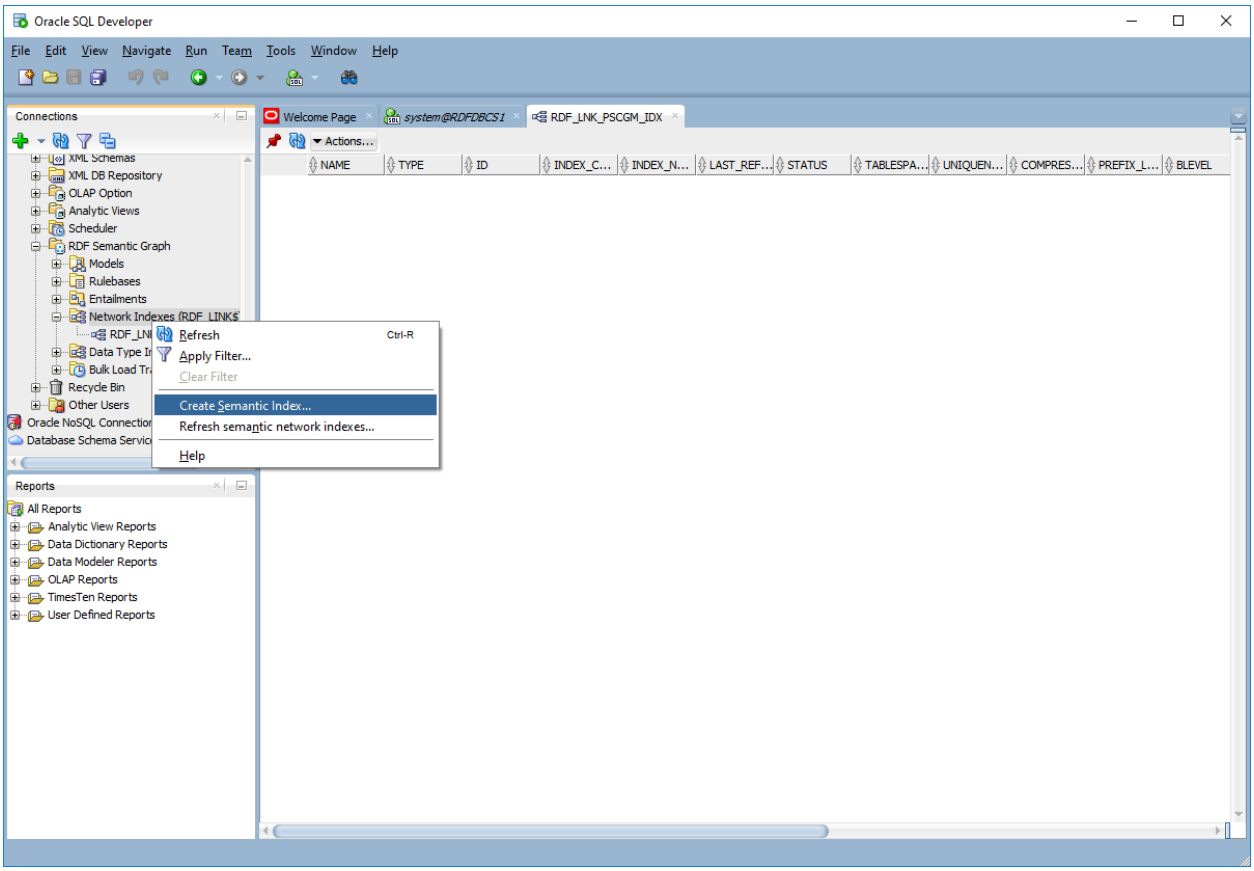




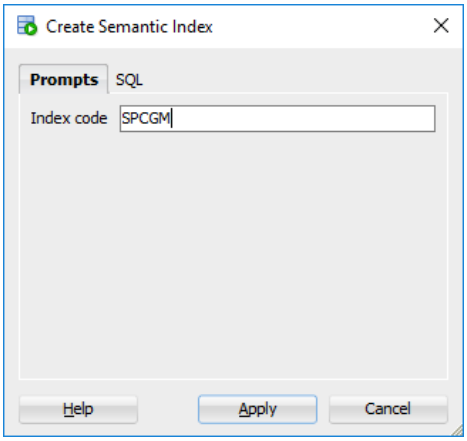
Only the PCSGM index should appear under Network Indexes now.

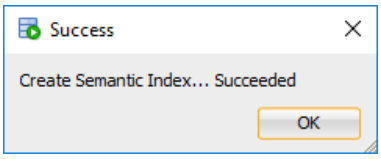


Right click on Network Indexes and select Create Semantic Index.

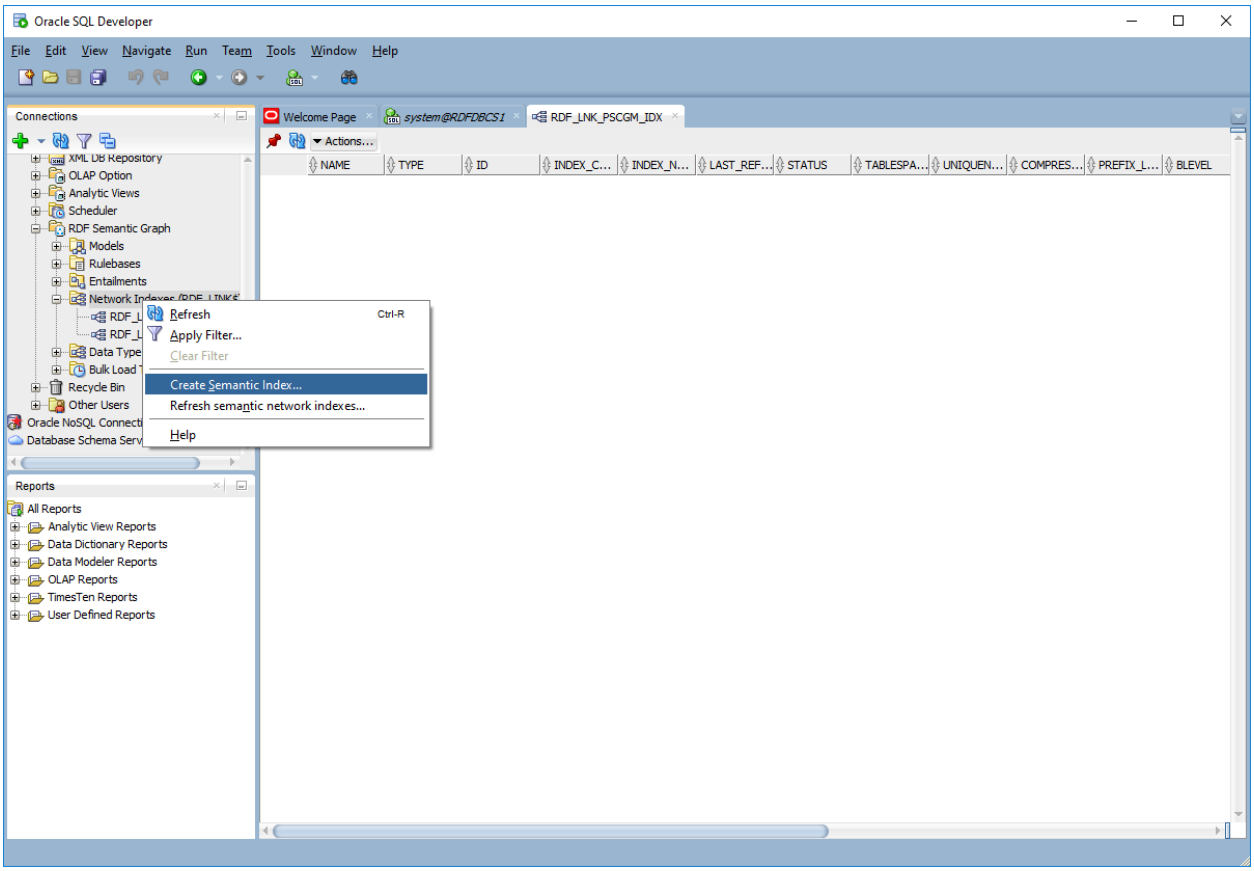


Enter SPCGM as the Index code and click Apply.

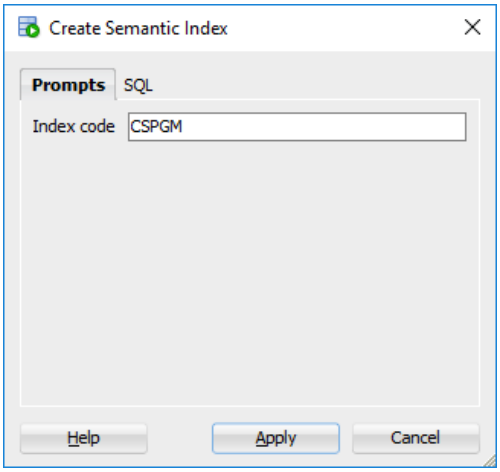


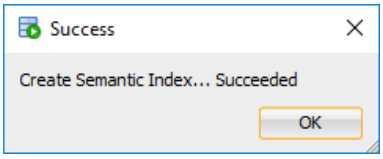


Right click on Network Indexes and select Create Semantic Index.

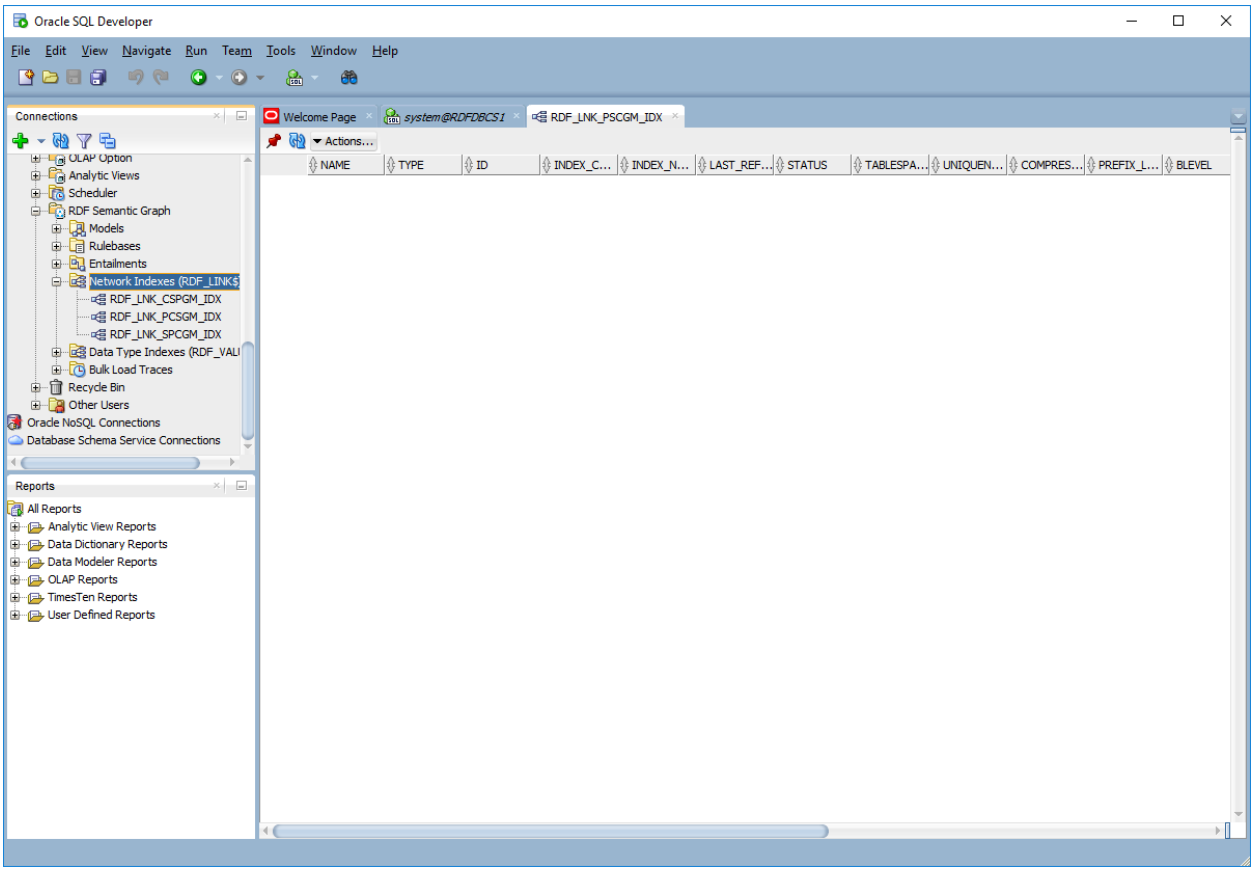


Enter CSPGM as the index code and click Apply.



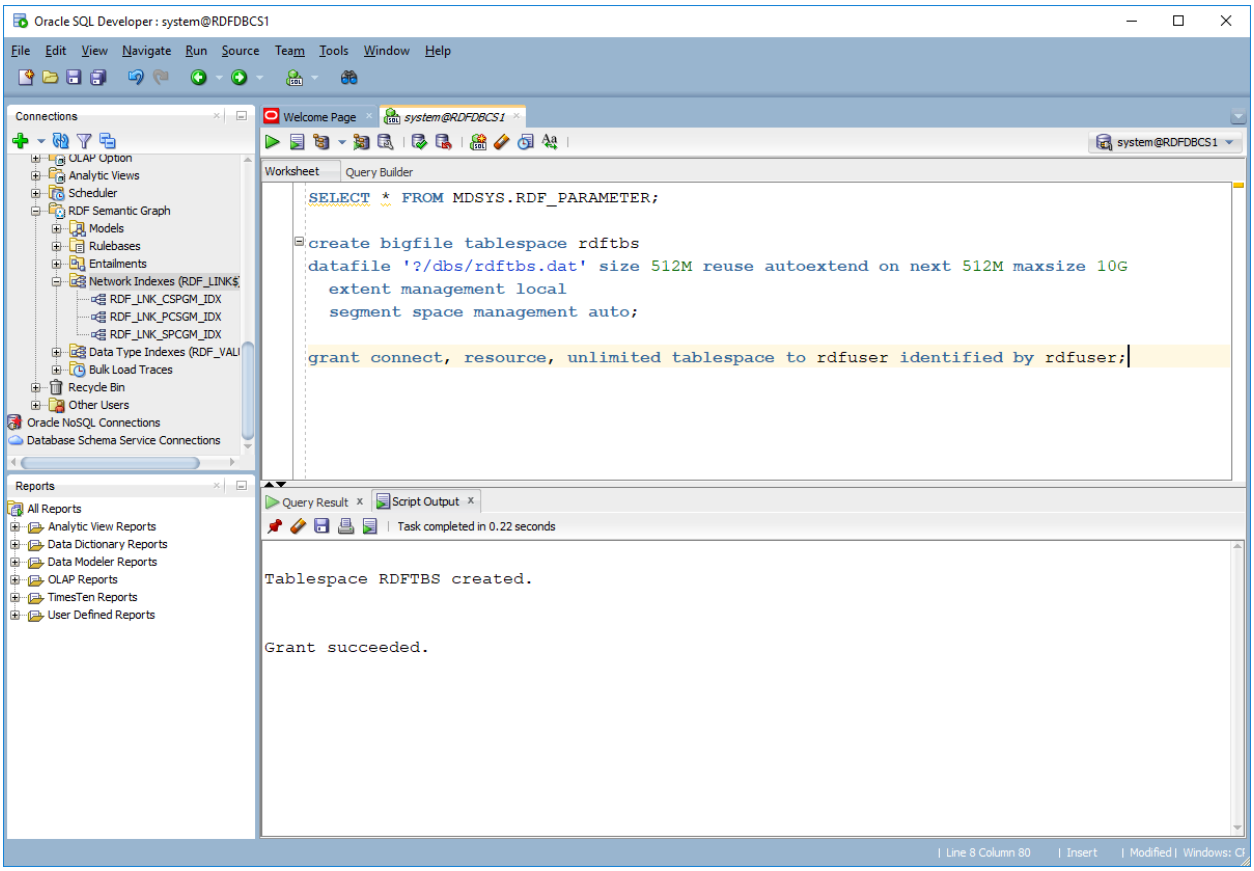


You should now see CSPGM, PCSGM and SPCGM indexes under Network Indexes.

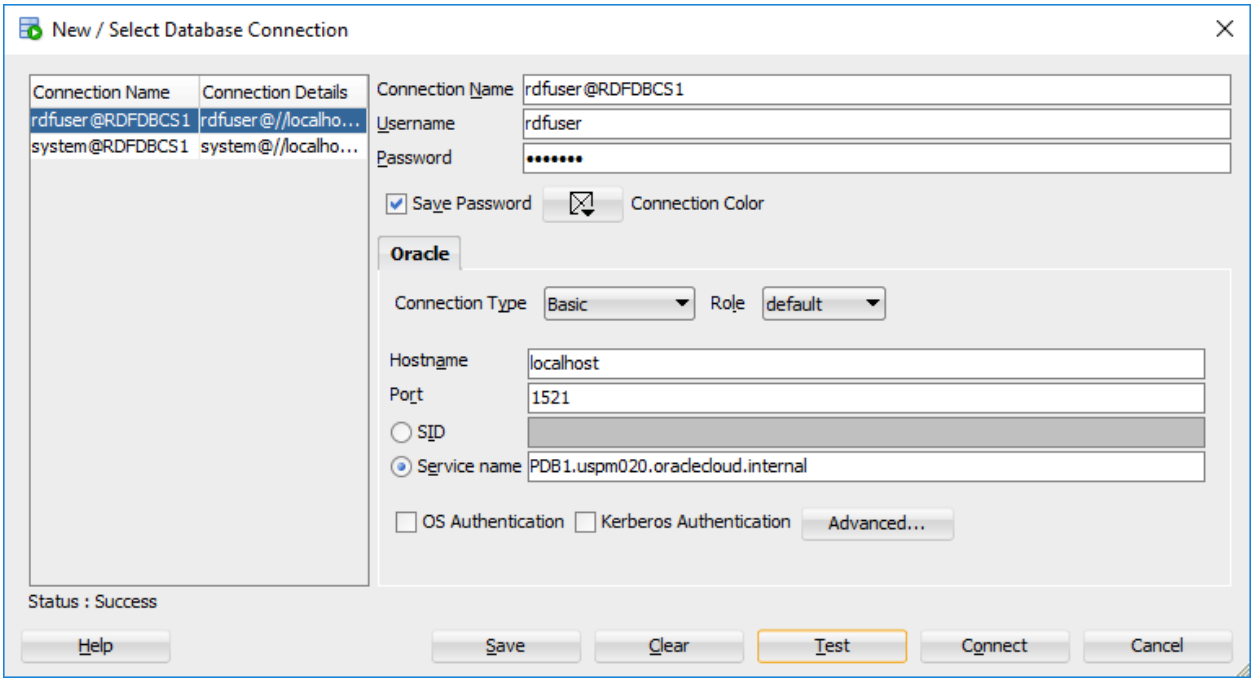


The next step is to create a regular non-administrator user to perform operations with the Semantic Network. We can create user RDFUSER by clicking on the SQL Worksheet tab for SYSTEM and running the following SQL statement as SYSTEM:

```
grant connect, resource, unlimited tablespace to rdfuser identified by
rdfuser;
```

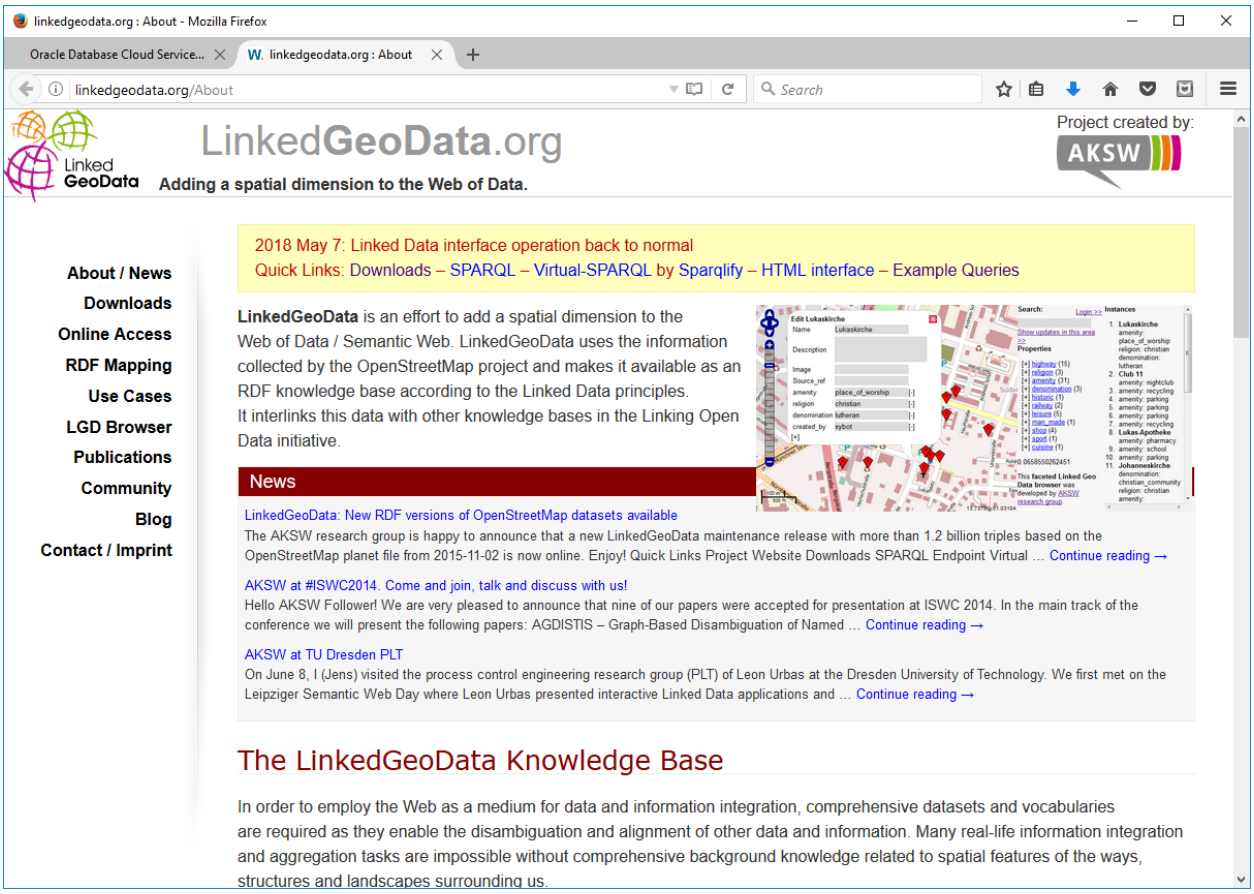


Now create a new connection in SQL Developer for RDFUSER. Use rdfuser for the Username and Password, and use the same Hostname, Port and Service name as we did before for the SYSTEM connection.



Now we will use the RDFUSER account to perform a few operations. We will download a publicly available RDF dataset and transfer it to our DBCS instance. Then we will use SQL Developer to bulk load this RDF dataset and execute some SPARQL queries against it.

The data we will use comes from the Linked Geo Data project: http://linkedgeodata.org/About

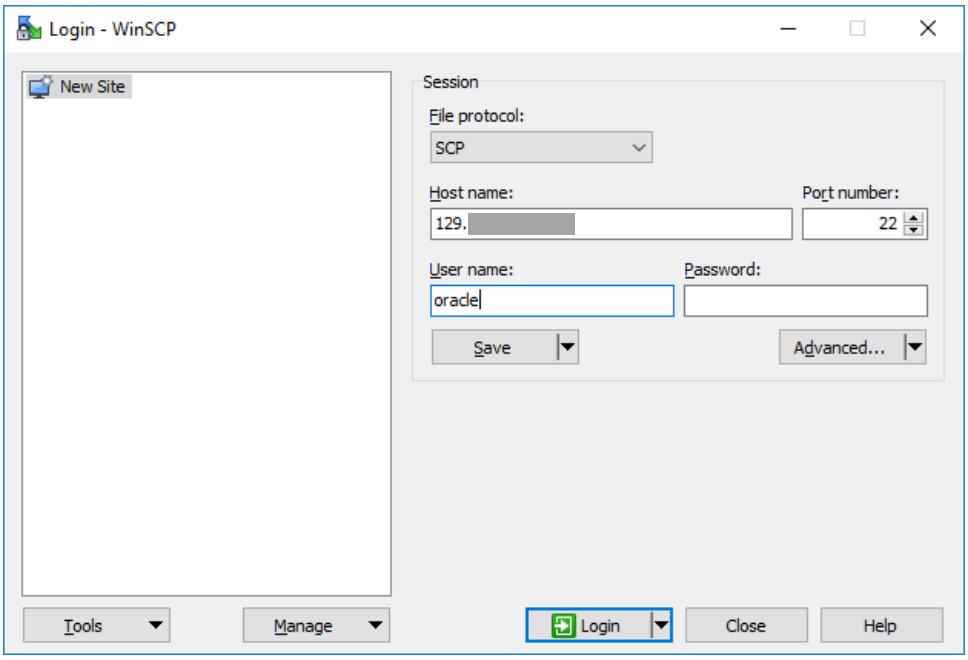


The project provides several downloadable datasets, which are indexed here: https://hobbitdata.informatik.uni-leipzig.de/LinkedGeoData/downloads.linkedgeodata.org/releases/

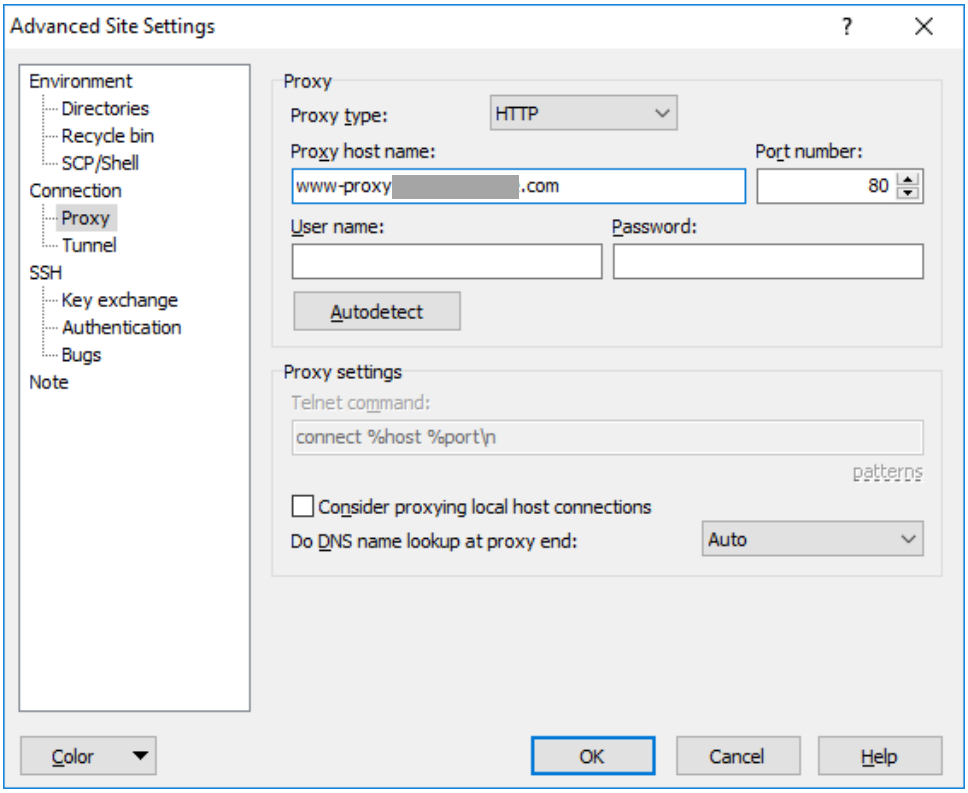
We will be using a 1.2 million triple dataset of sports facilities: https://hobbitdata.informatik.uni-leipzig.de/LinkedGeoData/downloads.linkedgeodata.org/releases/2015-11-02/2015-11-02-SportThing.node.sorted.nt.bz2

After downloading this file, you can use WinSCP to copy it to the DBCS instance.

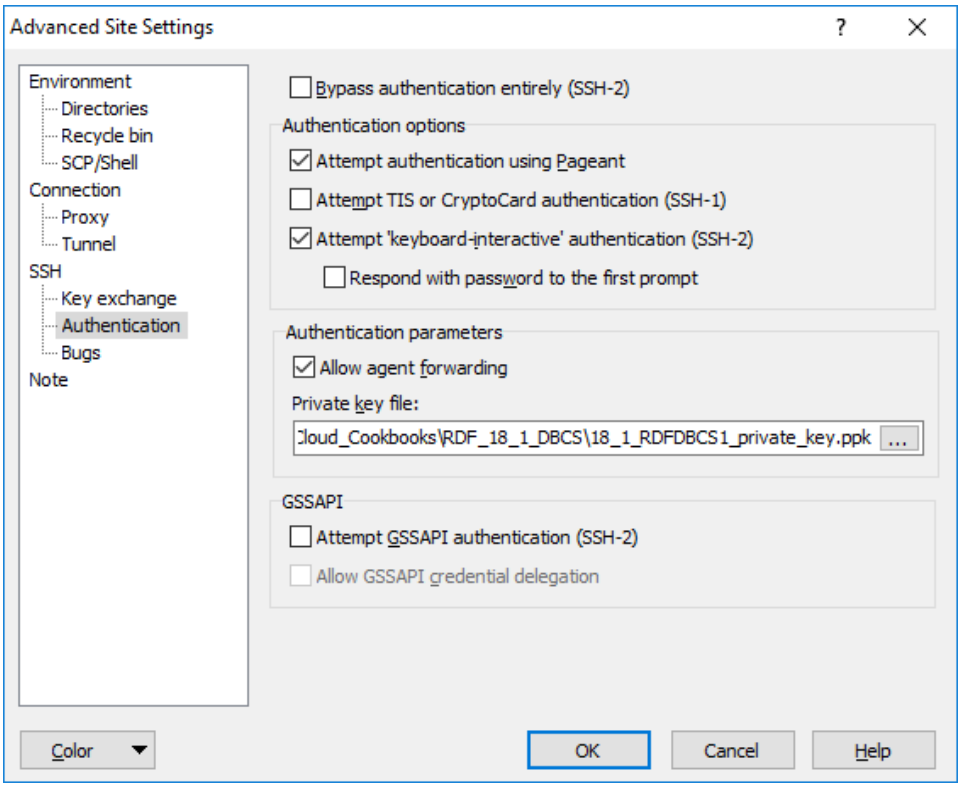
Open WinSCP. Choose SCP File protocol and enter your DBCS public IP for Host name. Enter oracle for User name and leave Password blank.



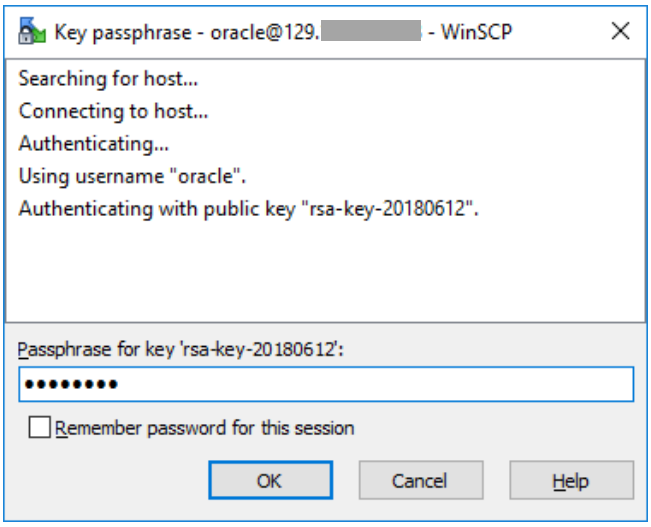
Click Advanced to configure SSH authentication and any proxy settings. Click Proxy and enter any proxy information for your network.



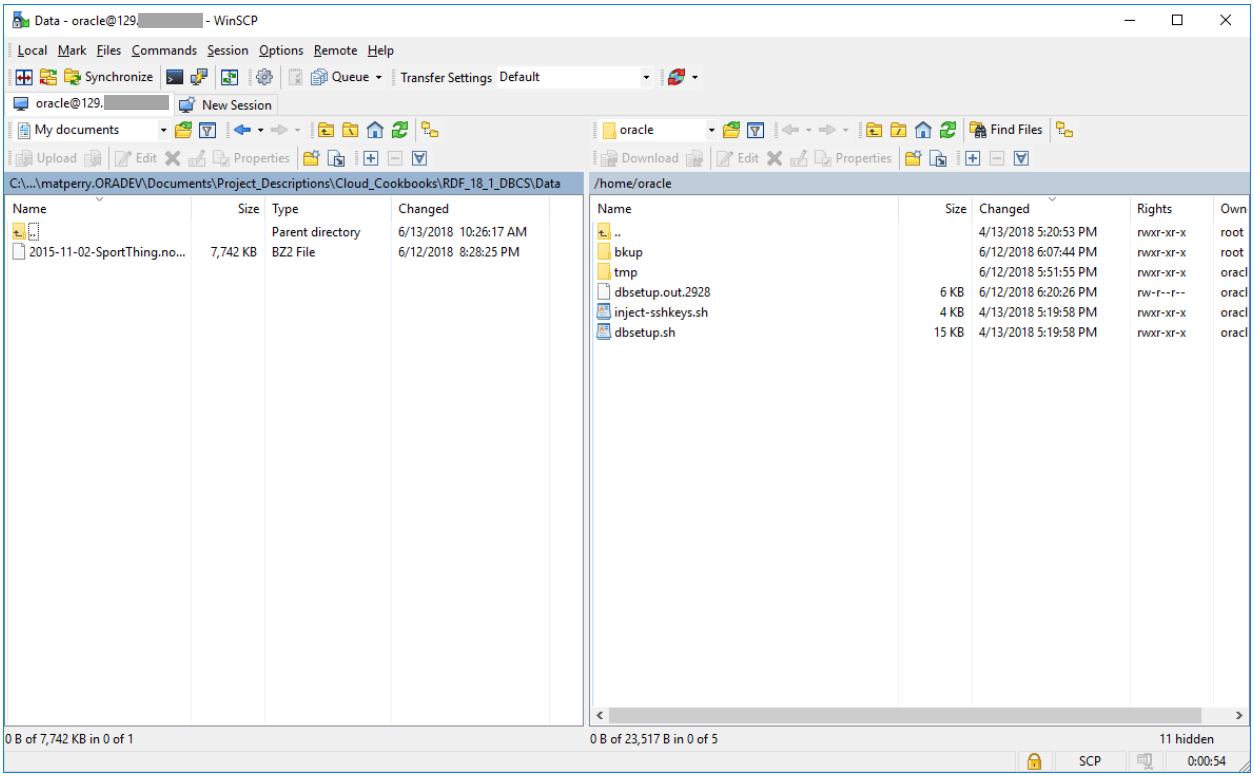
Click Authentication and add your .ppk private key file. Also, uncheck “Respond with password to the first prompt”.



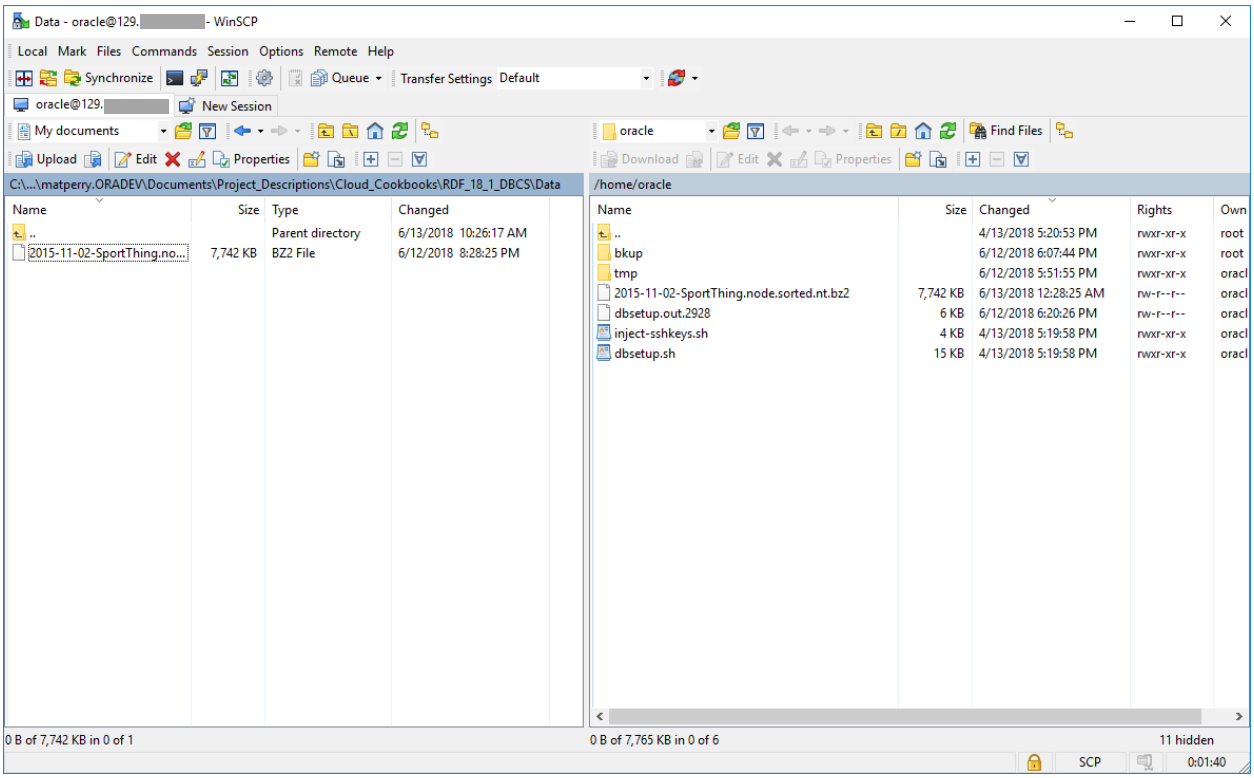
Click OK and then Login. Enter passphrase if necessary.



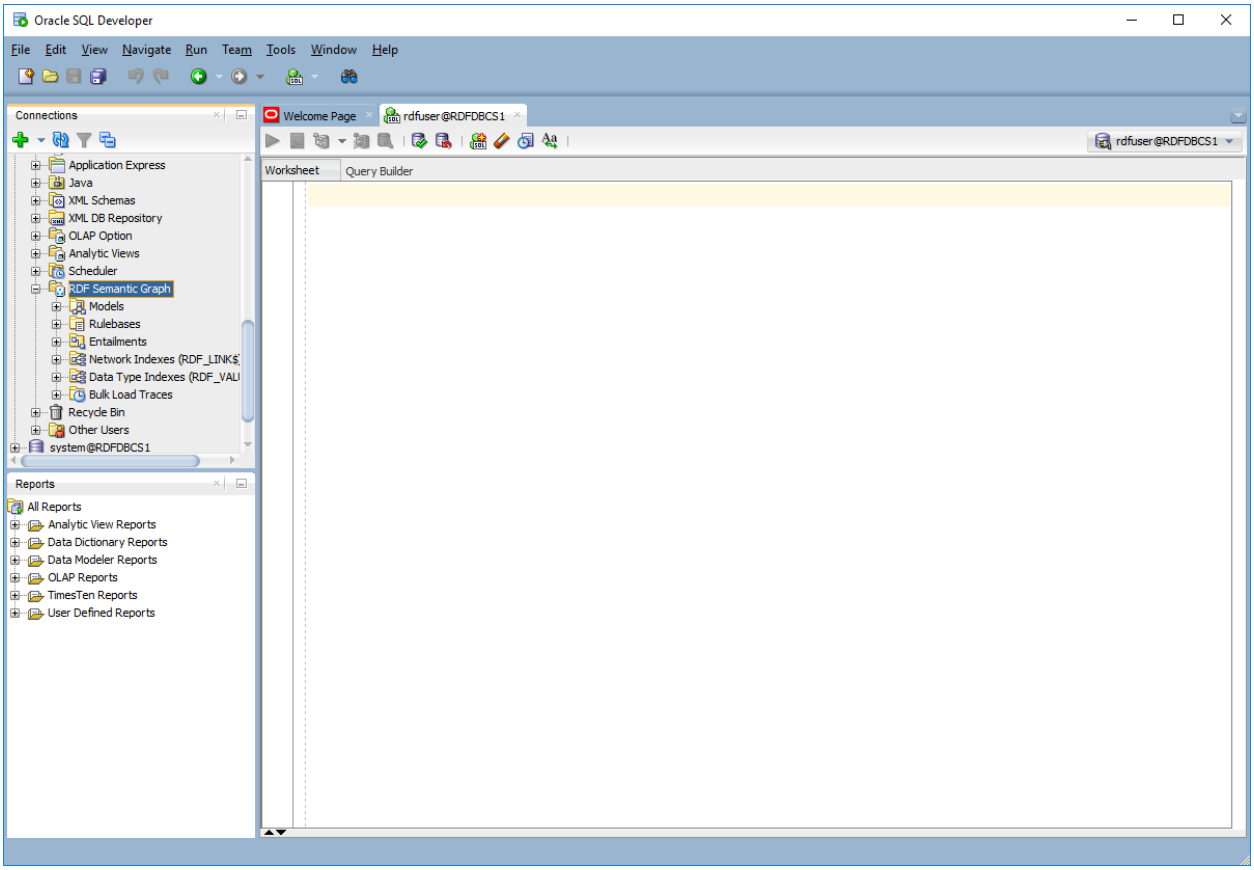
Browse to appropriate directories on the local and remote machines.



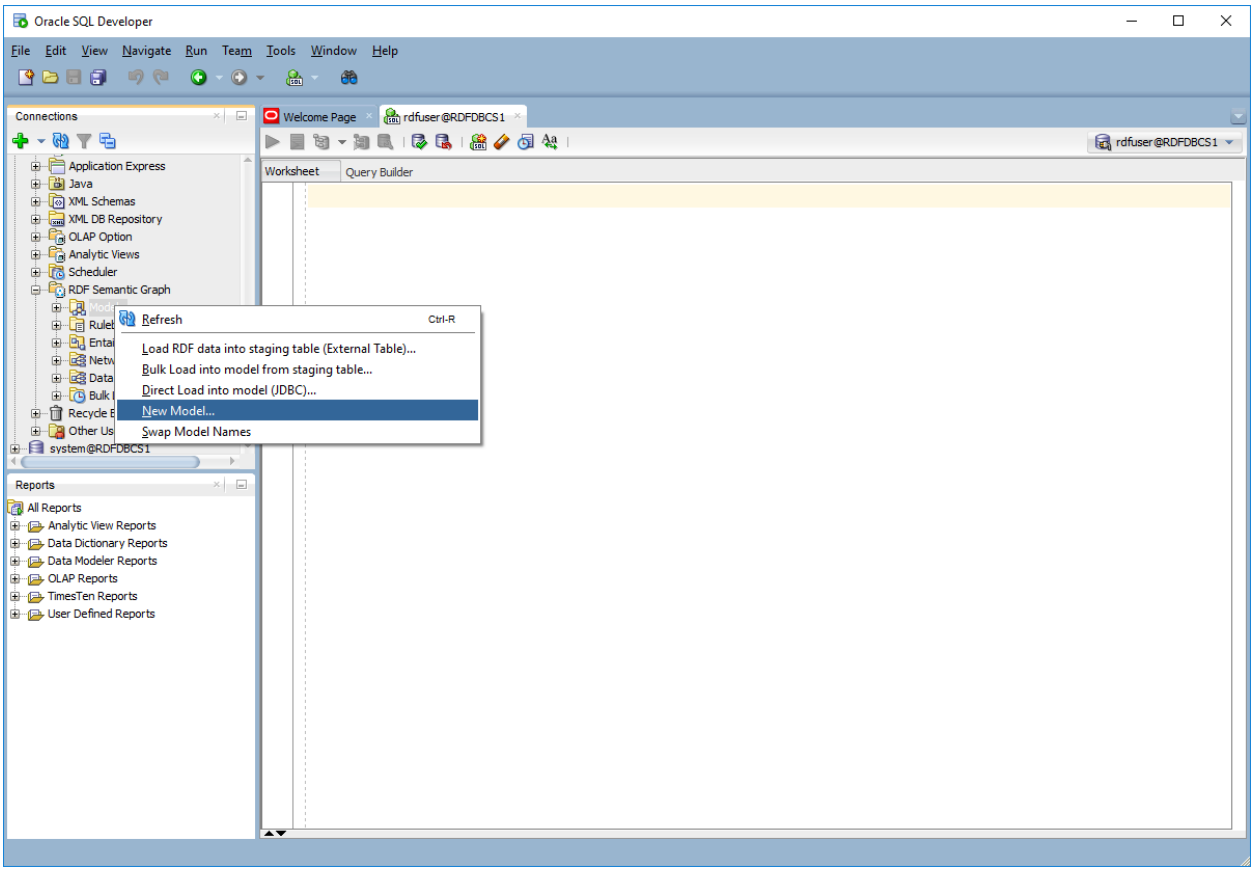
Drag the downloaded file from Linked Geo Data from left to right to do the transfer. In this case, we are storing the data in /home/oracle on the remote DBCS machine.



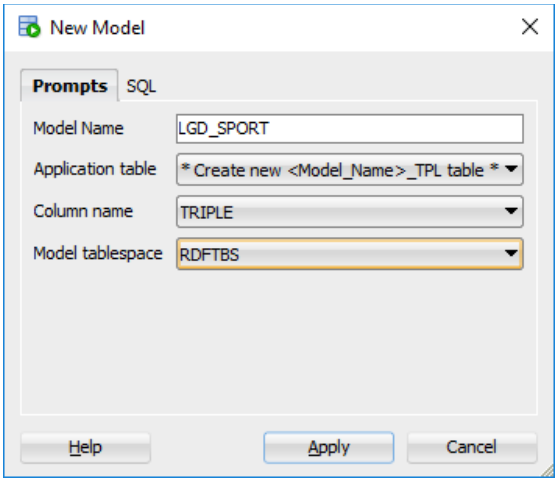
Next, we will create a semantic model to hold our RDF dataset. Open the RDFUSER connection in SQL Developer and expand the RDF Semantic Graph component.

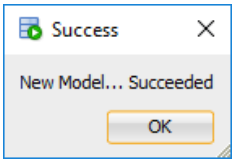


Right click on Models and choose New Model.

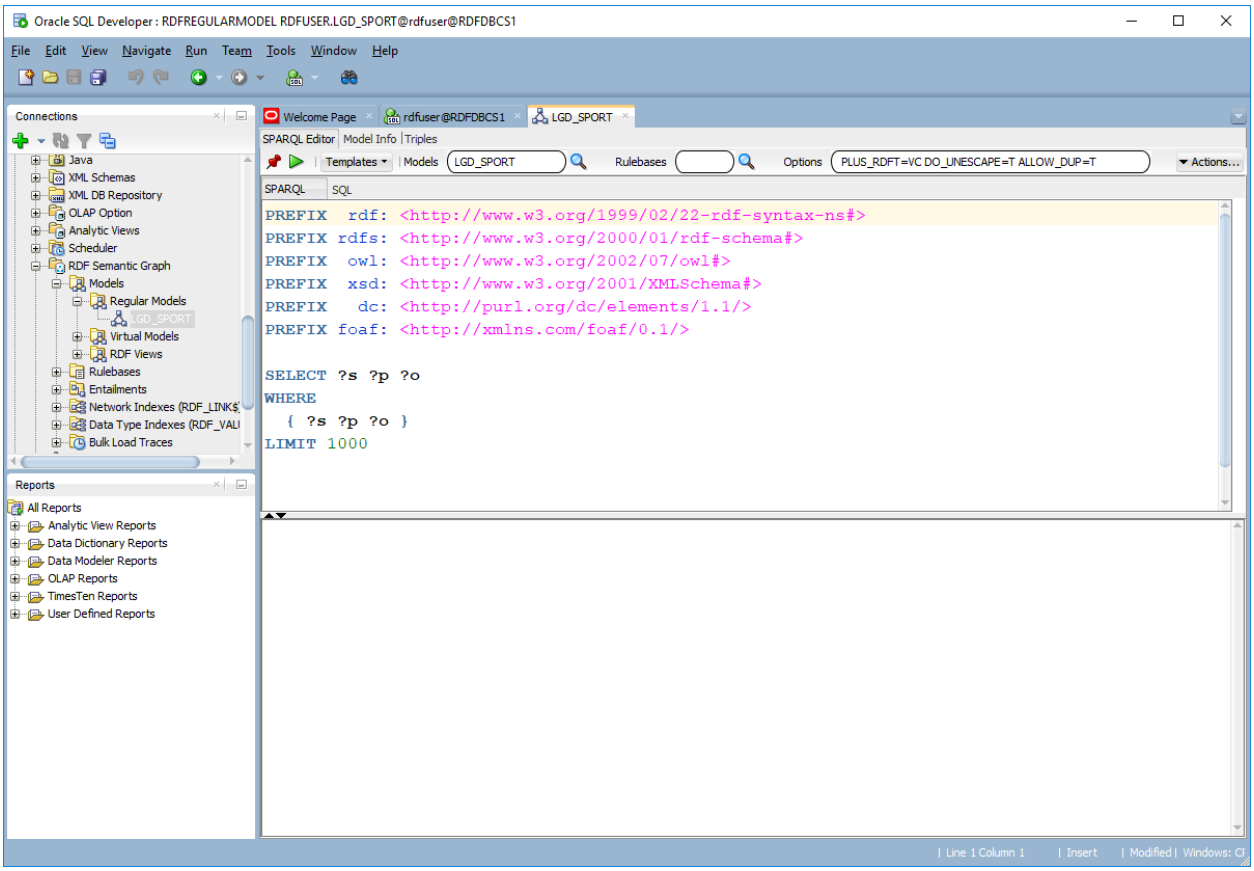


Enter a Model Name and choose to create a new Application table with TRIPLE column. Choose the tablespace you used when creating the semantic network for Model tablespace. Click Apply.

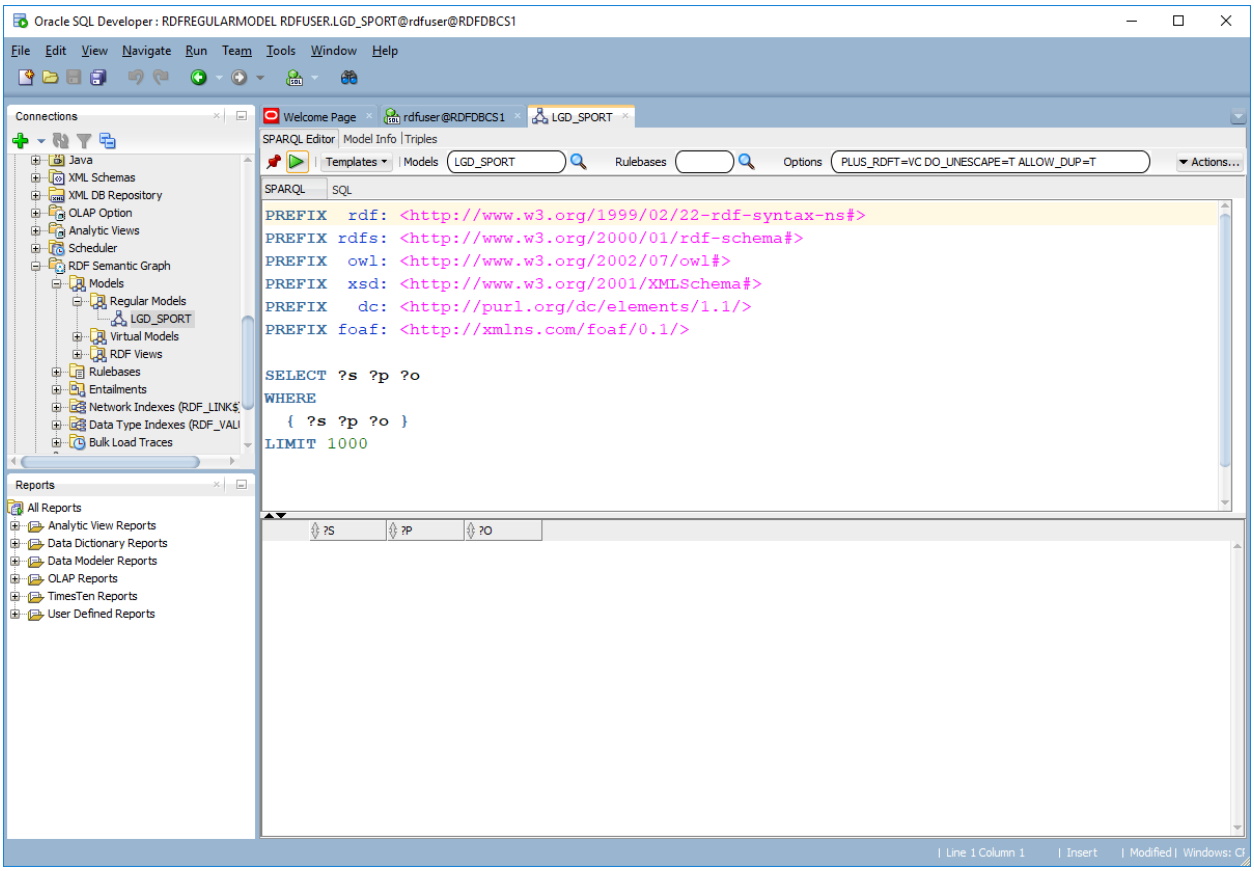




Expand Models and Regular Models and click the LGD_SPORT model we created. This will bring up the SPARQL query editor for LGD_SPORT.



We can click the green triangle to run the default query and see that the model is empty.

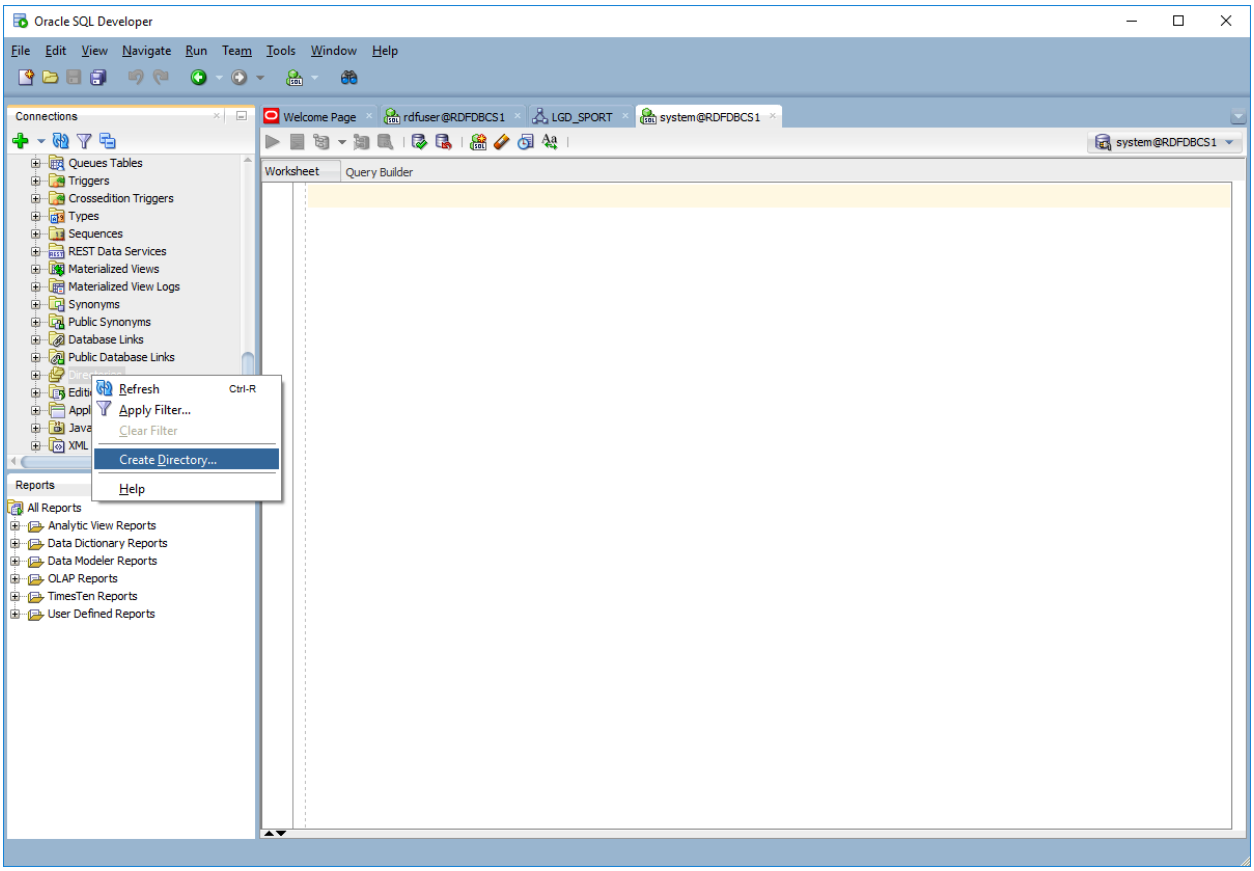


Now, we will bulk load the downloaded RDF file. The bulk load process consists of two major steps:

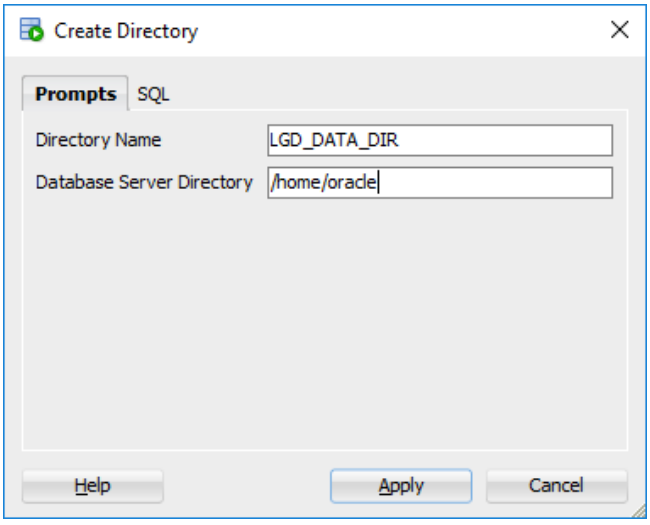
1. Loading the file from the file system into a simple staging table in the database.
2. Loading data from the staging table into our semantic model.

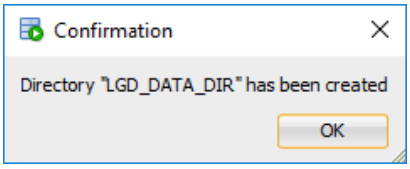
The first step involves loading from an external table, so we need to use the SYSTEM connection to create a DIRECTORY in the database and grant privileges on it to RDFUSER.

Expand the SYSTEM connection and right-click on Directories. Then choose Create Directory.

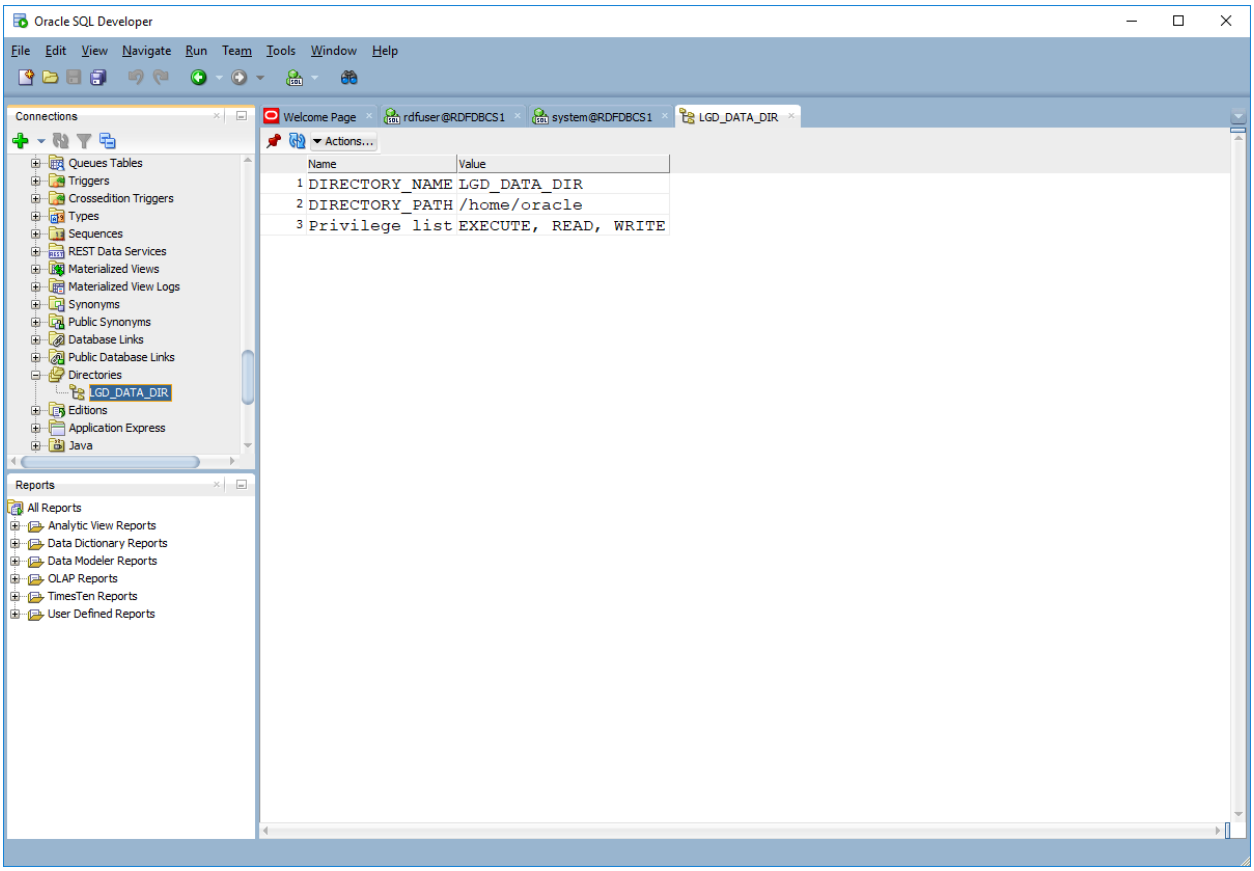


Enter a Directory Name and the full path for the directory on the database server. Click Apply.

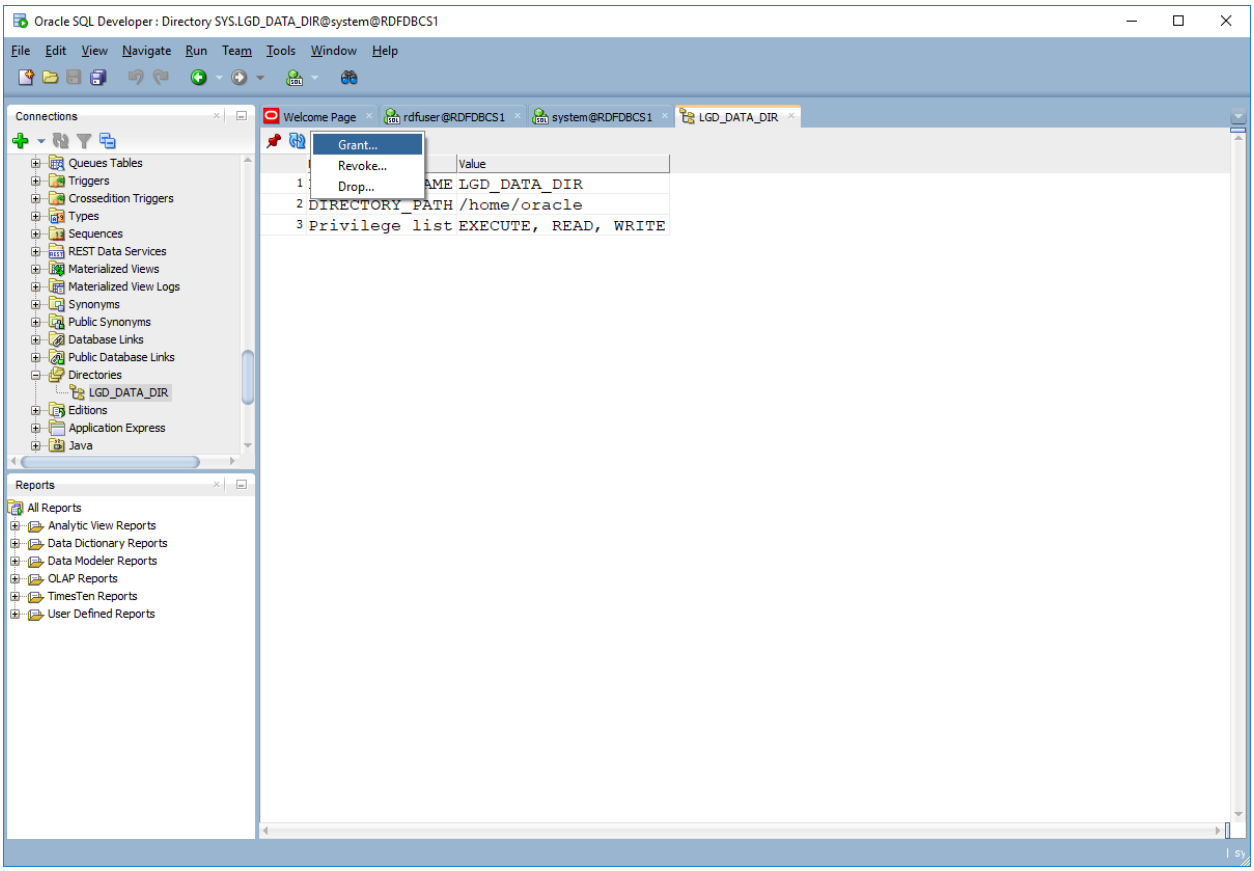




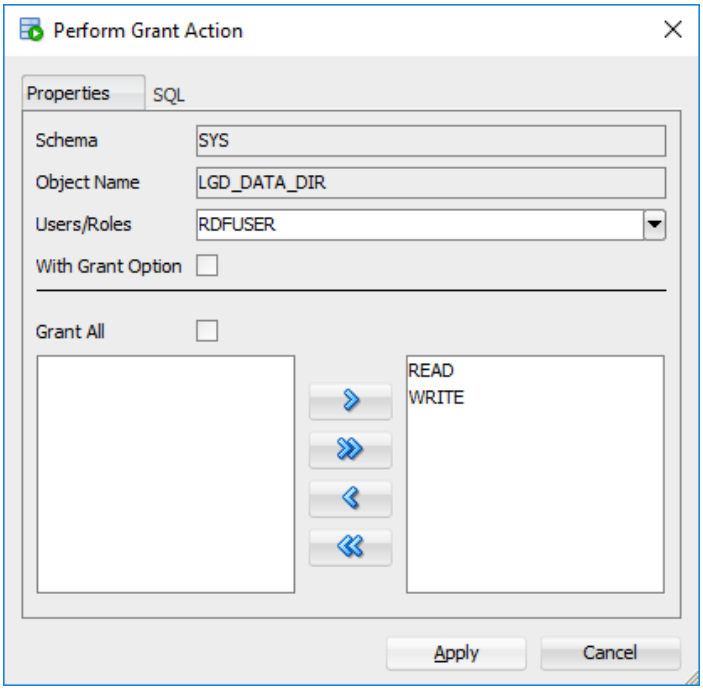
Expand Directories and click the directory name to see details.

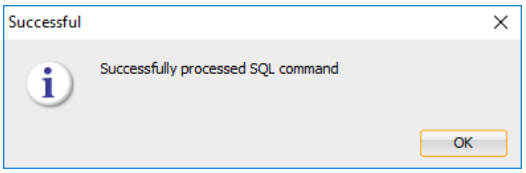


Now we need to grant privileges on this directory to RDFUSER. Click actions and select Grant.



Grant READ and WRITE privileges to RDFUSER. Click Apply.

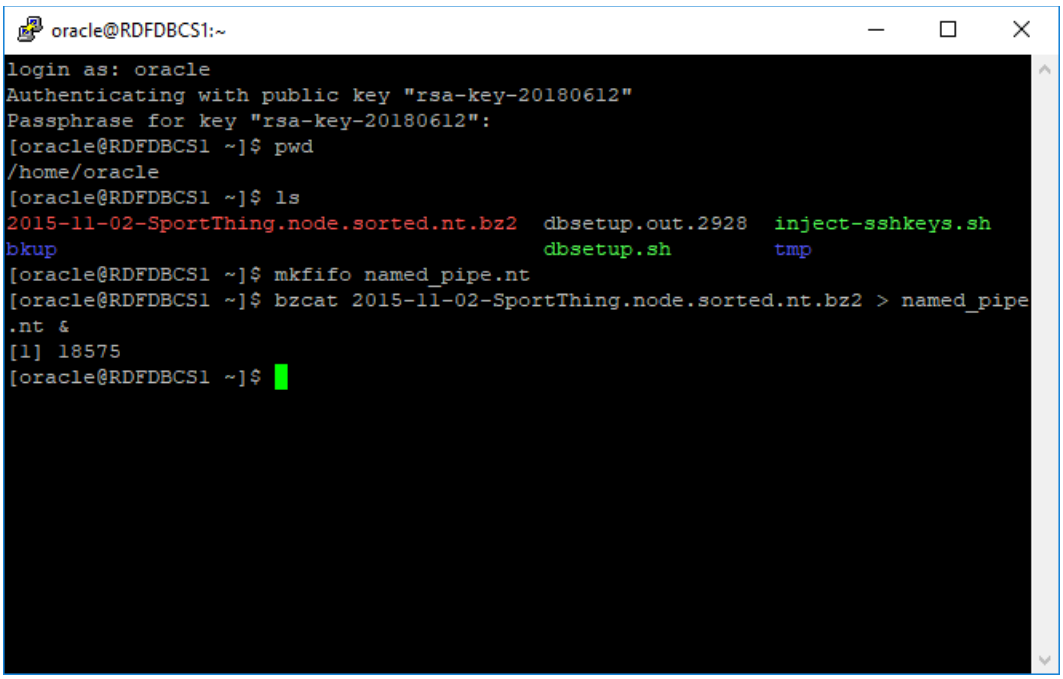




The downloaded RDF file is compressed, so we will use a Unix named pipe to stream the uncompressed data rather than storing an uncompressed version of the file.

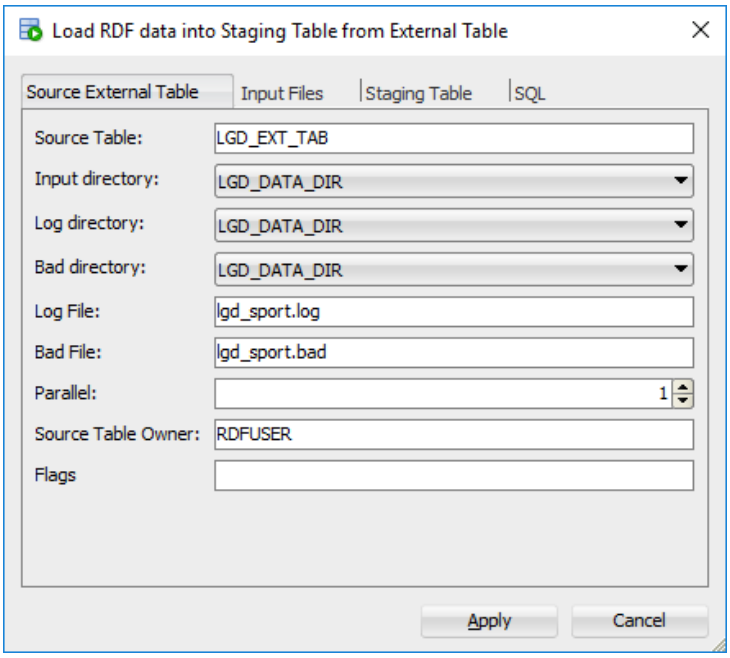
Use PuTTY to make an SSH connection to the remote DBCS instance as the oracle user. Then execute the following commands to create a named pipe to stream the uncompressed data.

```
mkfifo named_pipe.nt
bzcat 2015-11-02-SportThing.node.sorted.nt.bz2 > named_pipe.nt &
```

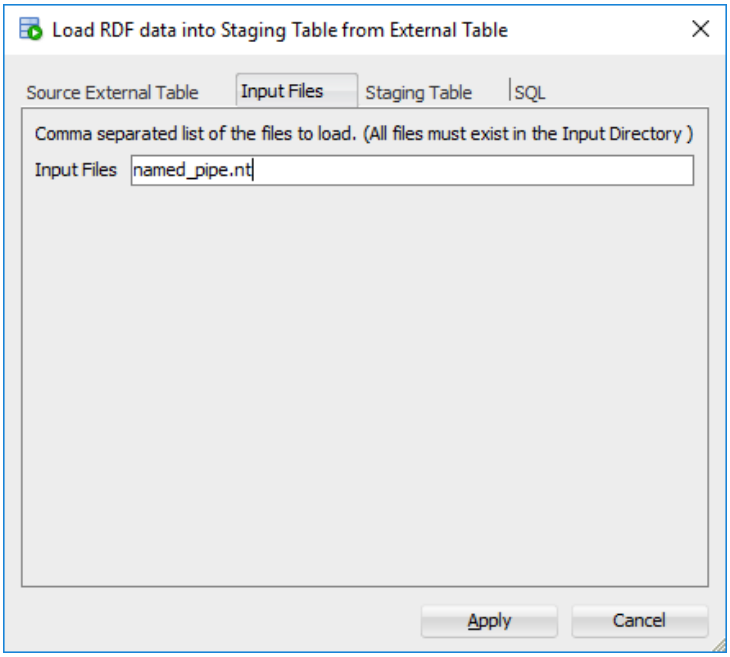


Now, expand the RDFUSER connection in SQL Developer and expand the RDF Semantic Graph component. Then right click on Models and select “Load RDF data into staging table (External Table).”

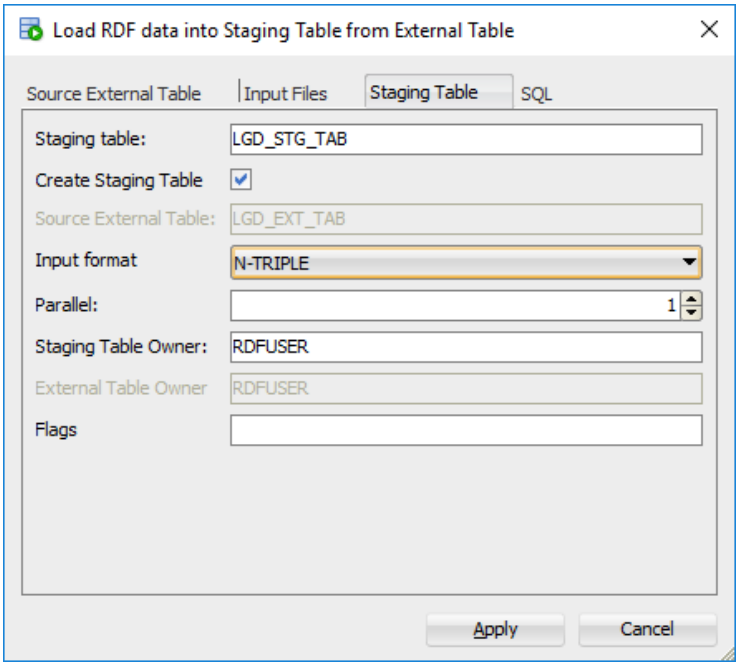
Choose a name for the external table to create (we are using LGD_EXT_TAB) and fill in the other fields on the Source External Table tab.



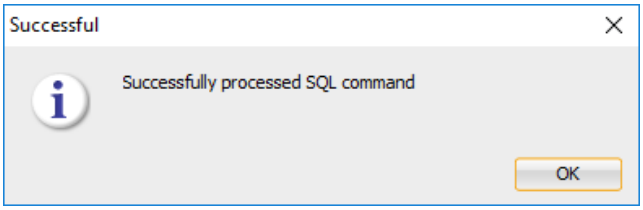
Enter the names of the files to load (we are using named_pipe.nt) on the Input Files tab.



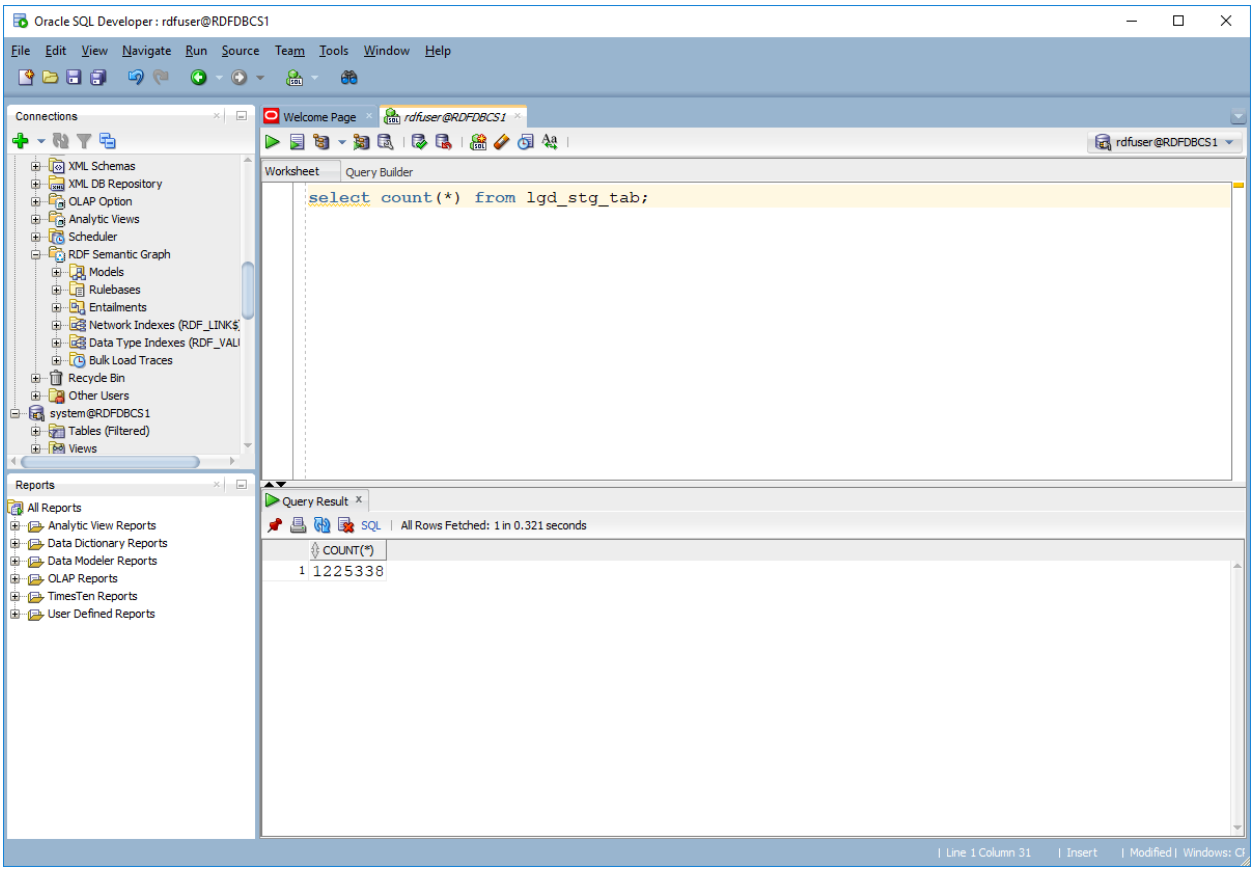
Finally, use the Staging Table tab to enter a name for the staging table that will be created (we are using LGD_STG_TAB) and choose the appropriate format.

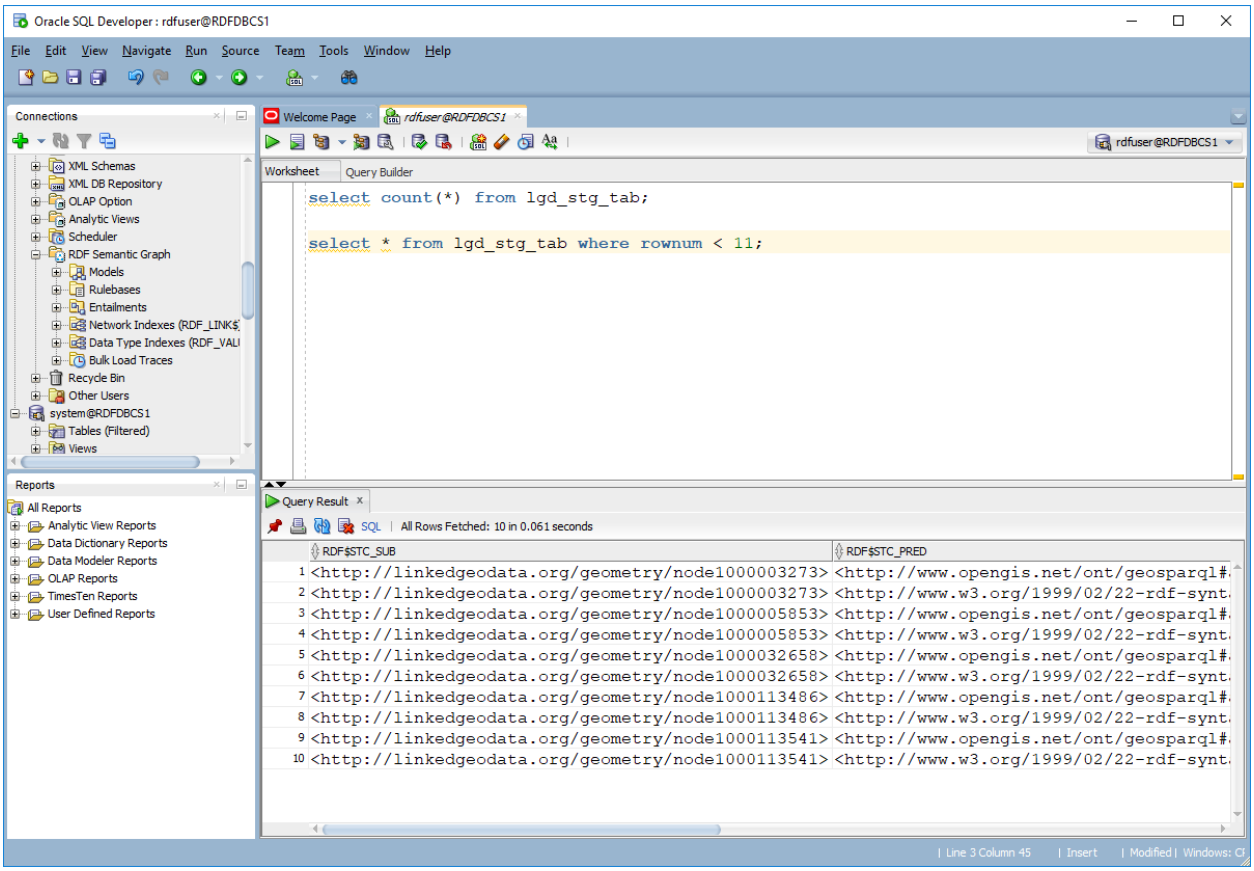


Now click Apply to load the data into LGD_STG_TAB.

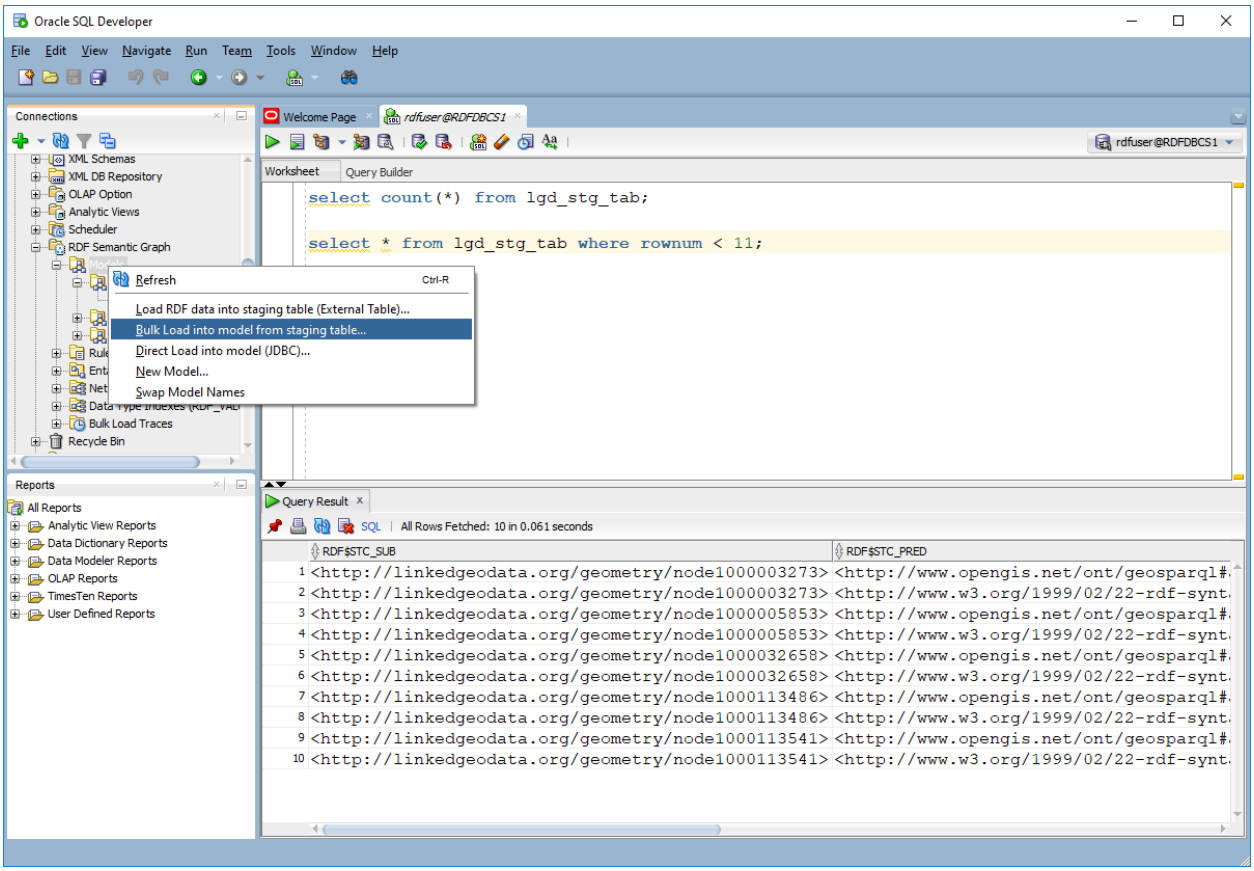


Check the contents of LGD_STG_TAB.

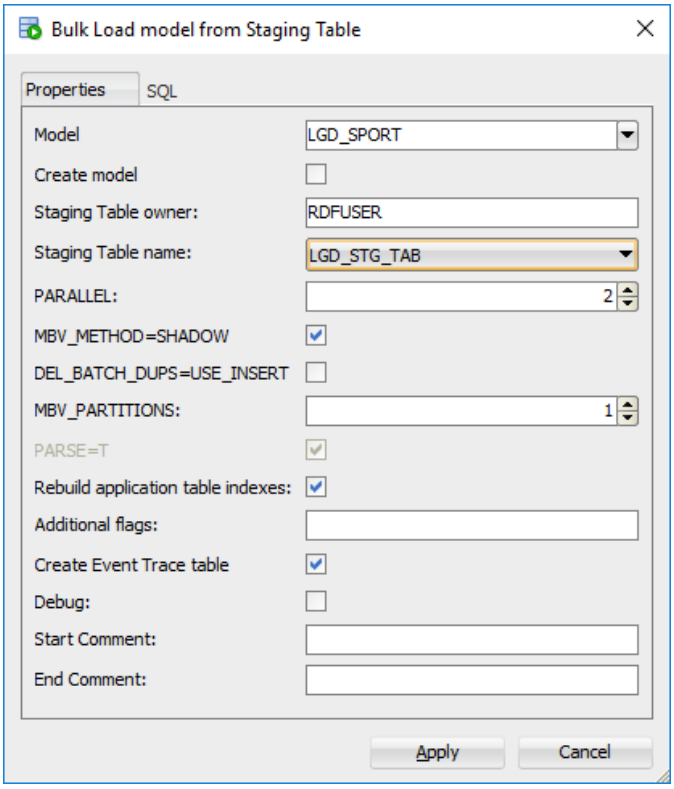

Oracle SQL Developer : rdfuser@RDFDBCS1
File Edit View Navigate Run Source Team Tools Window Help
Connections
XML Schemas
XML DB Repository
OLAP Option
Analytic Views
Scheduler
RDF Semantic Graph
Models
Rulebases
Entailments
Network Indexes (RDF_LINK$)
Data Type Indexes (RDF_VALI
Bulk Load Traces
Recycle Bin
Other Users
system@RDFDBCS1
Tables (Filtered)
Views
Reports
All Reports
Analytic View Reports
Data Dictionary Reports
Data Modeler Reports
OLAP Reports
TimesTen Reports
User Defined Reports
Welcome Page
rdfuser@RDFDBCS1
rdfuser@RDFDBCS1
Worksheet
Query Builder
select count(*) from lgd_stg_tab;
Query Result
SQL | All Rows Fetched: 1 in 0.321 seconds
COUNT(*)
1 1225338
Line 1 Column 31 | Insert | Modified | Windows:

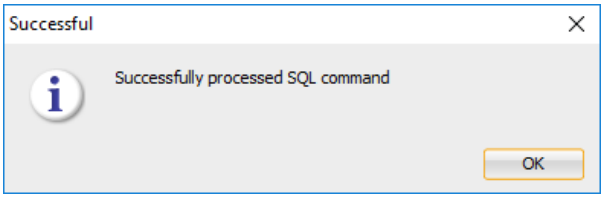


Next, we will load the data from LGD_STG_TAB into the LGD_SPORT semantic model. To bring up the bulk load interface, expand RDF Semantic Graph under the RDFUSER connection. Then, right-click on Models and select “Bulk Load into model from staging table”.

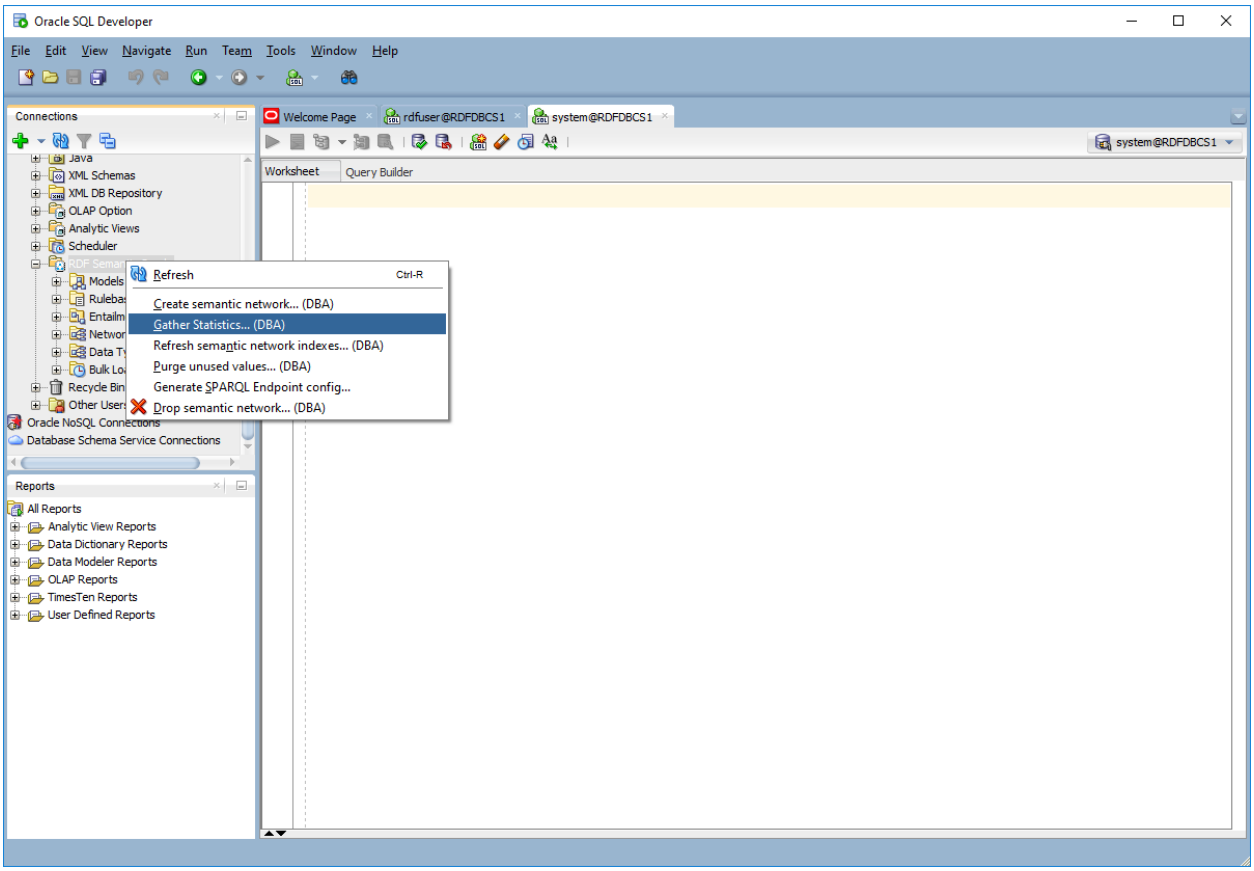


Enter LGD_SPORT for Model and unselect the Create model option since we have already created this semantic model. Also, choose the LGD_STG_TAB for Staging table name. Be careful not to select the external table (LGD_EXT_TAB), as it will also be listed. Consult the user guide for more information on the other options for bulk load. Click Apply, and the load will finish in a minute or so.

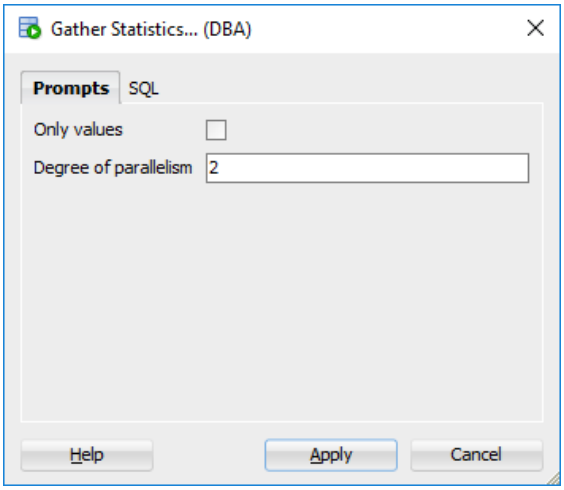


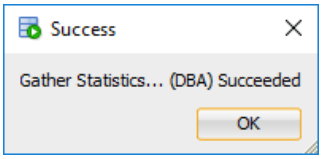


Now that we have completed the bulk load, it is a good idea to gather statistics on the whole RDF network. Only a privileged user can gather statistics on the whole RDF network. Expand the RDF Semantic Graph component under the SYSTEM connection, right click on RDF Semantic Graph and select Gather Statistics.

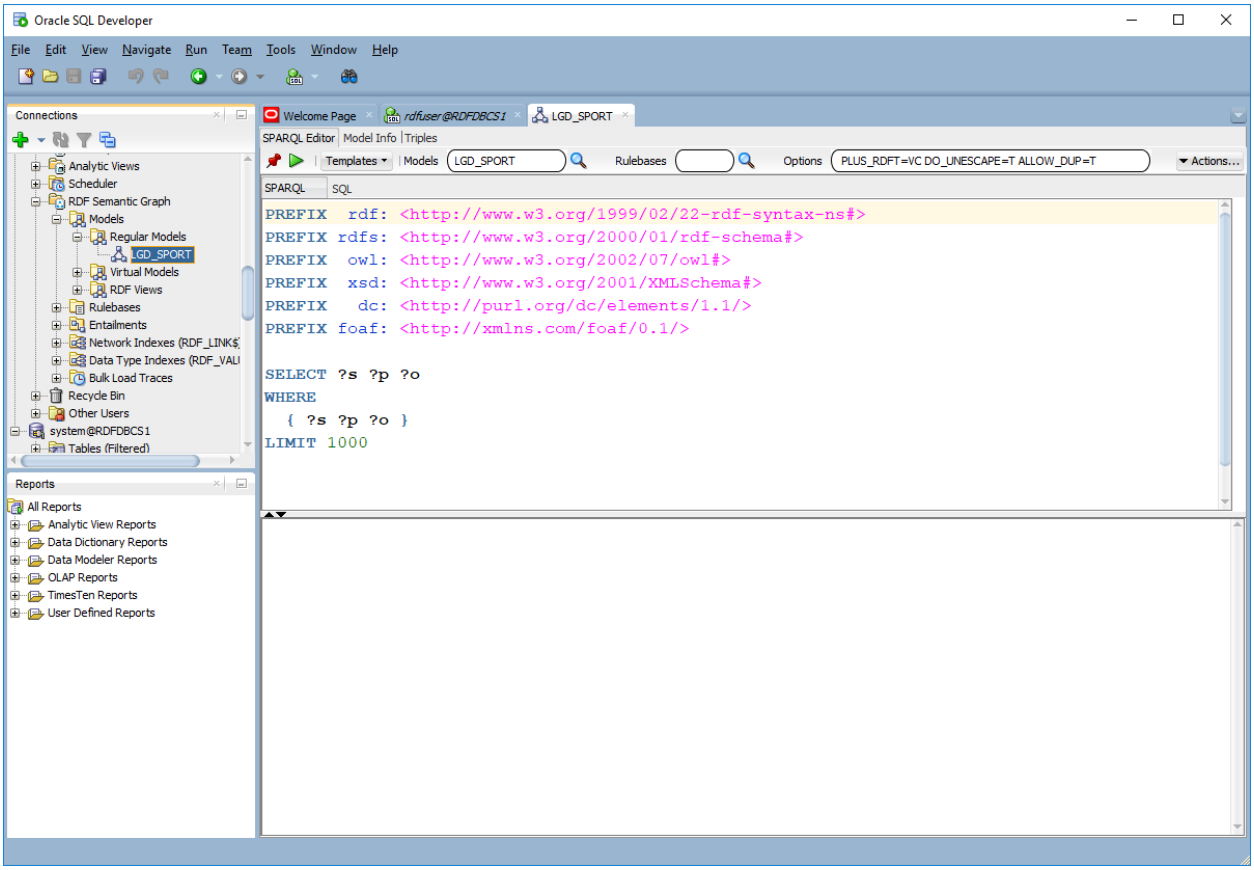


Enter the desired Degree of parallelism and click Apply.



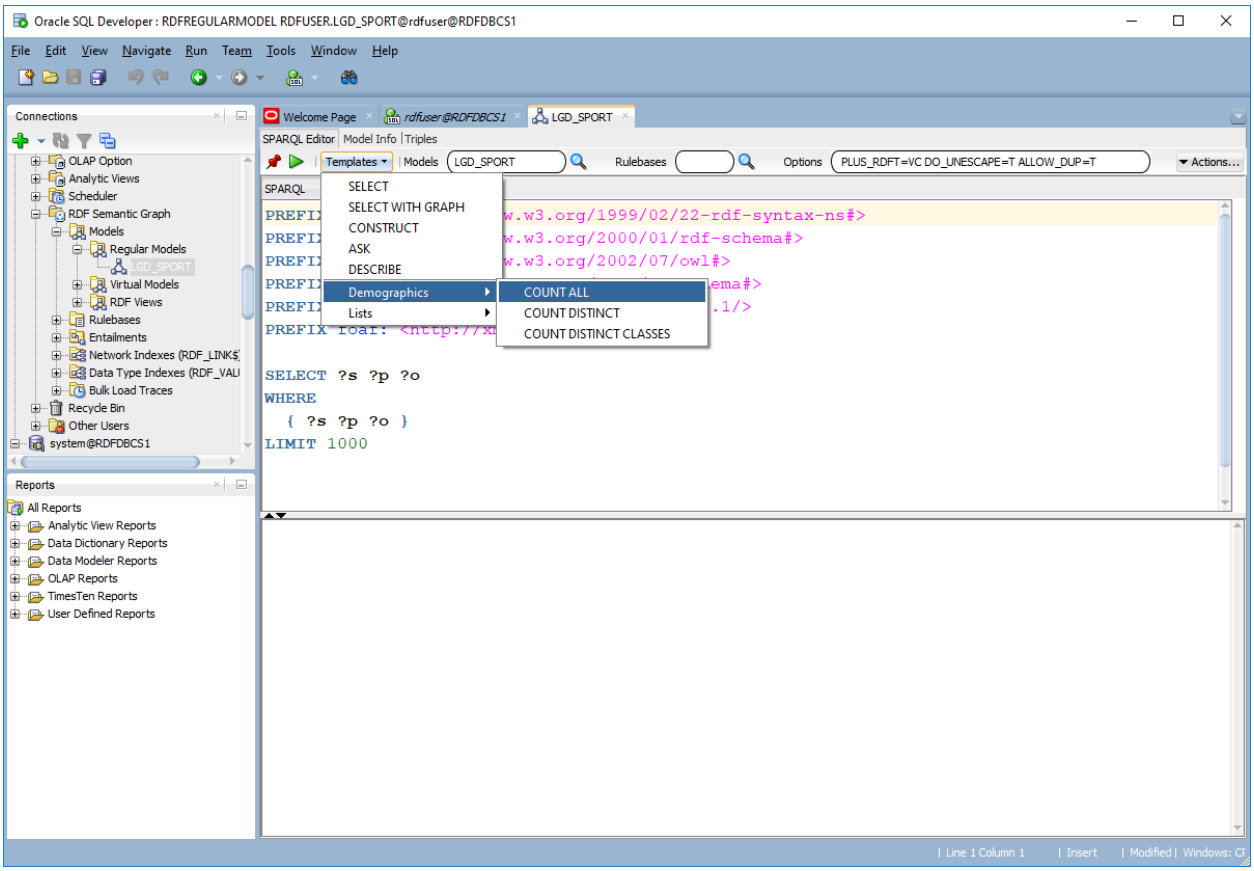


After gathering statistics, the data is ready to query. Go back to the RDFUSER connection and expand Models and Regular Models under RDF Semantic Graph. Clicking LGD_SPORT will open the SPARQL query editor for this semantic model.

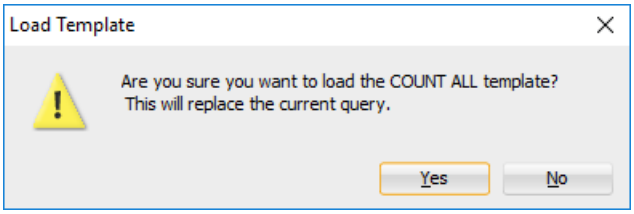


You can edit and execute SPARQL queries here. In addition, several pre-created templates are available.

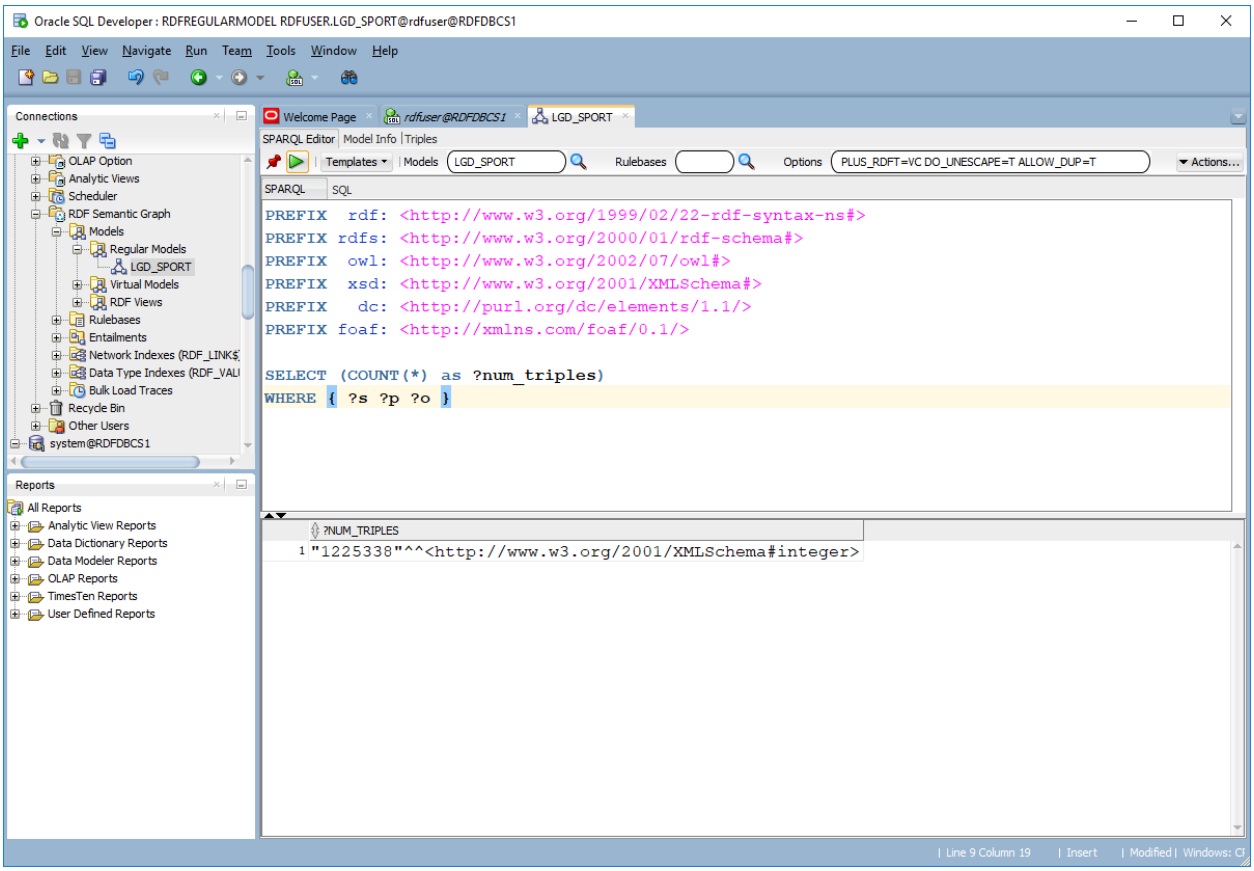
Count All triples.



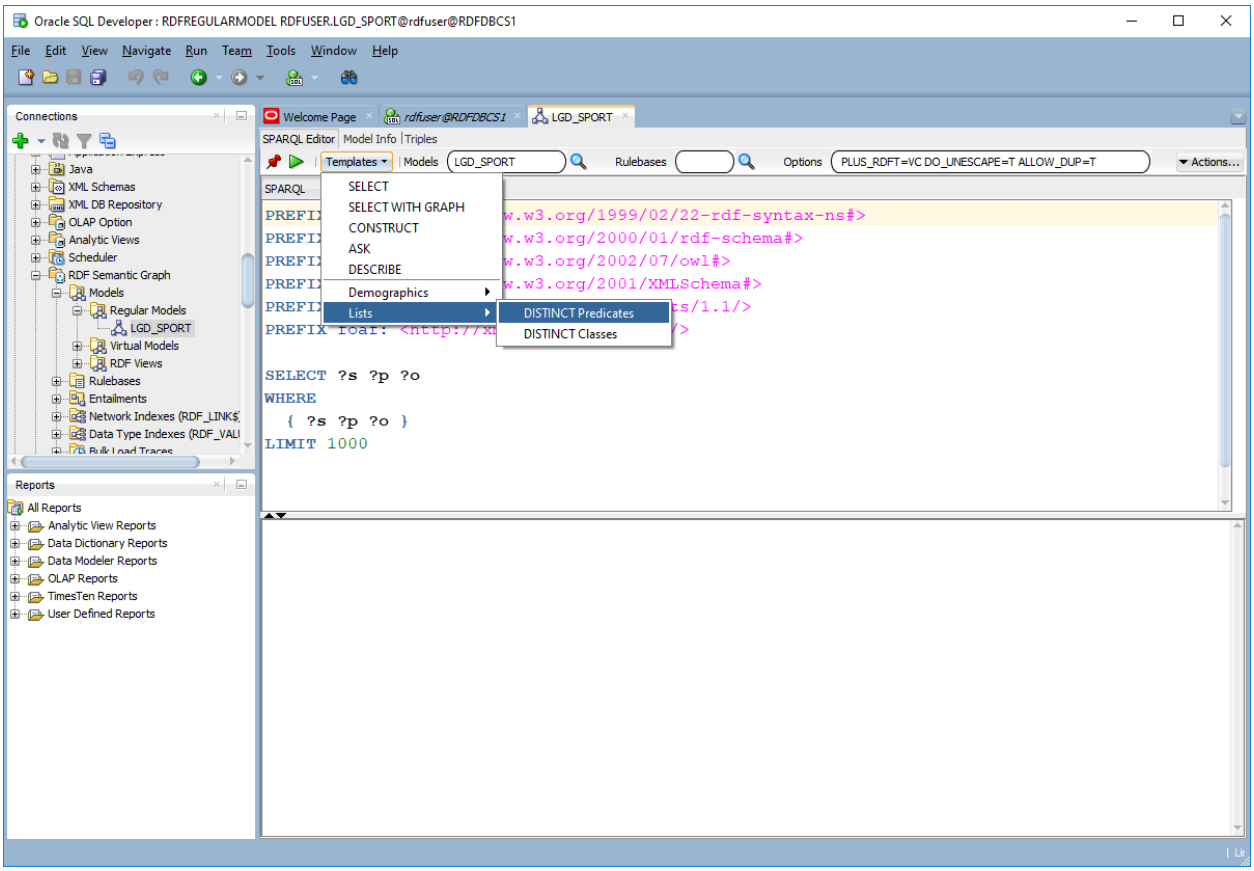
Click Yes when the warning comes up.



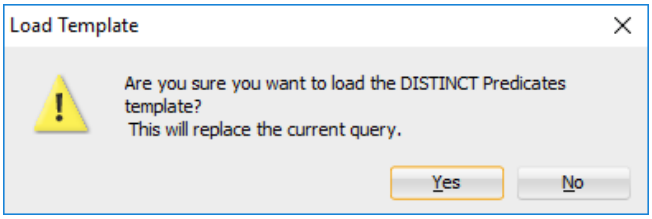
Click the green triangle to run the query.



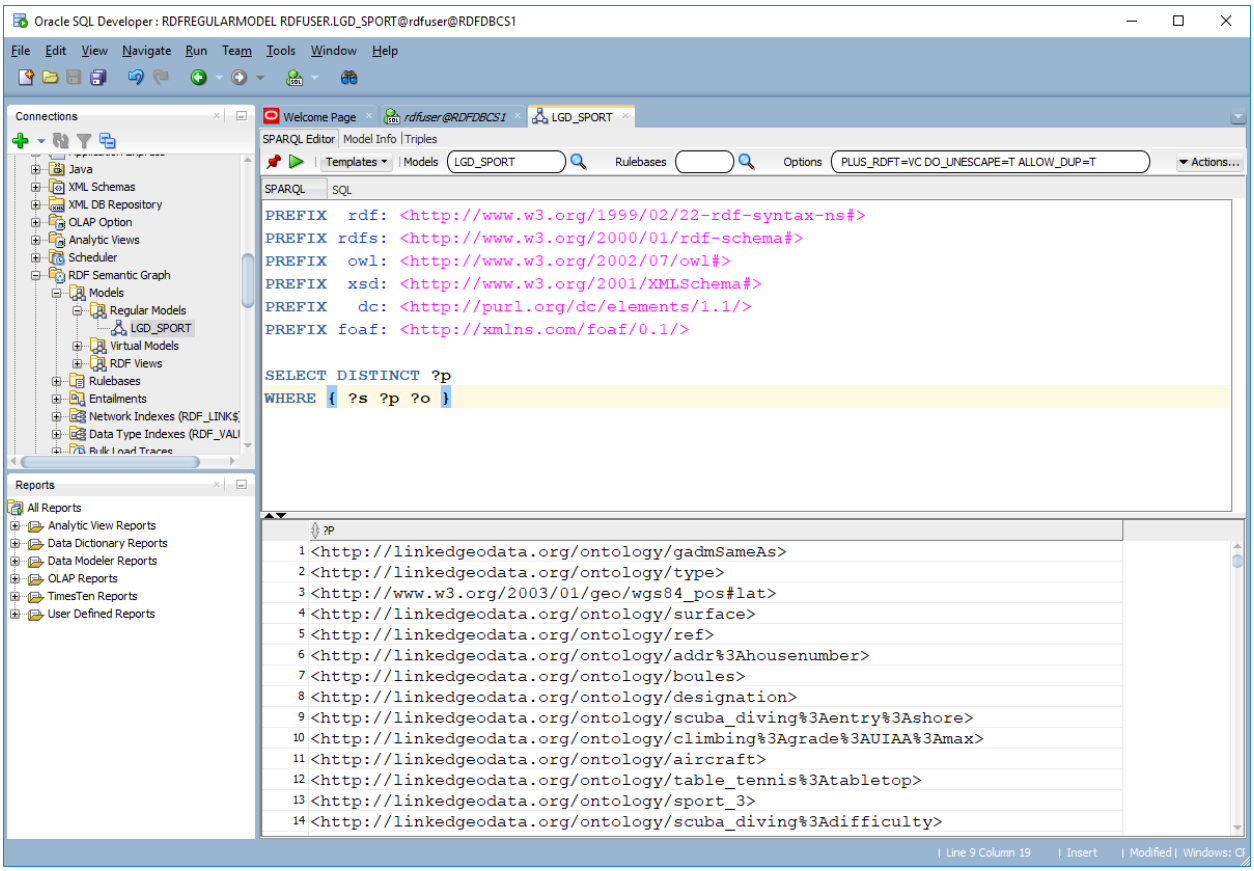
List of distinct predicates.



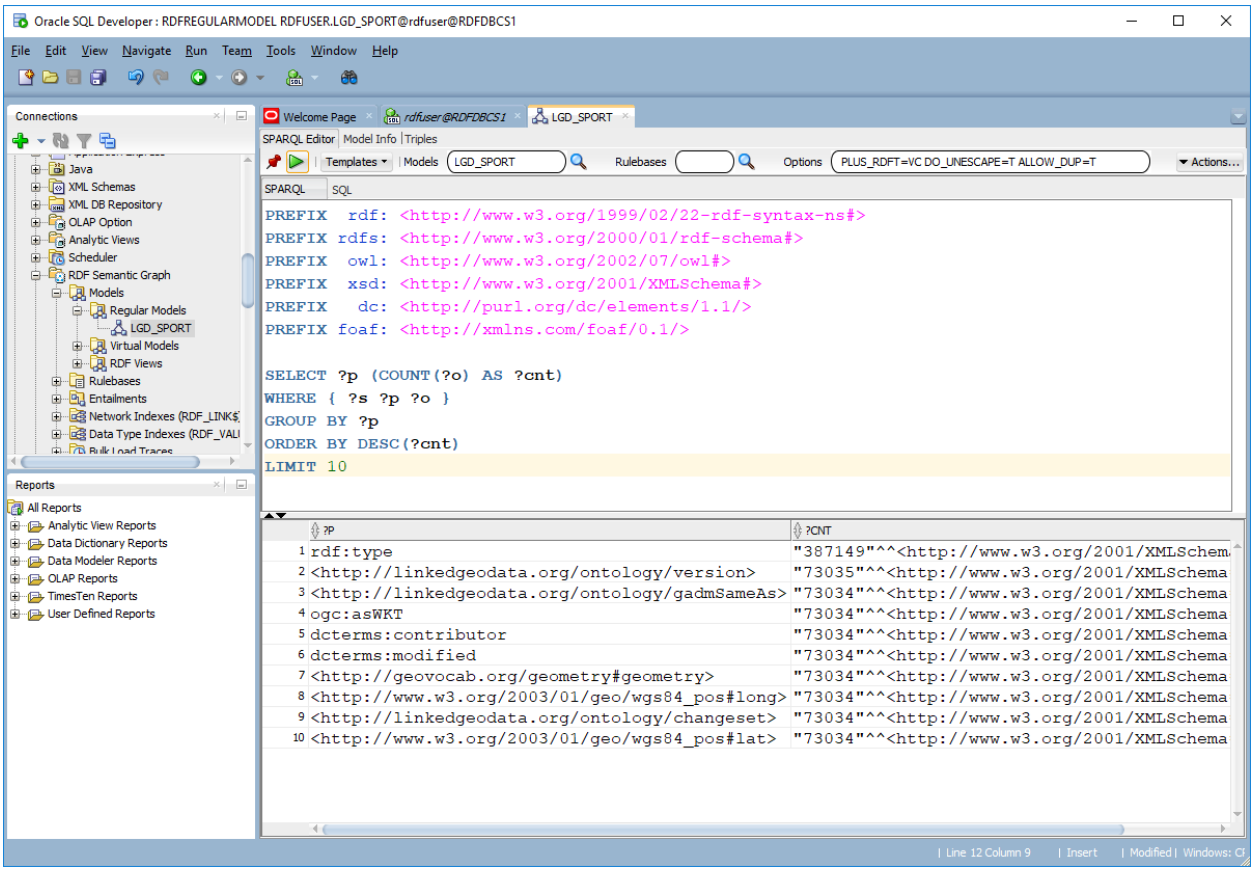
Click Yes.



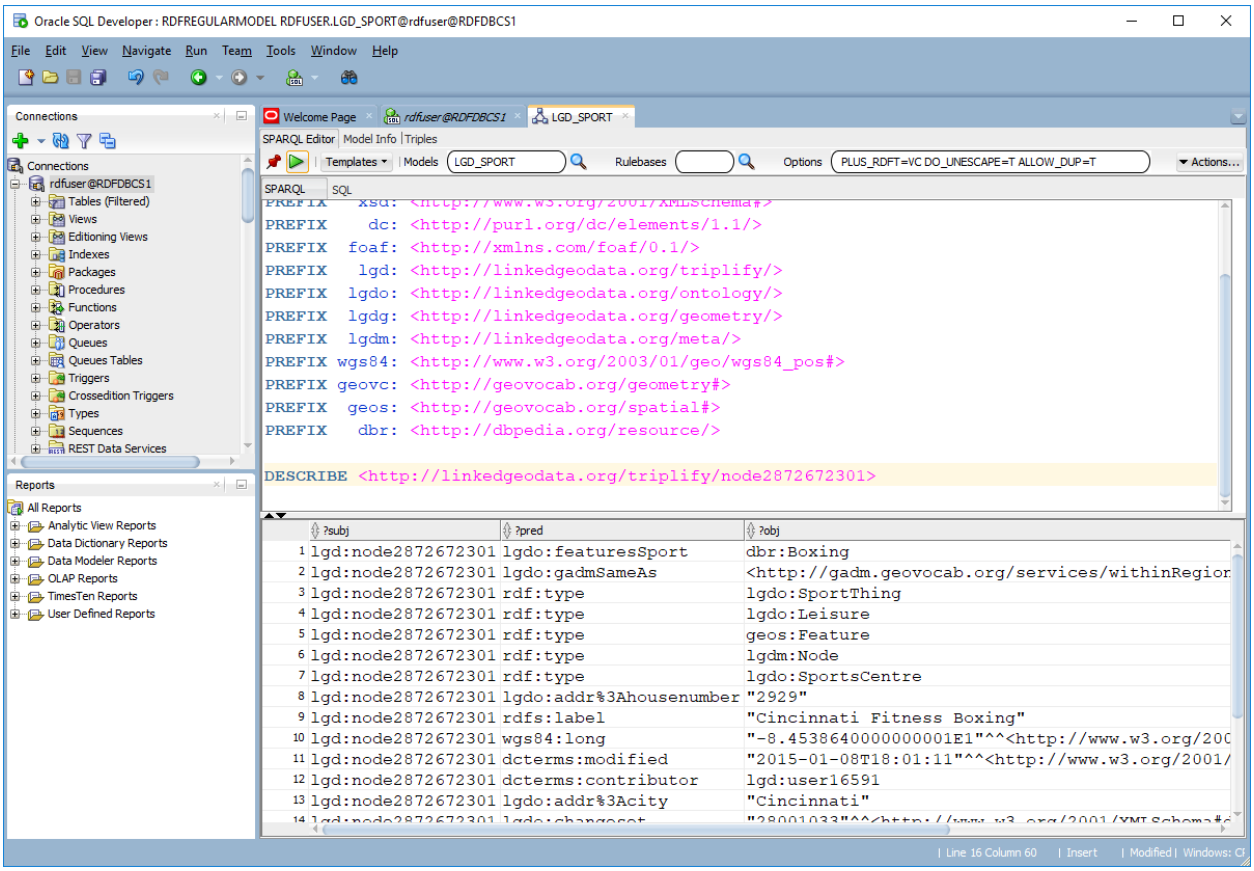
Click the green triangle to run the query.



For the next query, we have modified the distinct predicates query to get the top 10 properties and their triple counts.



In addition to SPARQL SELECT queries, CONSTRUCT and DESCRIBE queries are also supported. The query below describes a particular resource in the LGD_SPORT model. Note that any namespace PREFIXes used in the query will also be used to simplify values in the query result. Here we have added several more PREFIXes.



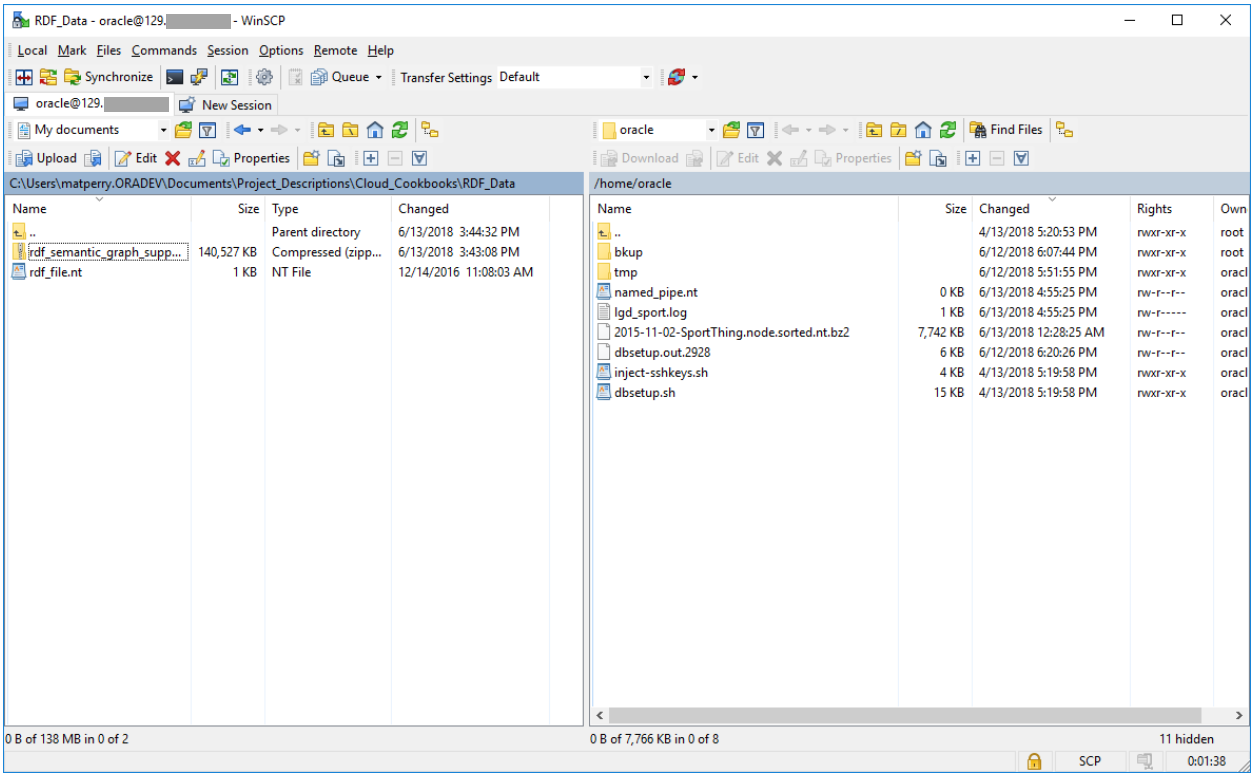
6 SETTING UP A FUSEKI ENDPOINT

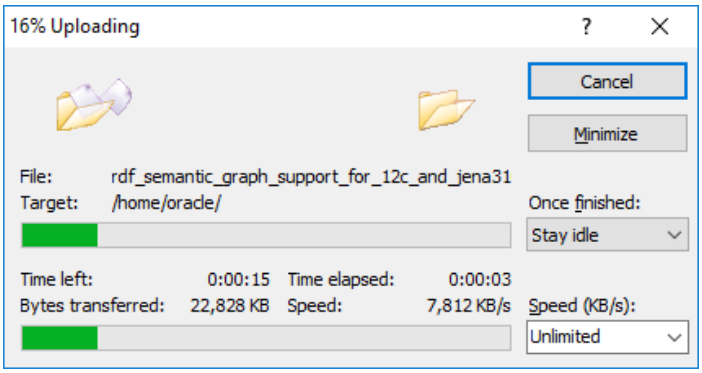
The W3C defines several standard REST APIs for querying and updating RDF data. Oracle RDF Semantic graph leverages Apache Jena Fuseki to provide an implementation of those interfaces. Oracle Support for Apache Jena provides a tight integration between Apache Jena and Oracle RDF Semantic Graph through Oracle-specific implementations of Apache Jena interfaces.

In this Section, we will show how to setup and run Apache Jena Fuseki on our DBCS instance. Fuseki can run as a standalone server or a Java web application. In this case, we will run Fuseki as a standalone server on our DBCS instance. You could also setup an Oracle Java Cloud Service instance and deploy the Fuseki Java web application into the included Web Logic Server instance.

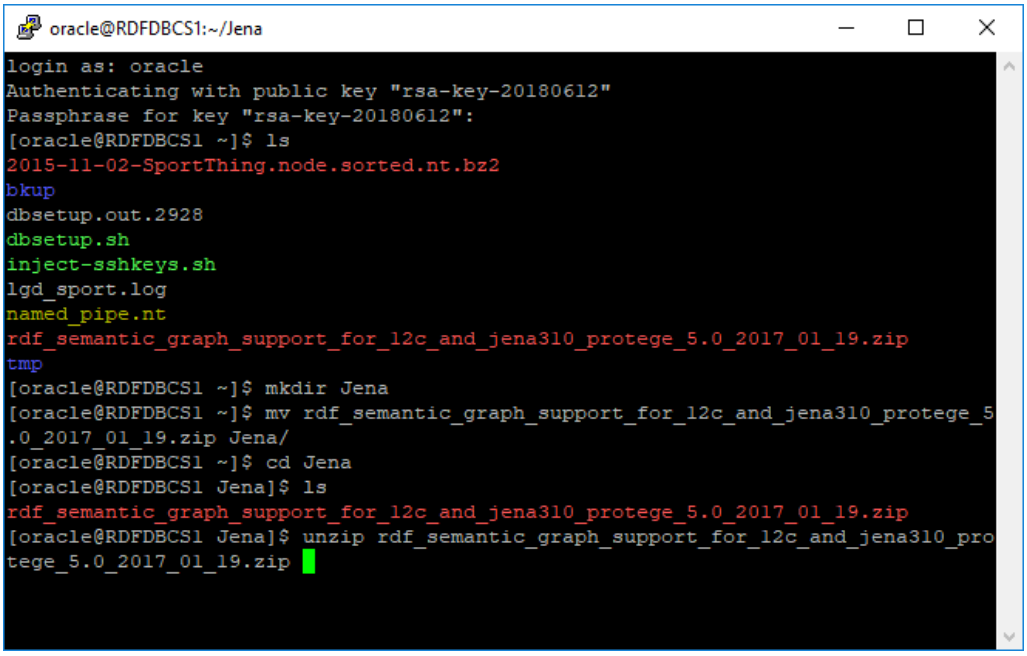
The first step is to download the latest Oracle Support for Apache Jena from OTN. Open a web browser to the Spatial and Graph downloads page. Choose: Download Oracle Database 12c Release 12.1.0.2 Support for Apache Jena 3.1, Apache Jena Fuseki 2.4, and Protégé Desktop 5.0. Note that this download works with Oracle Database version 12.1.0.1 and later.

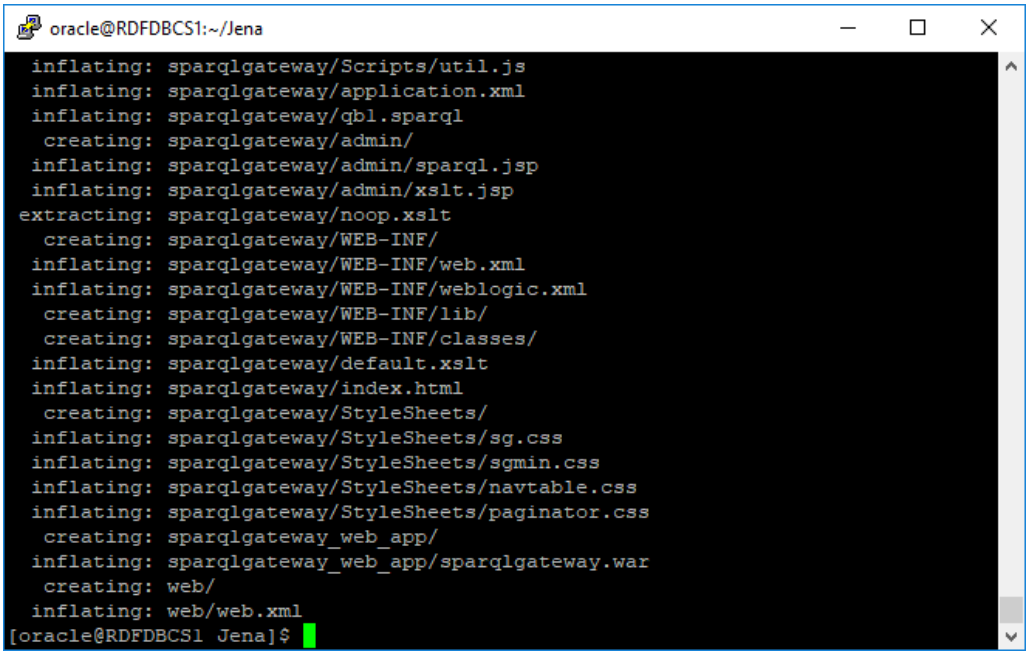
After the download completes, follow the instructions from earlier to connect WinSCP to your DBCS instance. Then copy the downloaded Oracle Support for Apache Jena file to your DBCS instance. In this example, we copied the file to /home/oracle.



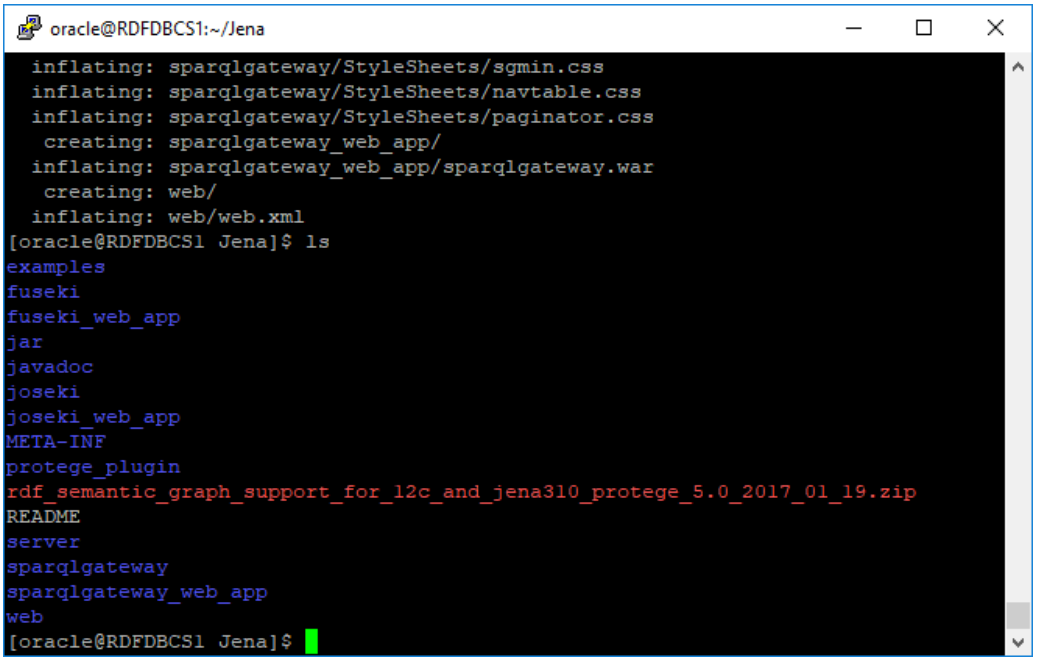


Follow the instructions from earlier to open an SSH connection to your DBCS instance. Create a directory named Jena in /home/oracle. Copy rdf_semantic_graph_support_for_12c_and_jena310_protege_5.0_2017_01_19.zip to the newly created Jena directory and unzip the file.

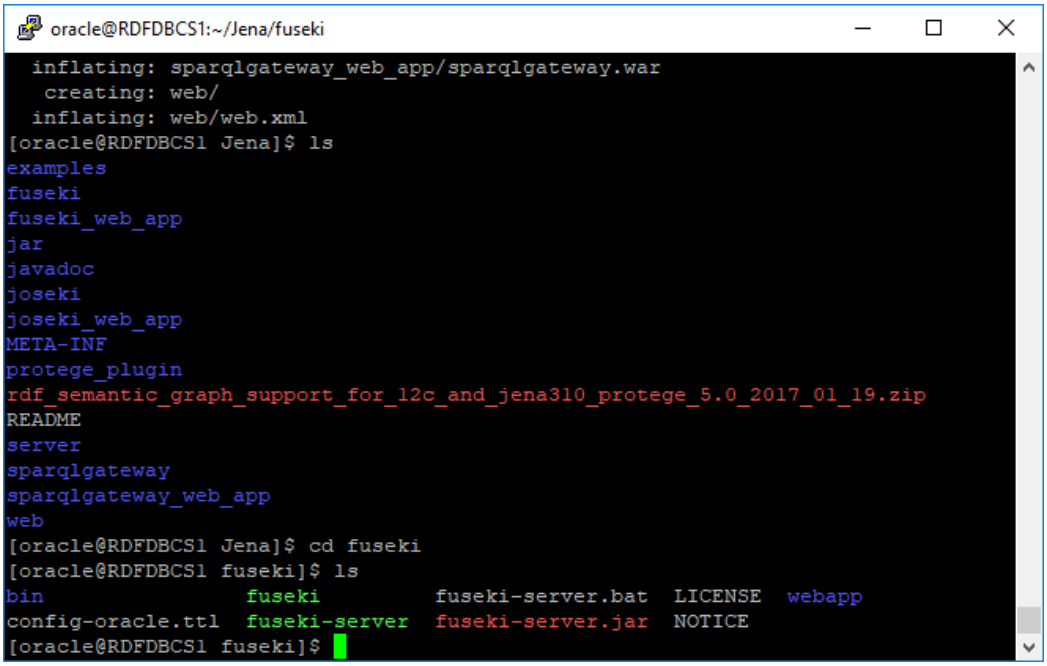




After the unzip command has finished, you will see several directories and a README file.



Now, we will configure Fuseki to access the LGD_SPORT semantic model that we created earlier. Change directory to /fuseki and edit the config-oracle.ttl file.



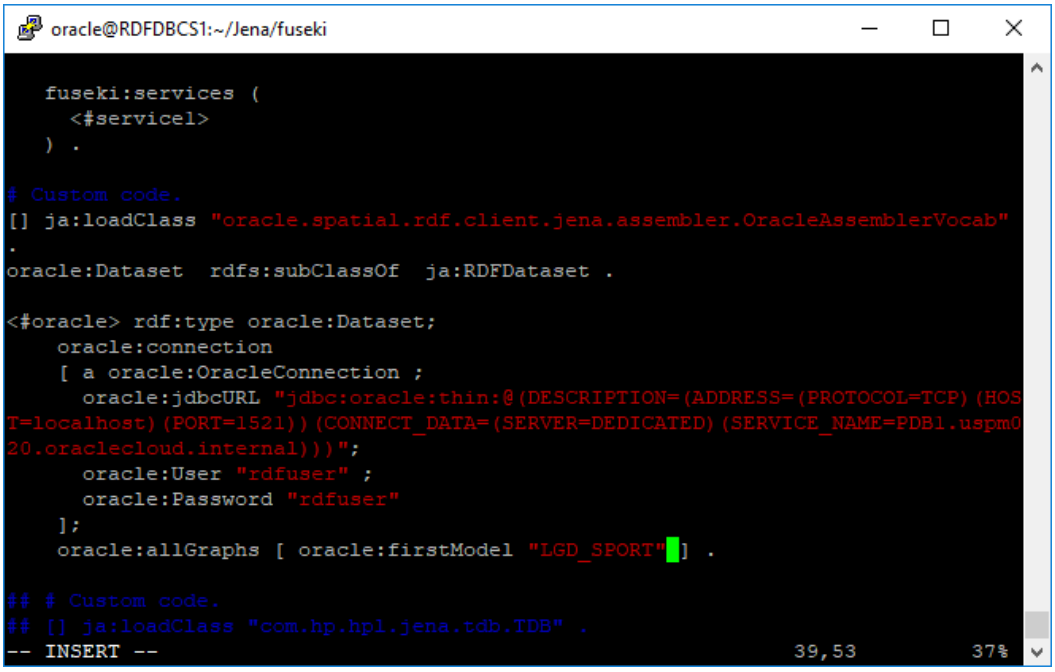
Change the following default <#oracle> dataset specification from

```
<#oracle> rdf:type oracle:Dataset;
    oracle:connection
    [ a oracle:OracleConnection ;
      oracle:jdbcURL
"jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)(PORT=
1521))(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=orcl)))";
```

```
      oracle:User "rdfuser" ;
      oracle:Password "rdfuser"
    ];
  oracle:allGraphs [ oracle:firstModel "TEST_MODEL" ] .
```

to

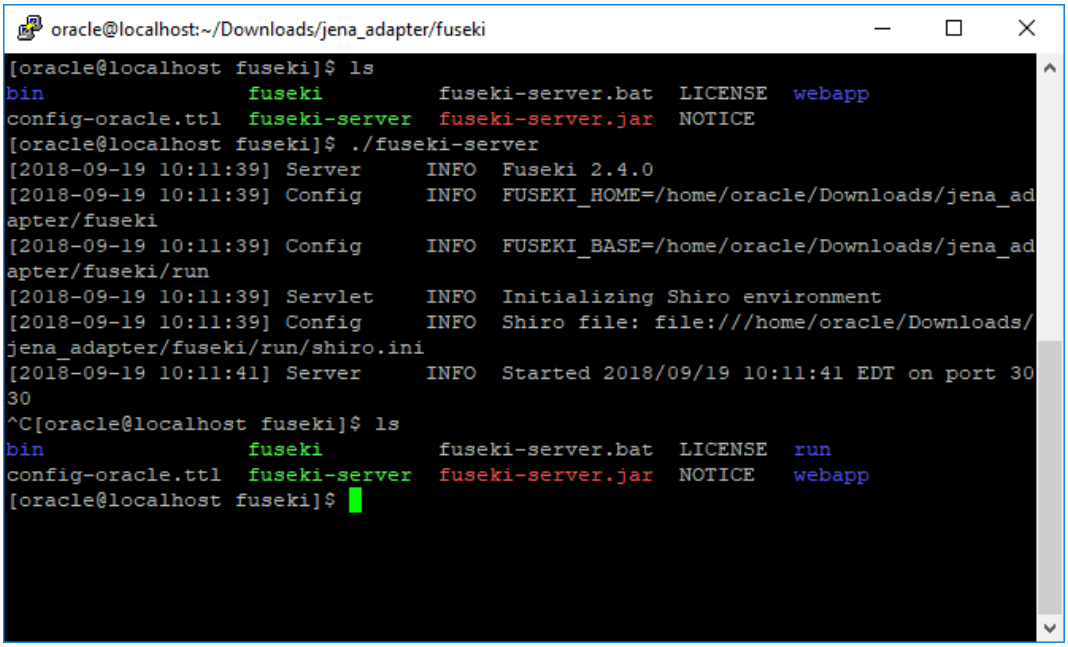
```
<#oracle> rdf:type oracle:Dataset;
    oracle:connection
    [ a oracle:OracleConnection ;
      oracle:jdbcURL
"jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)(PORT=
1521))(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=PDB1.uspm020.oraclecloud.
internal)))";
      oracle:User "rdfuser" ;
      oracle:Password "rdfuser"
    ];
    oracle:allGraphs [ oracle:firstModel "LGD_SPORT" ] .
```



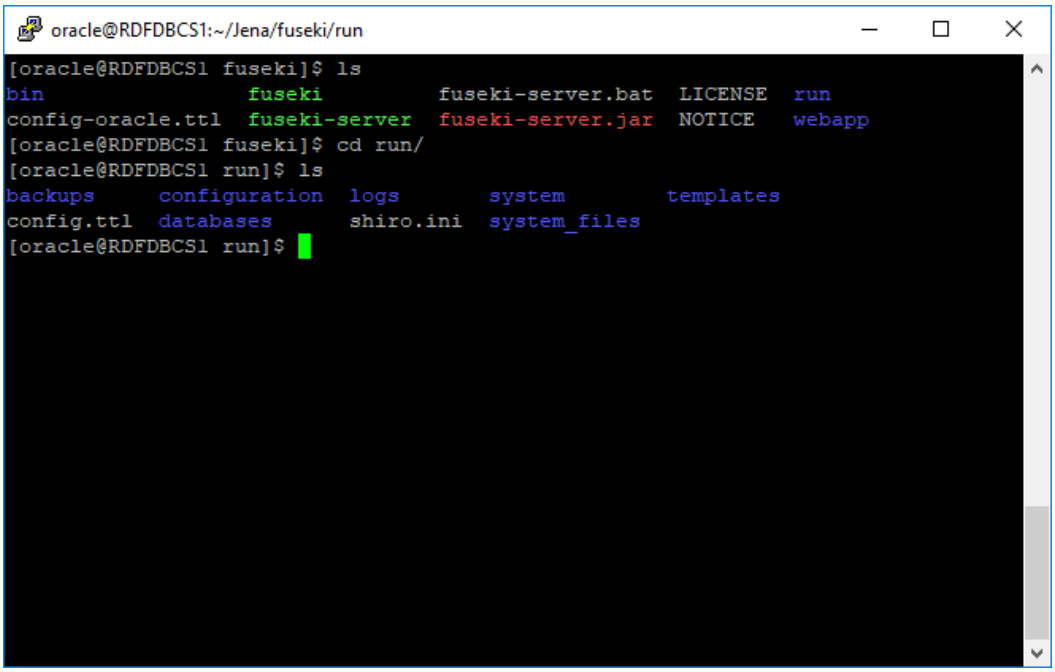
Next, we will change the default shiro.ini configuration to allow non-localhost connections. First, we need to startup Fuseki to create a /run directory. Simply execute the following command in the current /fuseki directory.

```
./fuseki-server
```

Once you see the message that Fuseki has started on port 3030, kill the process with Ctrl-C.



Now the /run directory should be created. Change directory to /run and edit shiro.ini.

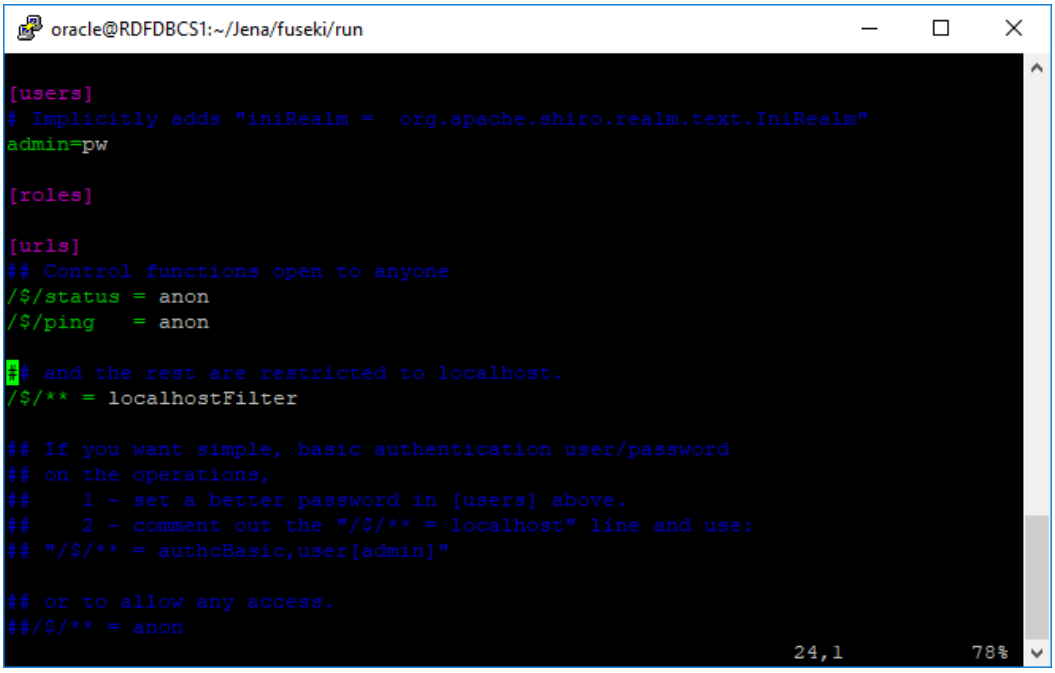


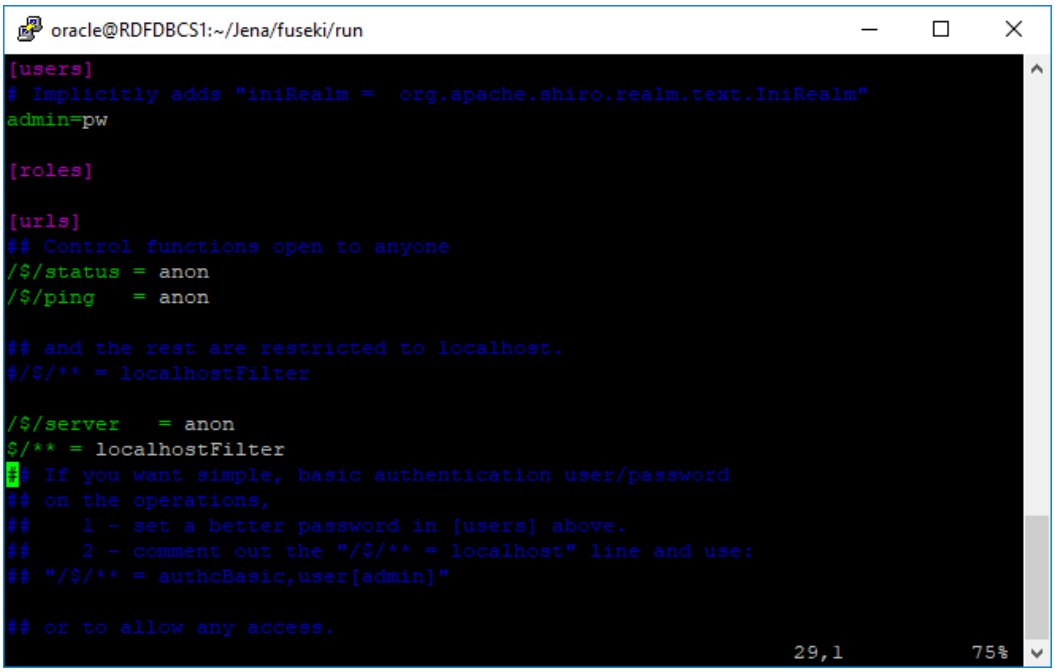
Replace

```
/$/** = localhostFilter
```

with

```
/$/server = anon
$/** = localhostFilter
```

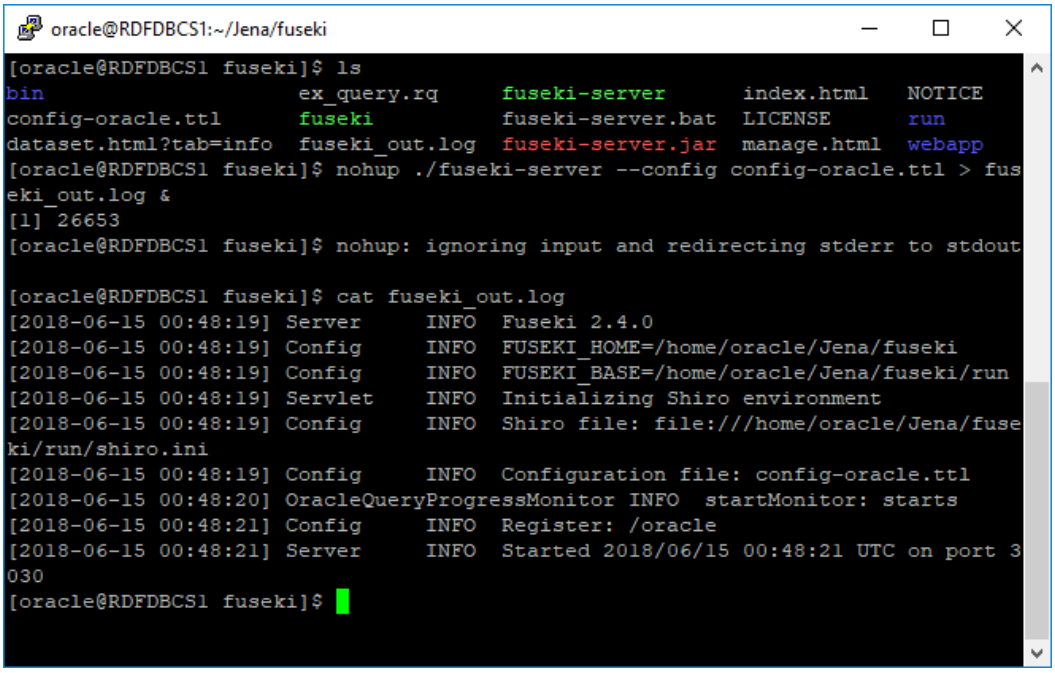




Change directory back to /fuseki and start the Fuseki service by running the following command:

```
nohup ./fuseki-server --config config-oracle.ttl > fuseki_out.log &
```

Note that we are using nohup to prevent the Fuseki process from terminating if our connection is closed.



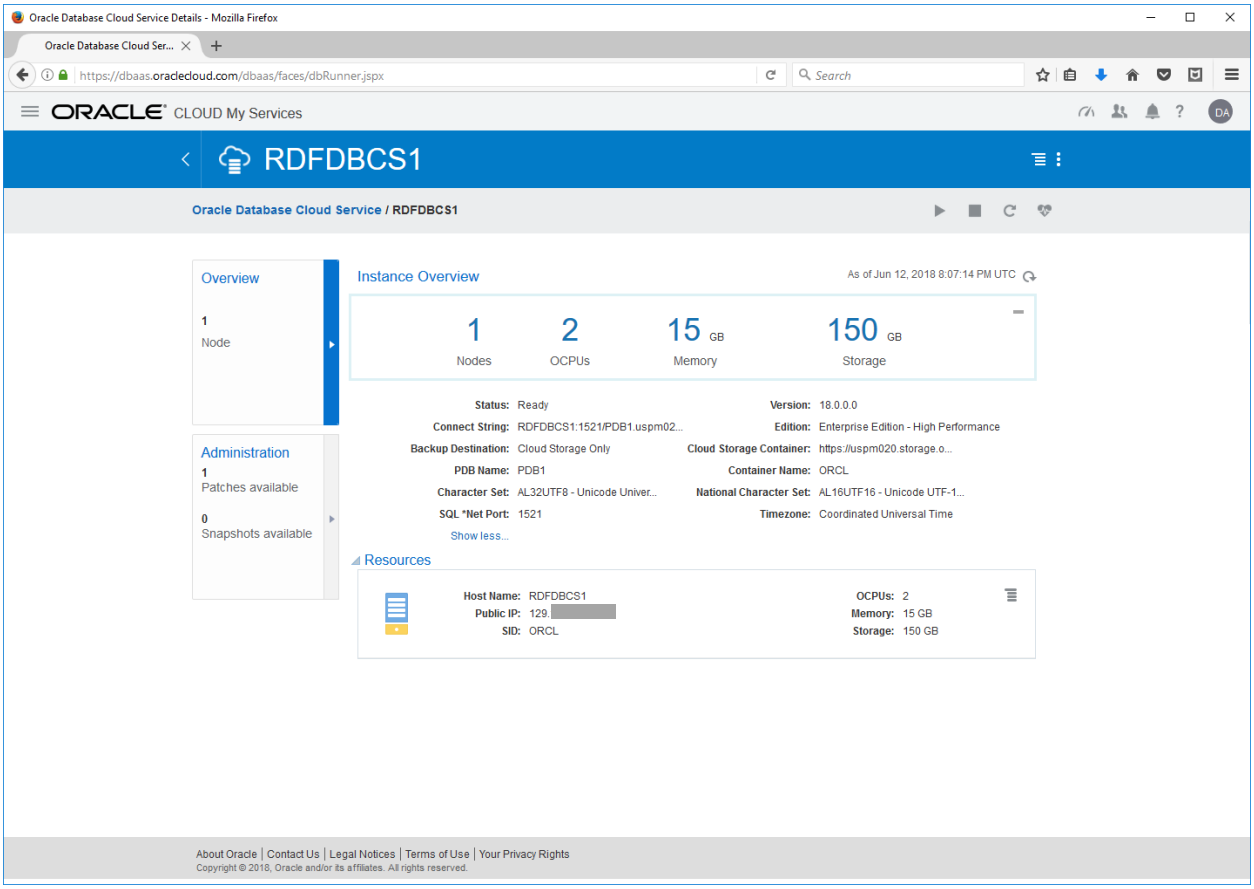
Now the Fuseki server is up and running on port 3030 of our DBCS instance. There are two options for connecting:

1. Create an access rule to open up port 3030 of our DBCS instance to the public internet.
2. Create an SSH tunnel to our DBCS instance for port 3030

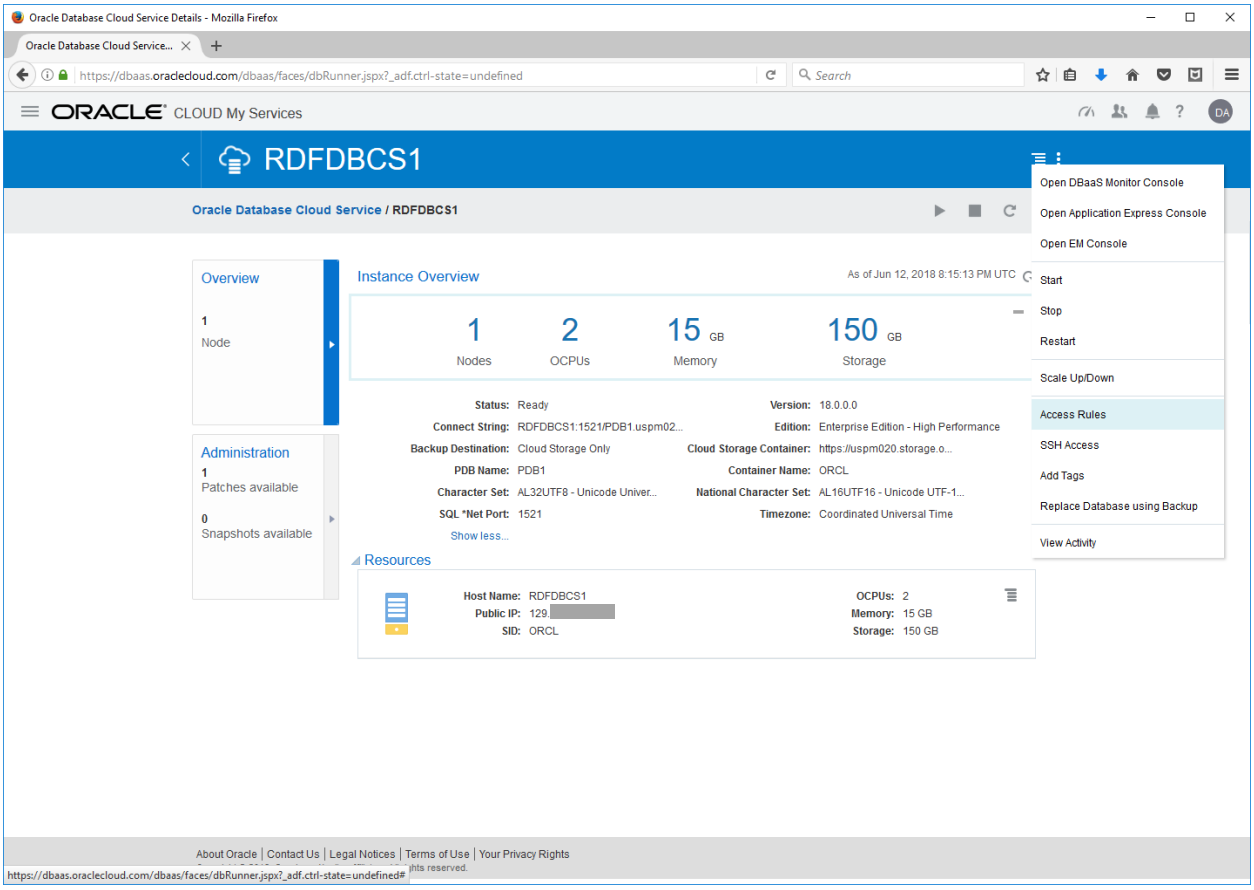
6.1 Method 1: Create an access rule

This Section describes how to create an access rule for port 3030 to open up our SPARQL endpoint to the public internet.

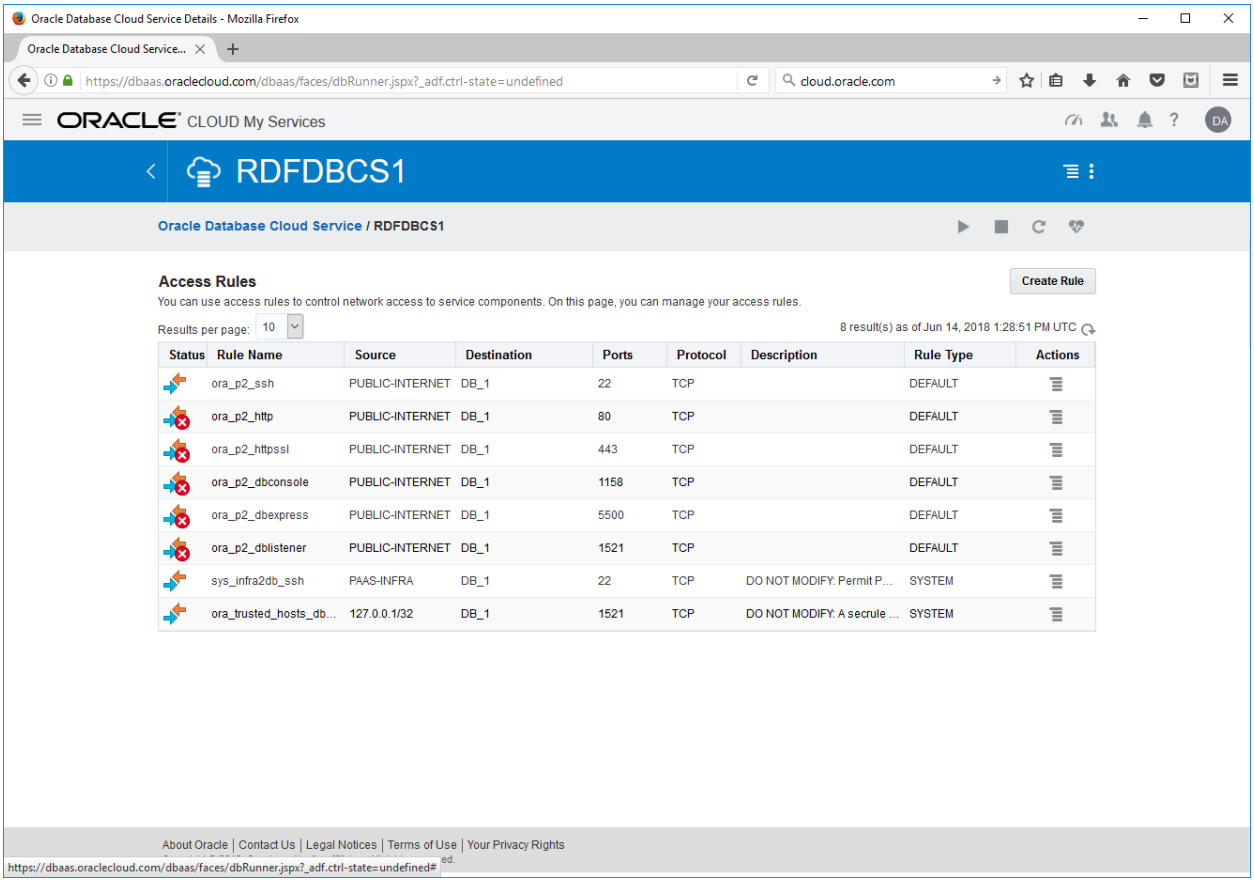
First, login to cloud.oracle.com and open the details page for your DBCS instance.



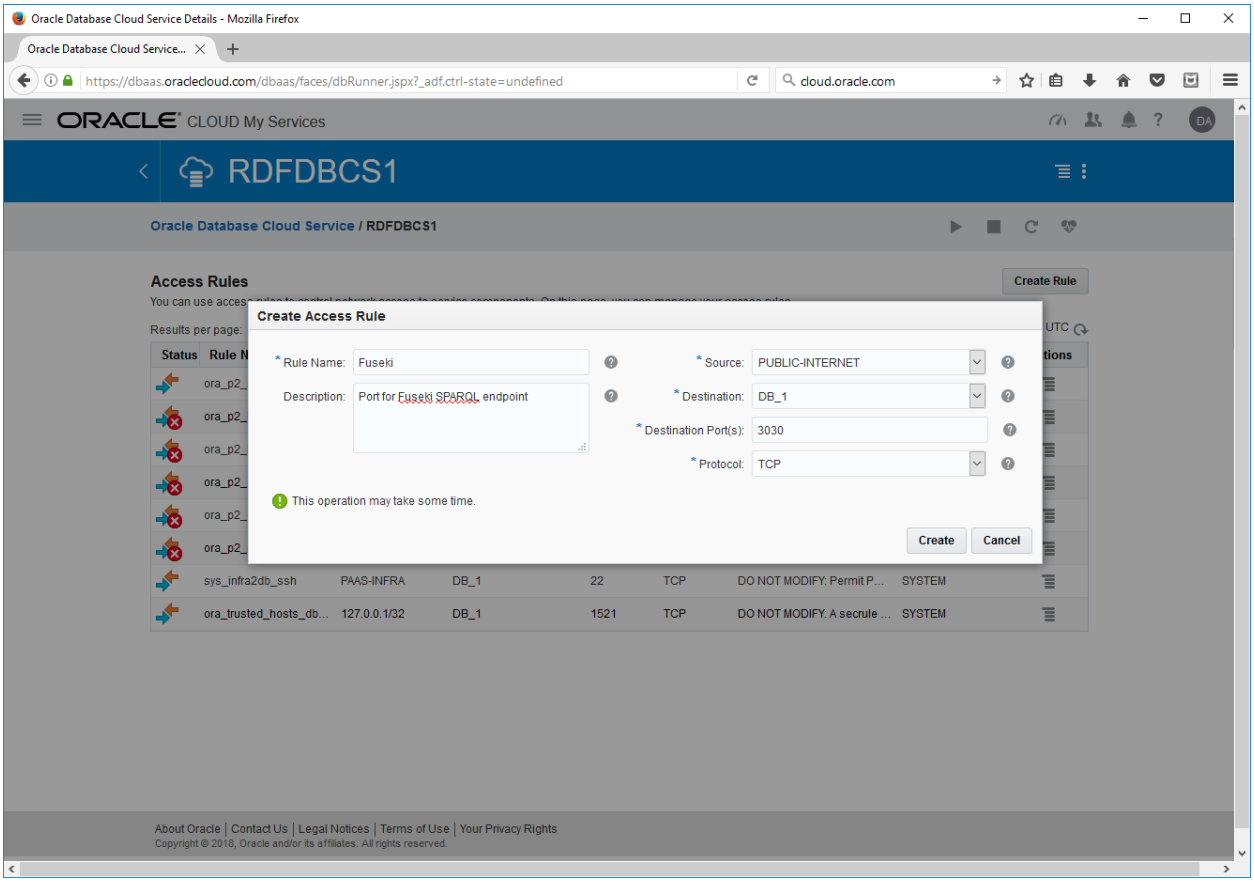
Click the menu on the right and select Access Rules.



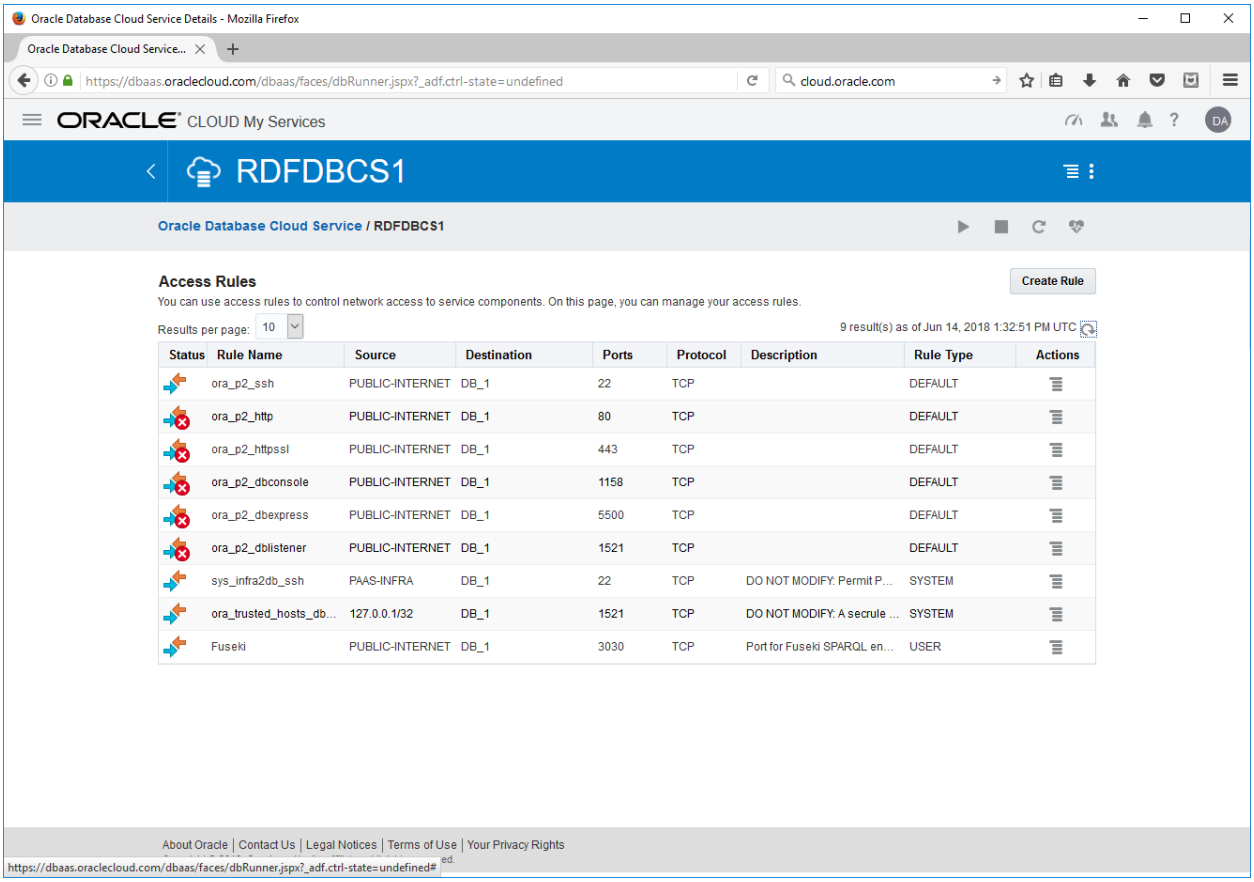
This will bring up the Access Rules page. Click Create Rule to add an access rule for port 3030.



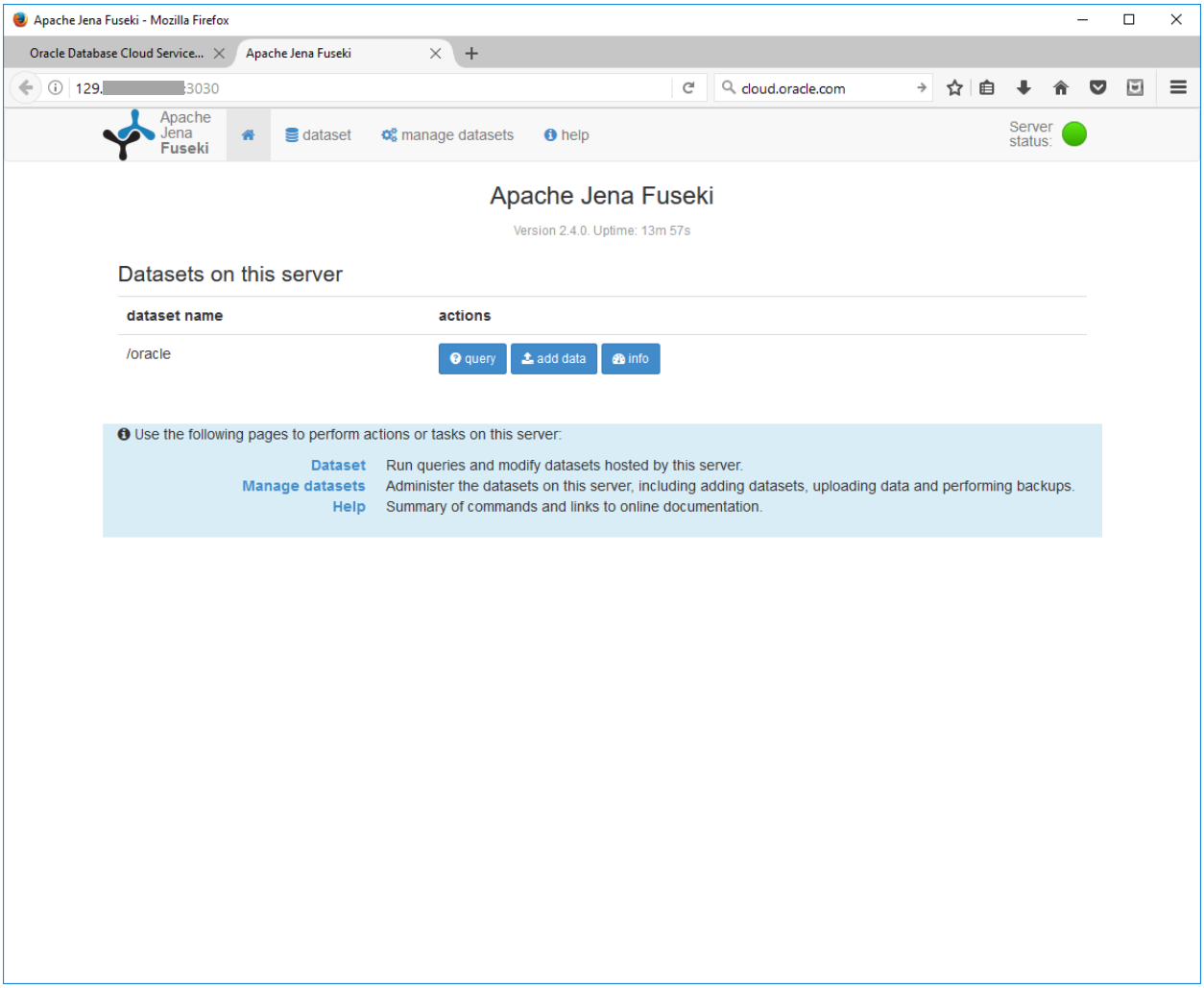
Enter a Rule Name and Description. Choose PUBLIC-INTERNET for Source and DB_1 for Destination. Enter 3030 for Destination Port(s) and choose TCP for Protocol.



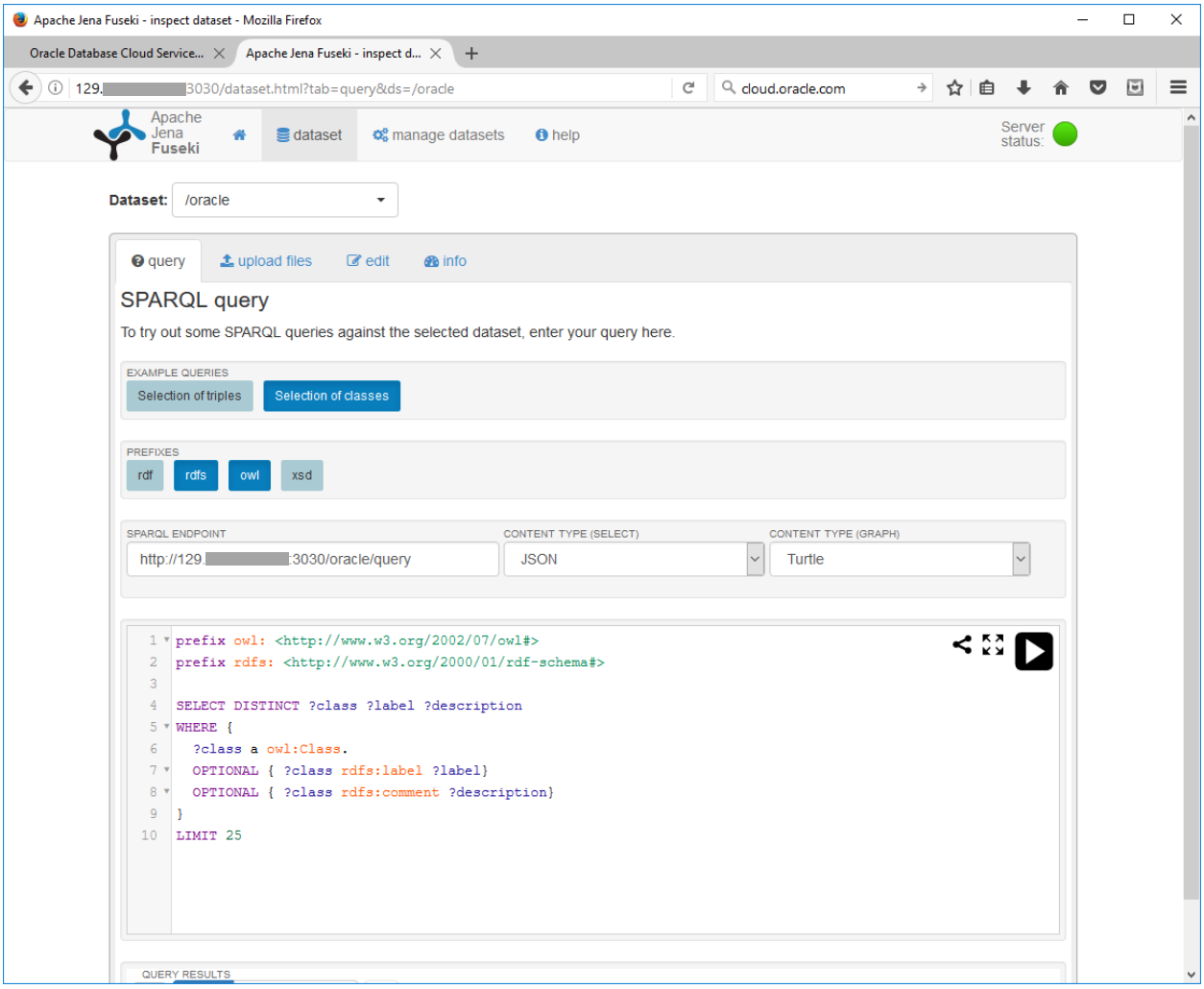
After some time it will complete.



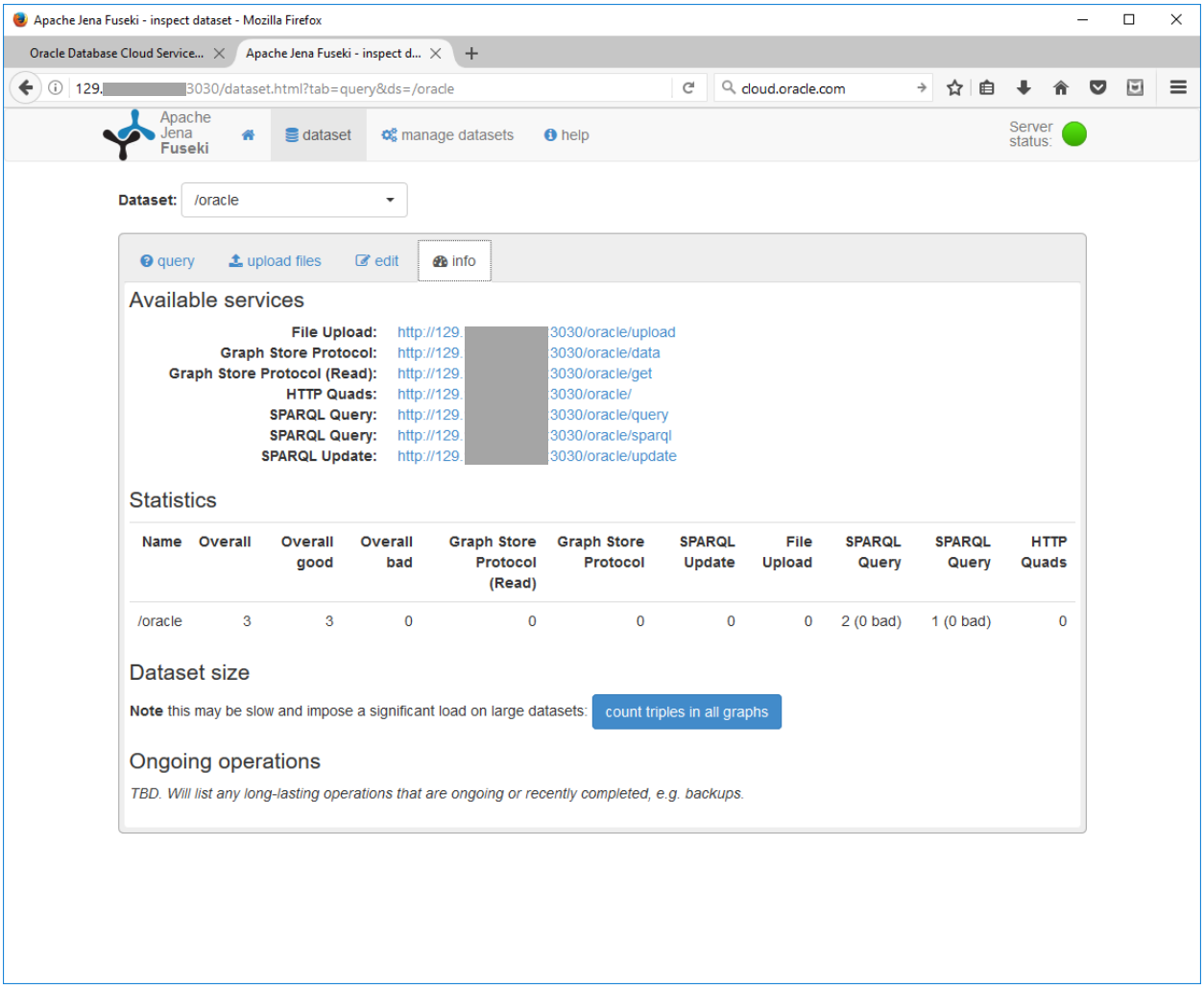
After rule creation has finished, open a web browser to http://<your_dbcs_ip>:3030



Click query to open the SPARQL query interface.



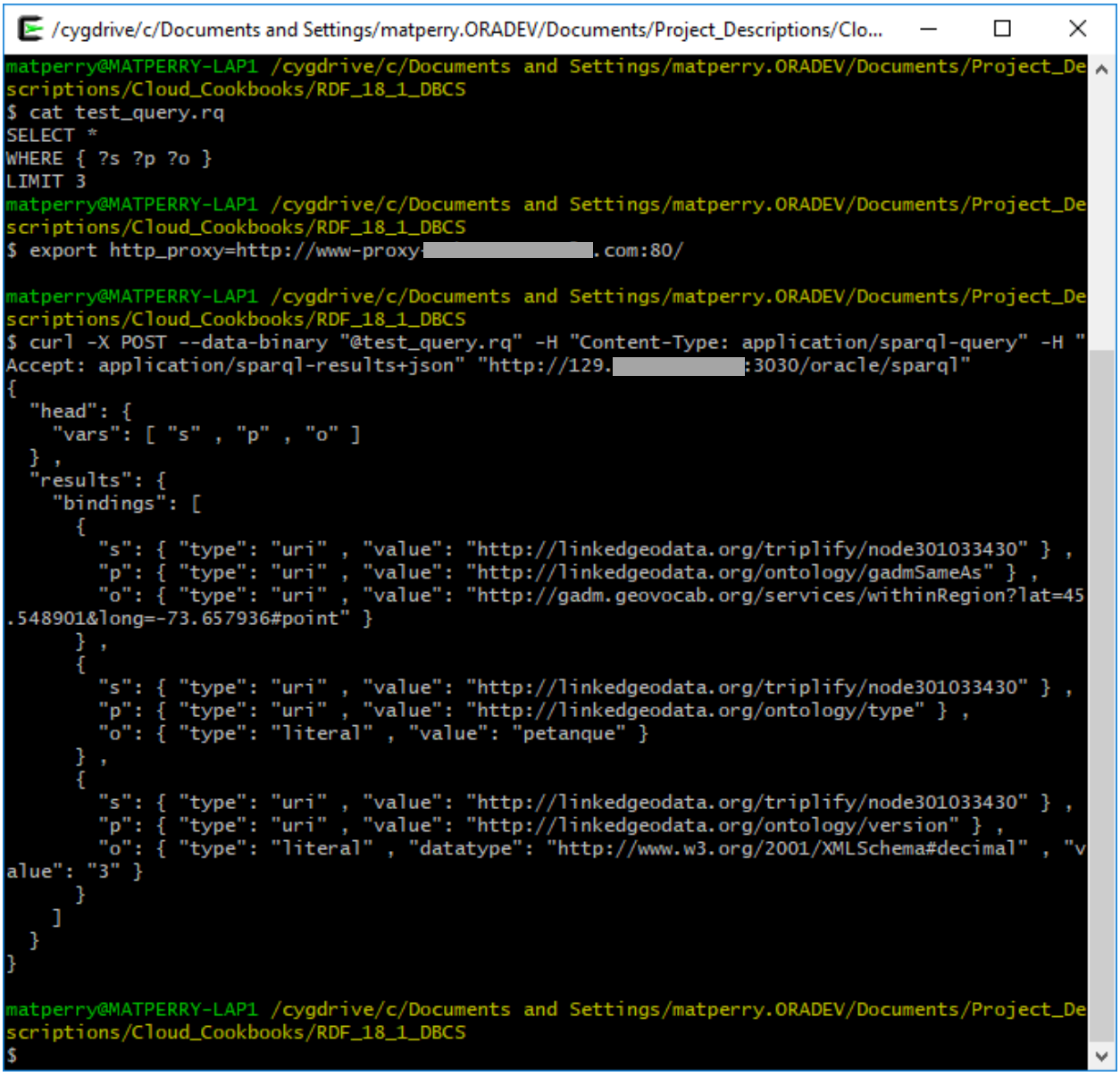
Click info to see all the available REST endpoints.



We can also use Cygwin to test the REST SPARQL interface directly with curl. When run from a Cygwin terminal, the following command

```
curl -X POST -data-binary "@test_quey.rq" -H "Content-Type:
application/sparql-query" -H "Accept: application/sparql-results+json"
"http://<your_dbcs_ip>:3030/sparql"
```

will send the SPARQL query in the file test_query.rq to the Fuseki endpoint running on your DBCS instance and print the result to stdout.



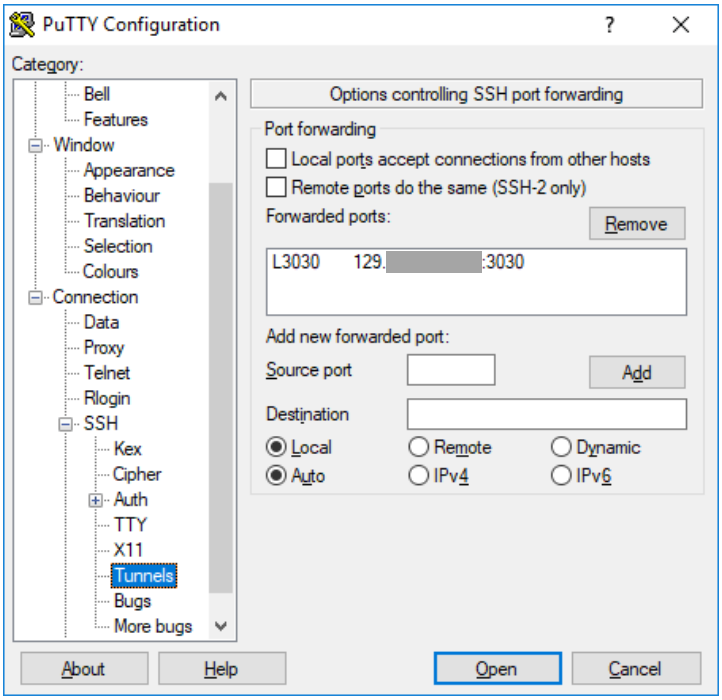
Now we have a W3C standard SPARQL endpoint up and running on our DBCS instance.

6.2 Method 2: Use PuTTY to create an SSH tunnel to our DBCS instance for port 3030

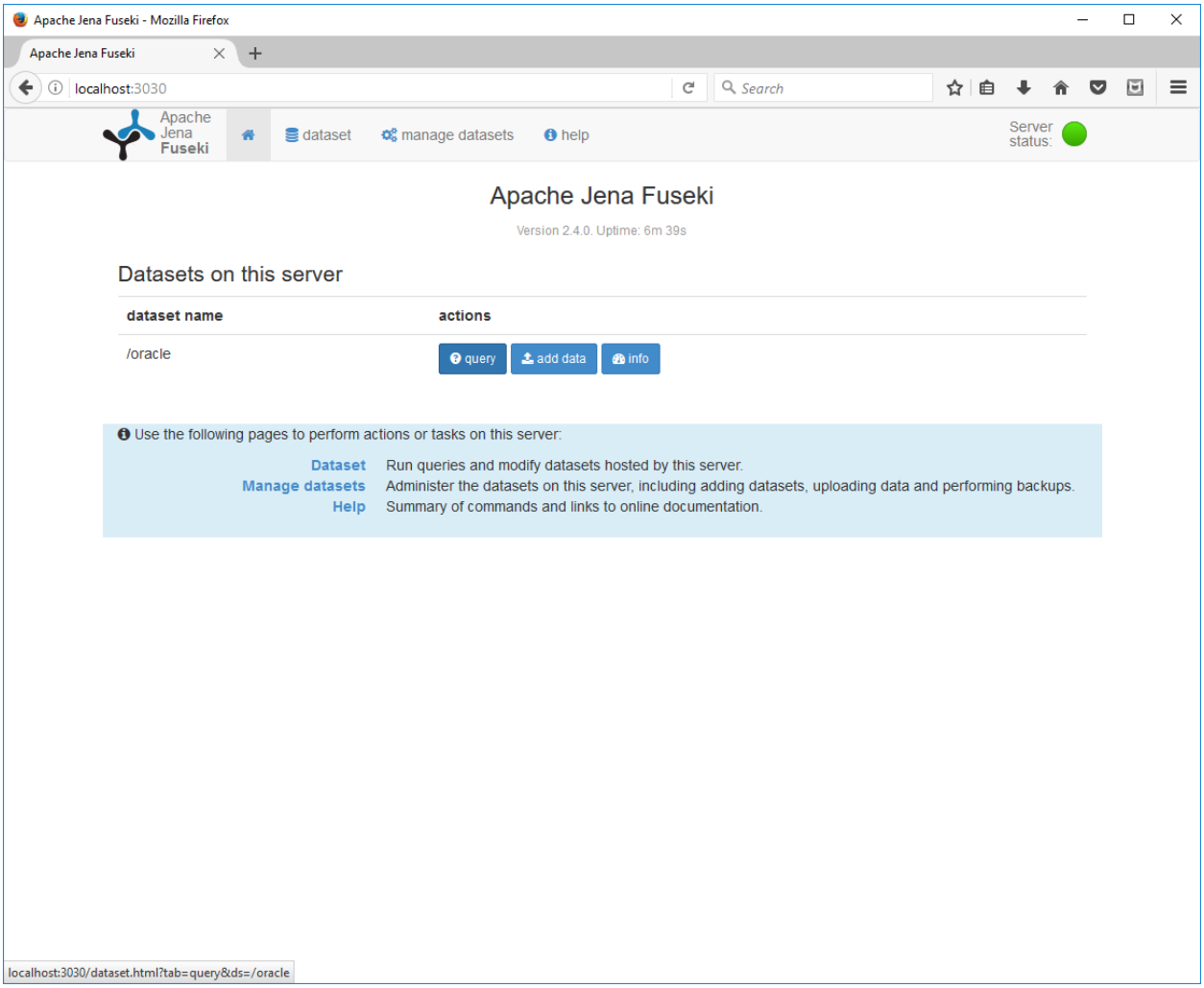
In this Section, we will show how to connect to our SPARQL endpoint on port 3030 without opening port 3030 to the public internet. This setup does not require the port 3030 access rule that we created previously. That access rule can be disabled or dropped.

First, use PuTTY to open an SSH connection to your DBCS instance and start fuseki-server on the DBCS instance the same way we did before.

Next, use PuTTY to create an SSH tunnel to forward port 3030 on your local machine to port 3030 on your DBCS instance. The steps are the same as the ones we used earlier to forward port 1521 for SQL Developer, but we are using port 3030 in this case.



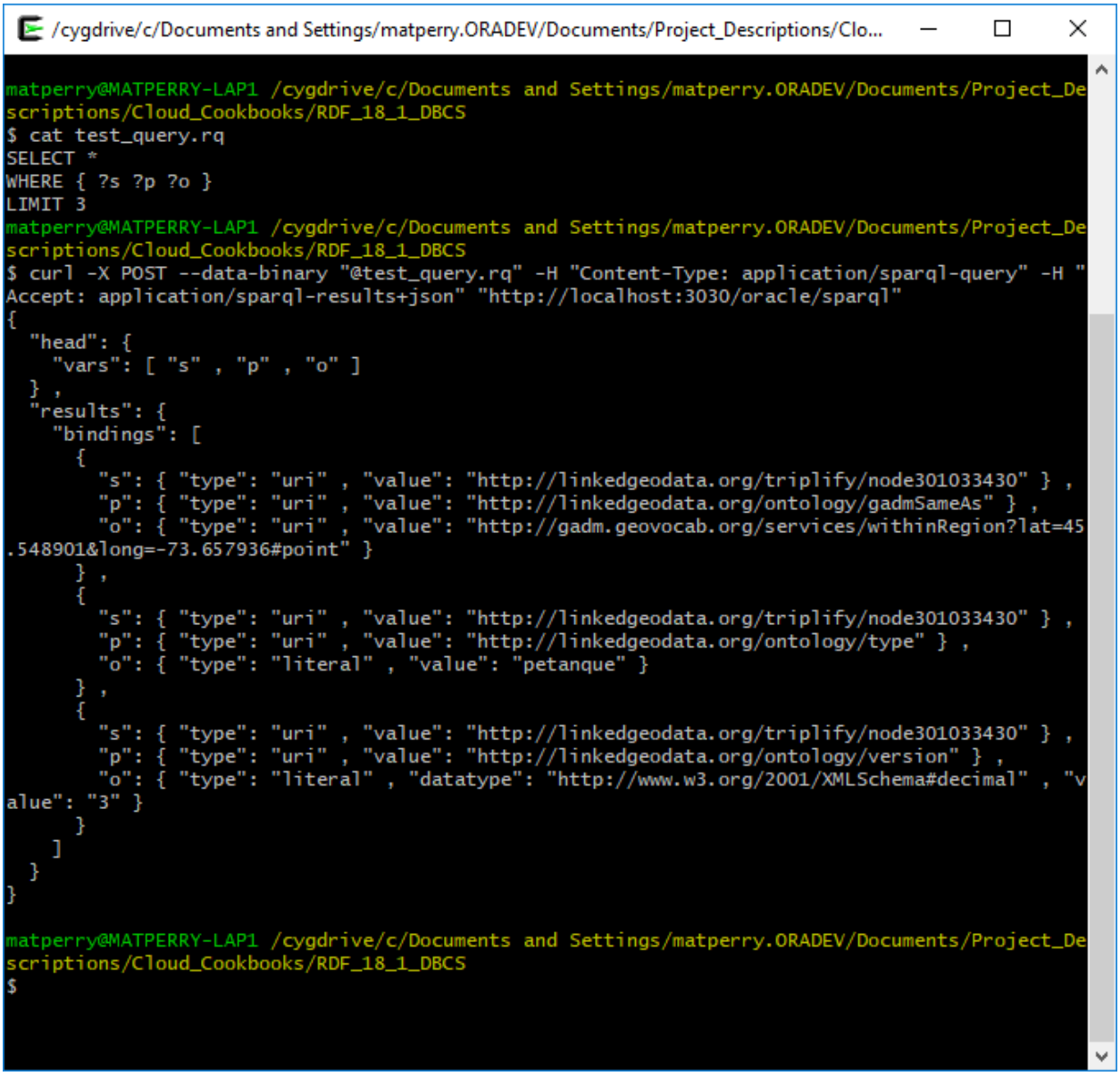
Click Open to open the SSH tunnel and then open a web browser to http://localhost:3030.



Now we have used an SSH tunnel to connect to the SPARQL endpoint running on our DBCS instance.

We can also use Cygwin to directly test the SPARQL REST interface over the SSH tunnel. Enter the same curl command as before but use http://localhost:3030/sparql as the endpoint URL this time:

```
curl -X POST -data-binary "@test_quey.rq" -H "Content-Type:
application/sparql-query" -H "Accept: application/sparql-results+json"
"http://localhost:3030/sparql"
```



Now, we have successfully used an SSH tunnel to access a W3C-standard SPARQL REST endpoint running on our DBCS instance.