Nipun Kularathne Follow
Jun 2, 2020 · 9 min read · Listen

Oracle Autonomous Database with REST API (ORDS) in action- Part I

This topic concludes in to two sections as it has some clear server related configurations and client related configurations. You can access the second part of this article which is related to client related configurations here but you would first need to go through the first section and get an idea about the server related configurations before jumping to the second section.

In these two articles we will look at how we can use an Oracle Autonomous Database (OAD) practically to store data in the cloud and how to retrieve/update/delete the data from our client applications using REST API services. We would be using Oracle provided Oracle Rest Data Services (ORDS) to achieve this.

To provision an OAD I have used the Oracle free tier which you'll be automatically eligible for when you create an Oracle cloud account.

In simple terms we are looking at how to host a database(DB) in a separate secure environment where the management of the DB is done by an automated database administration tools for us, and we are going to connect to that database using web service endpoints to perform Insert/update/delete operations using our client applications. If you go through my previous article on Oracle Autonomous Database you would understand the unique nature and benefits of using these self-managed databases.

Things we are going to cover in this first section

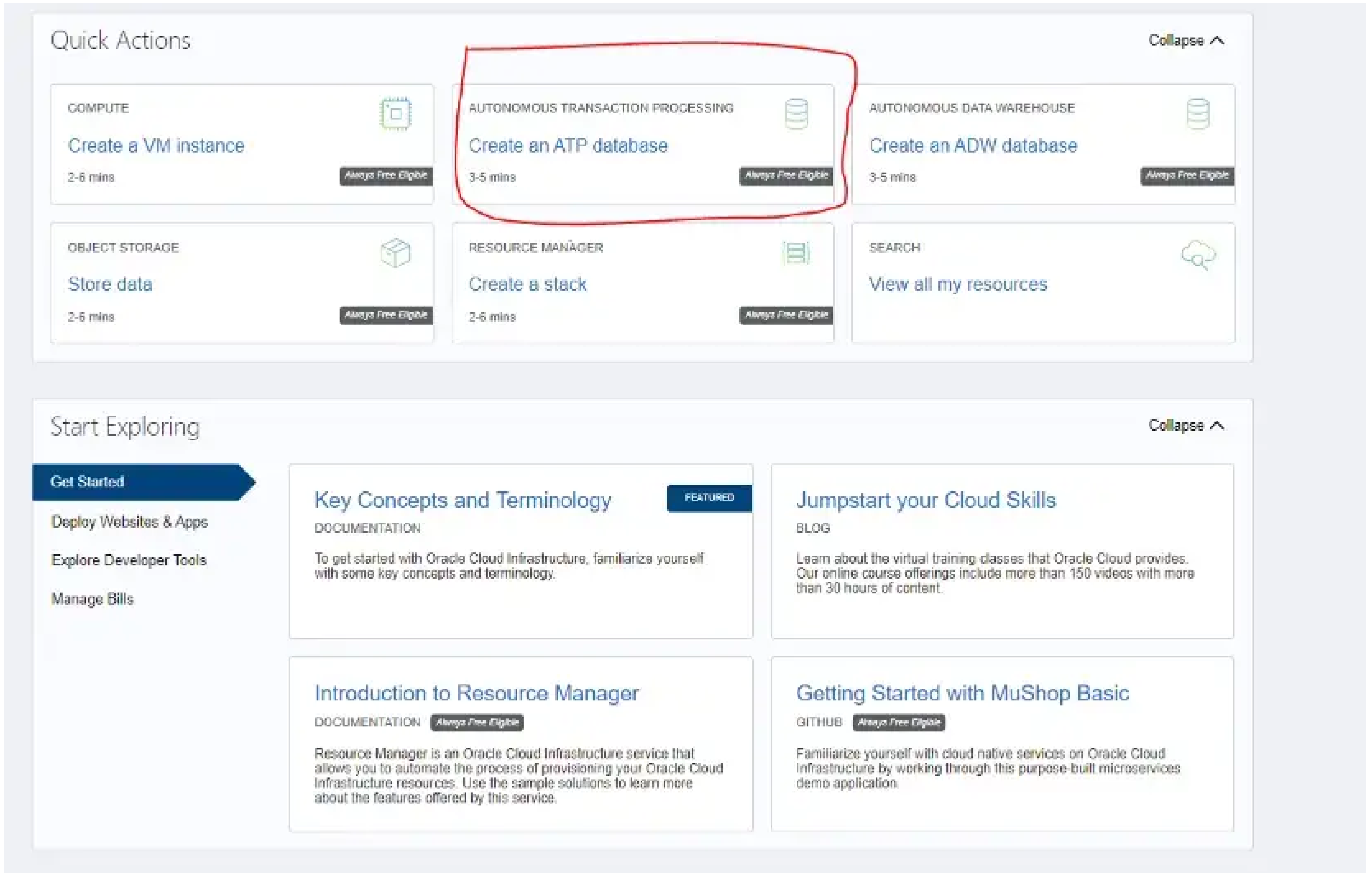
● Provisioning an OAD in Oracle cloud free tier
● Connecting to the OAD (using standalone oracle db administrating app)
● Creating a user and giving necessary grants to the OAD for that user to connect and create objects
● Creating a table and enable Oracle Rest data services and restrict the access to outsiders

Provisioning an OAD in Oracle cloud free tier

The first thing we need to do is to register get a free cloud account in Oracle. For that you need to get registered in https://cloud.oracle.com. You will be asked to enter your credit card details but you will not be charged unless you choose to upgrade the account to a paid one. You can find more information on Oracle cloud free tier and getting started info using this link.

Now that you have created the free account, you have always free services offered to you by Oracle which includes 2 Oracle Autonomous databases. You can log on to your cloud account to provision an OAD according to below instructions.

When you log in to the cloud account you will get the below dashboard where you can click on the marked Create ATP Database option. OAD has two variants to choose from depending on the workload type you are going to have on them. For now let’s choose an Autonomous Transaction Processing (ATP) database to do our job here. Other variant is the Autonomous Data Warehousing.


Quick Actions

Collapse

COMPUTE
Create a VM instance
2-6 mins
Always Free Eligible

AUTONOMOUS TRANSACTION PROCESSING
Create an ATP database
3-5 mins
Always Free Eligible

AUTONOMOUS DATA WAREHOUSE
Create an ADW database
3-5 mins
Always Free Eligible

OBJECT STORAGE
Store data
2-6 mins
Always Free Eligible

RESOURCE MANAGER
Create a stack
2-6 mins
Always Free Eligible

SEARCH
View all my resources

Start Exploring

Collapse

Get Started
Deploy Websites & Apps
Explore Developer Tools
Manage Bills

Key Concepts and Terminology
FEATURED
DOCUMENTATION
To get started with Oracle Cloud Infrastructure, familiarize yourself with some key concepts and terminology.

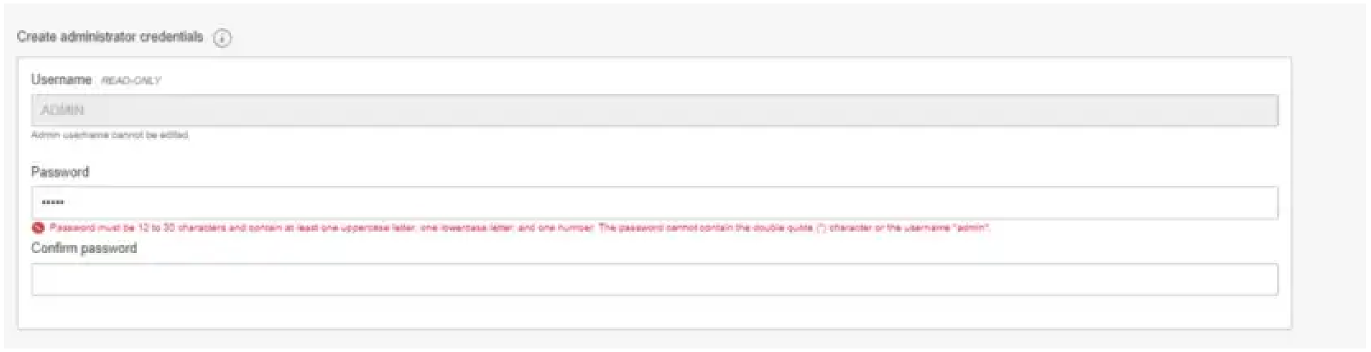
Jumpstart your Cloud Skills
BLOG
Learn about the virtual training classes that Oracle Cloud provides. Our online course offerings include more than 150 videos with more than 30 hours of content.

Introduction to Resource Manager
DOCUMENTATION Always Free Eligible
Resource Manager is an Oracle Cloud Infrastructure service that allows you to automate the process of provisioning your Oracle Cloud Infrastructure resources. Use the sample solutions to learn more about the features offered by this service.

Getting Started with MuShop Basic
GITHUB Always Free Eligible
Familiarize yourself with cloud native services on Oracle Cloud Infrastructure by working through this purpose-built microservices demo application.

When you click on the marked button you will get a form fill and to choose different features of the OAD. For now let's just keep everything as it is as our objective is to create a database in the cloud and get something out of it with the minimum configurations possible. Therefore we are only interested in the Admin password field and the display name field in the provisioning form in this exercise.

You can give a meaningful name to as the display name of the database which makes it easy for you to identify the database later. And just type a strong password adhering to the requirements that displayed on the same page in red. You cannot choose a username as it is already selected as ADMIN.


Create administrator credentials

Username READ-ONLY

ADMIN

Admin username cannot be edited.

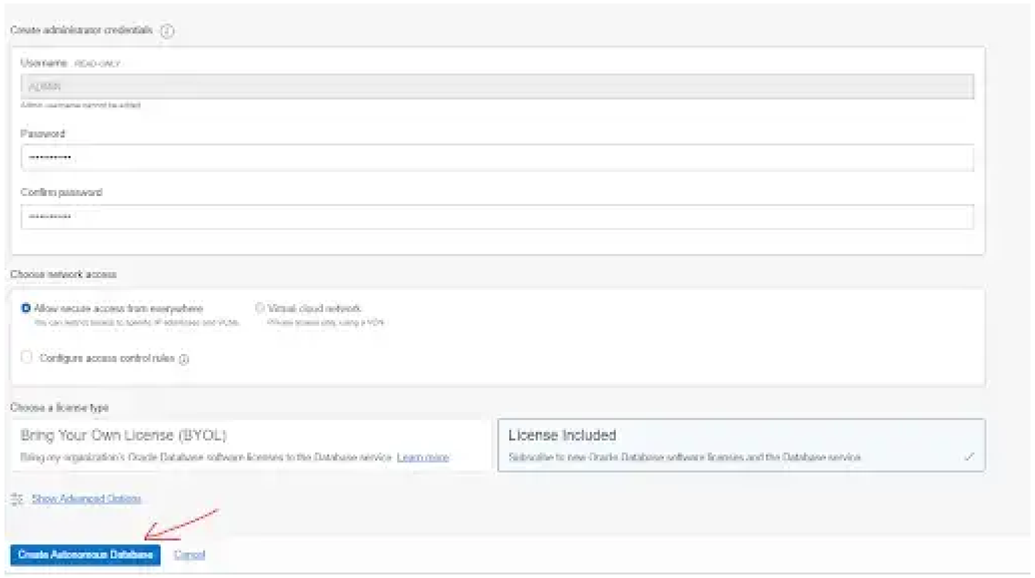
Password

•••••

Password must be 12 to 30 characters and contain at least one uppercase letter, one lowercase letter, and one number. The password cannot contain the double quote (") character or the username "admin".

Confirm password

After that click on the Create Autonomous Database button to order the database.



Create administrator credentials

Username READ-ONLY

ADMIN

Admin username cannot be edited

Password

Confirm password

Choose network access

[✓ Allow secure access from everywhere] You can restrict access to specific IP addresses and VCNs. Virtual cloud network Private access only, using a VCN

Configure access control rules

Choose a license type

Bring Your Own License (BYOL) Bring my organization's Oracle Database software licenses to the Database service. Learn more

[✓ License Included] Subscribe to new Oracle Database software licenses and the Database service

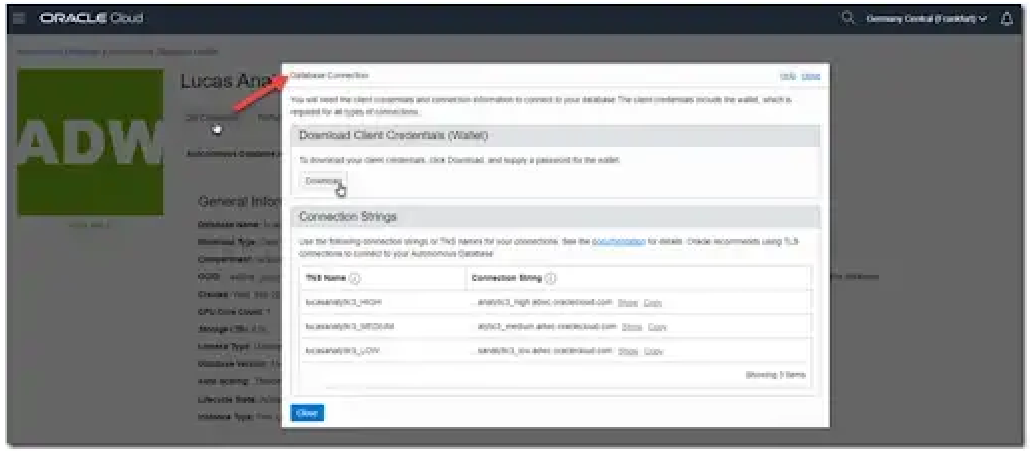
After the creation of the database you will be directed to a window where you can perform various tasks with the created Autonomous database instance.

Connecting to your database

We can use either built in Oracle Sql Developer web or a standalone Oracle Sql Developer app to connect to the database. Let’s look at how to use a standalone Oracle Sql Developer application to connect to the database from a remote machine.

Credentials file

To use Oracle Sql Developer you need to first download the credentials.zip file from your OAD. This file is an archive that holds the information of the connection and its properties for the client application to use when connecting to the database remotely. To download that click on the DB connection button on the OAD dashboard and in the dialog you get click on download. Note that you need to enter a password before downloading the file. After downloading, move the file to a location where you can access it using Oracle Sql Developer later.



Database Connection

You will need the client credentials and connection information to connect to your database. The client credentials include the wallet, which is required for all types of connections.

Download Client Credentials (Wallet)

To download your client credentials, click Download, and supply a password for the wallet.

Connection Strings

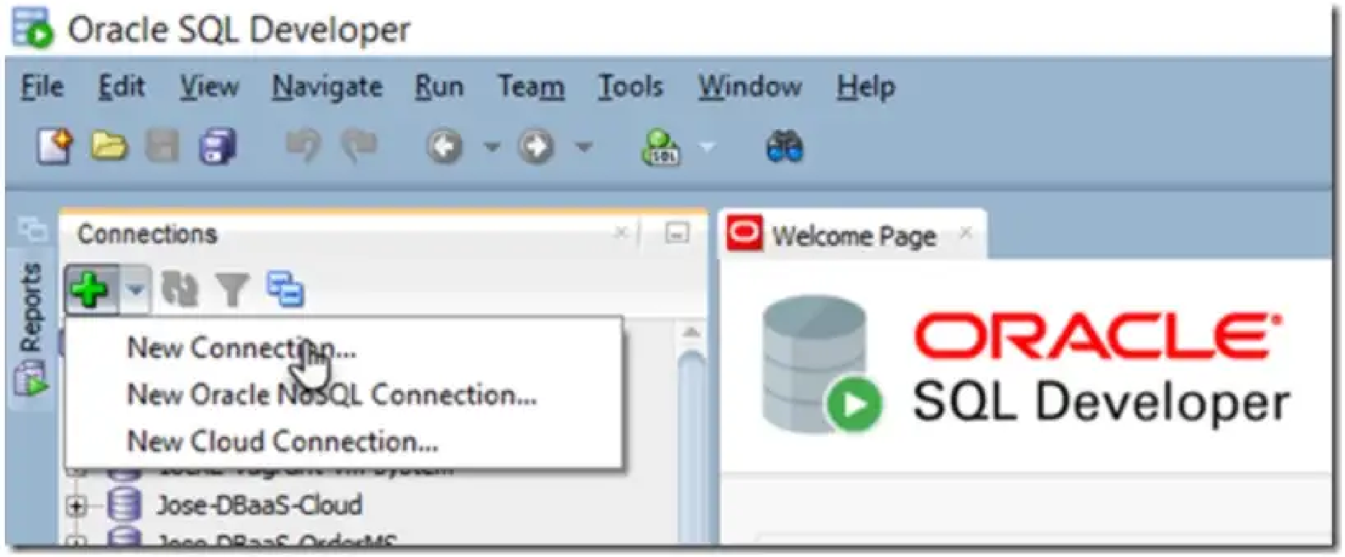
Use the following connection strings or TNS names for your connections. See the documentation for details. Oracle recommends using TLS connections to connect to your Autonomous Database.

TNS Name	Connection String
lucasanalytic3_HIGH	...analytic3_high.adwc.oraclecloud.com Show Copy
lucasanalytic3_MEDIUM	...alytic3_medium.adwc.oraclecloud.com Show Copy
lucasanalytic3_LOW	...sanalytic3_low.adwc.oraclecloud.com Show Copy

Showing 3 Items

Then we need to download the Oracle Sql Developer application from here. Its recommended to download the zip archive with the jdk so that you would not run into issues if you have not installed java in your machine. This relies on the jdk. After downloading, unzip the file and run the sqldeveloper.exe file to open the app.

In the application first create a connection to your oracle database hosted on the cloud. Click on the create new connection icon (the green + mark on the left panel)

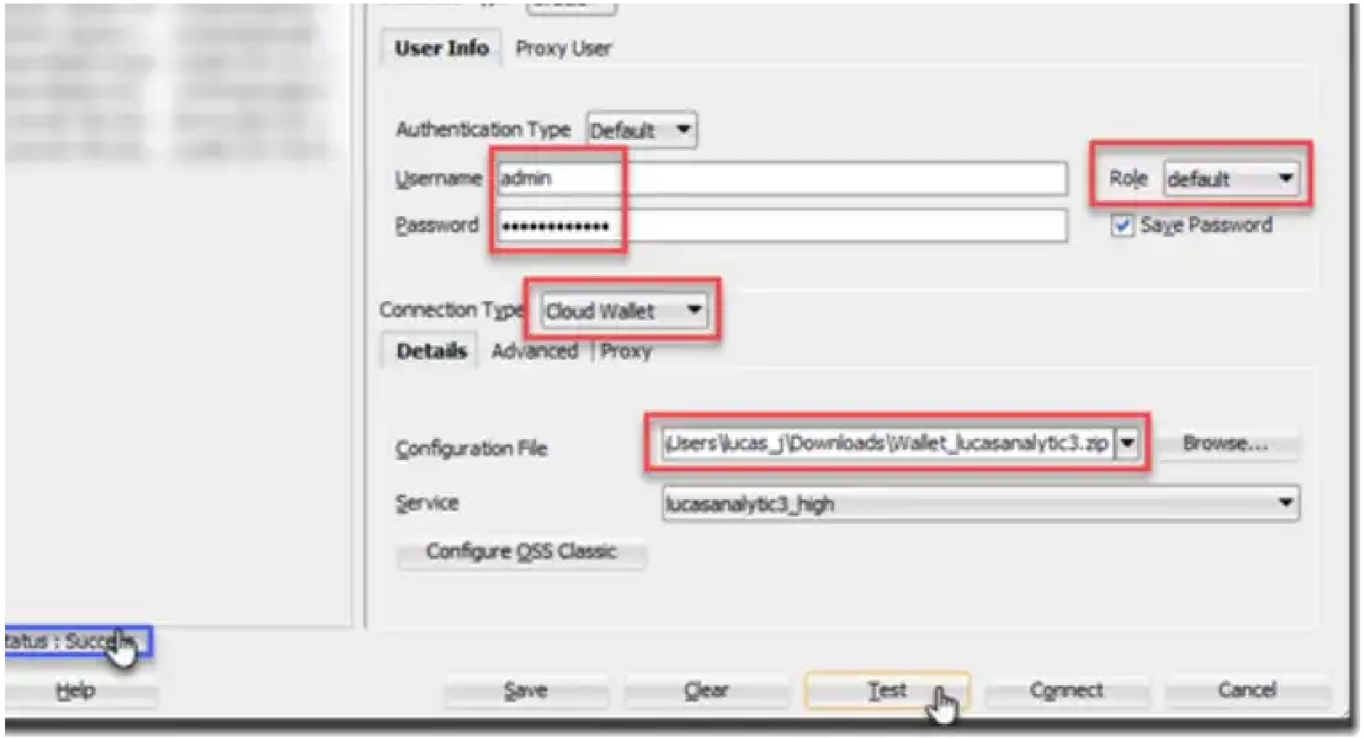


Next window enter a name for the connection and select the database type as Oracle. Keep the Role as default. Authentication type should also be default. In the username section type admin and for the password use the admin password you gave when creating the oracle autonomous database.

In the connection type section choose **cloud wallet.** Since we download the **credentials.zip** file we can use that as the cloud wallet here. For the configuration file section click on browse and locate the downloaded configuration file (credentials.zip file)

Leave other things as it is and click on the test button to see if the connection to the cloud db succeeds.

Note: You may check the save password checkbox to avoid always entering password when the connection expires and reconnecting your app to the cloud.


User Info | Proxy User

Authentication Type: Default

Username: admin

Password: •••••••••••••

Role: default

Save Password: ✓

Connection Type: Cloud Wallet

Details | Advanced | Proxy

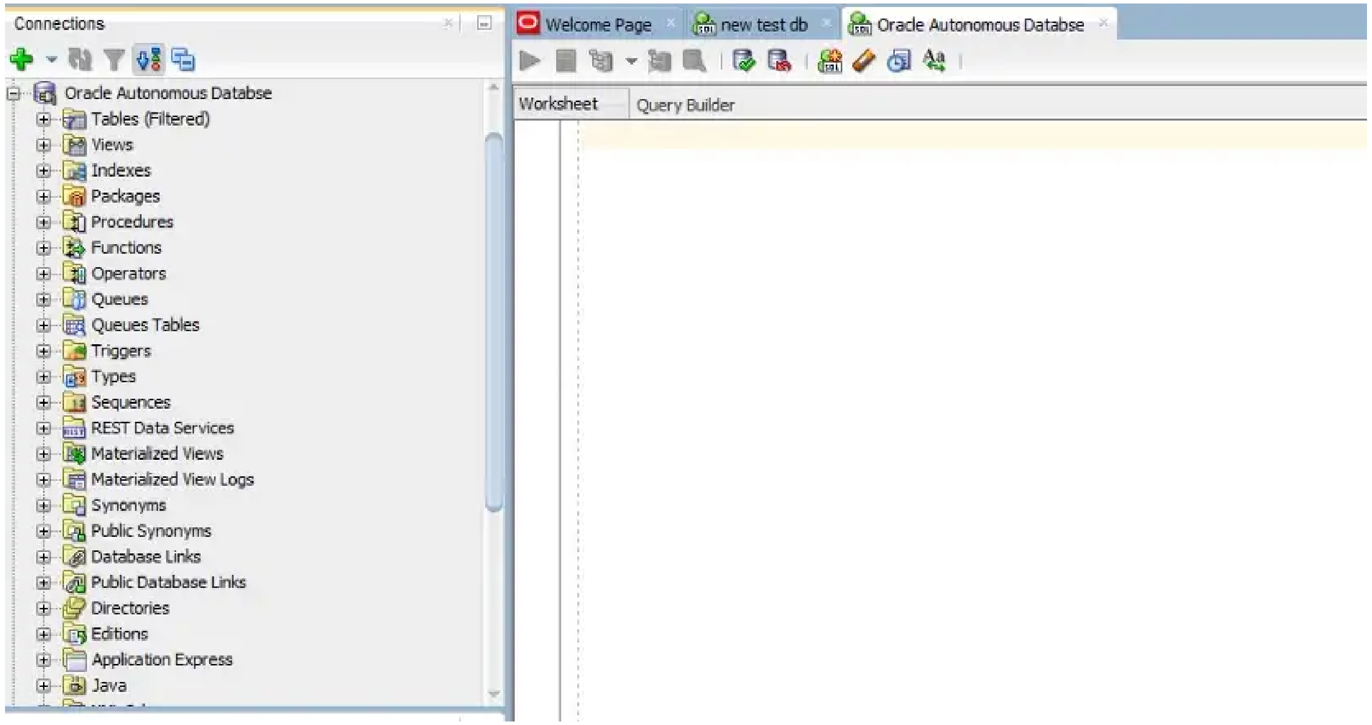
Configuration File: Users\lucas_j\Downloads\Wallet_lucasanalytic3.zip

Service: lucasanalytic3_high

Status : Success

When you see the Success status click on the connect button and you will get your database connection established in a few seconds.

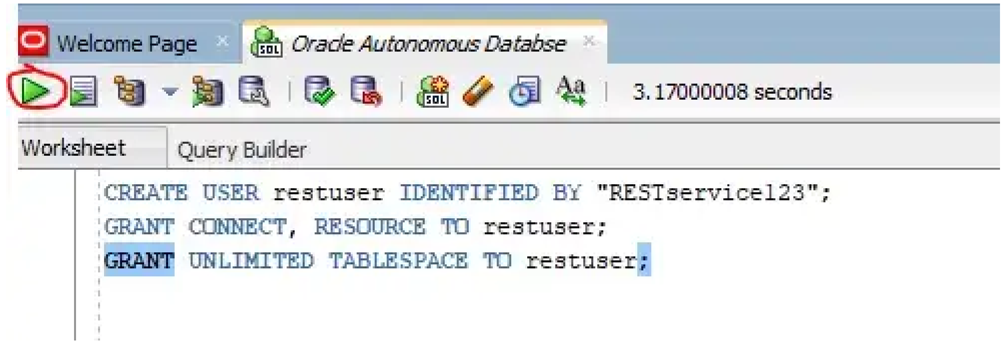
After creating the connection that will be visible on the left pane with the name you gave for the connection.



Now that we have connected to the database as the Admin user we can create other users in the database. Its not recommend use admin user's table space and privileges to create tables that are to be used with our external client applications. We create separate users with less privileges and create tables and views in those users table spaces for our client applications to use. In this way we can isolate table spaces and the applications that access those tables. Admin users normally have access to all these table spaces. Therefore for our rest data consuming applications to connect to the rest data endpoints we need to create a separate user and give that user the necessary privileges to connect and perform relevant operations in the database.

Creating a user and giving privileges for ORDS Access.

Open up a SQL worksheet on the Sql Developer app for the database. Right mouse click on the connection and choose Open Sql Worksheet to do this. In the blank worksheet you get you can type and execute sql queries and statements on the database for do DDL and DML operations. Lets run the below statement to create a user and grant the necessary privileges. After typing the queries execute those by selecting the queries and clicking on the green arrow icon as marked below.



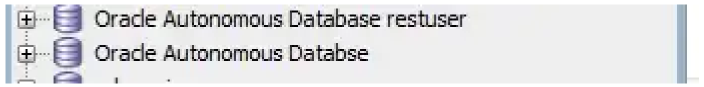
```
CREATE USER restuser IDENTIFIED BY "RESTservice123";
GRANT CONNECT, RESOURCE TO restuser;
GRANT UNLIMITED TABLESPACE TO restuser;
```

Upon successful execution you can see the result in the bottom pane: Script Output.

In the above case I am creating a user called **restuser** and my password here is **RESTservice123.**

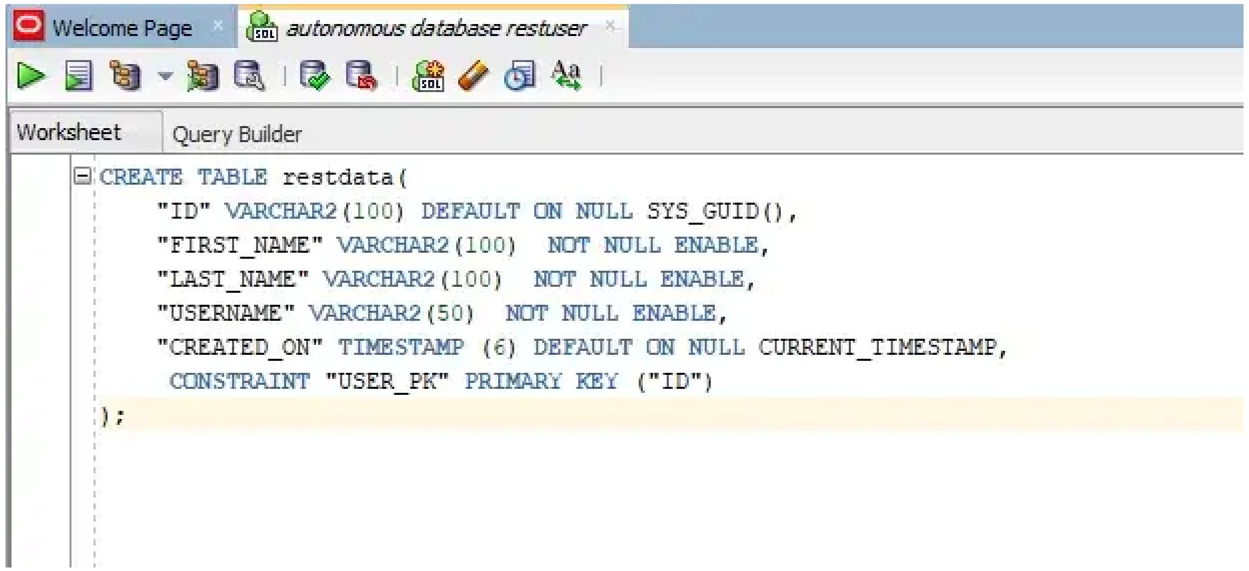
By creating the restuser and executing the GRANT TABLESPCACE statement a tables space to that user will be automatically created with the same user name and a quota is allocated to create tables in the database.

In the second statement i am **granting connect privilege to the restuser**. Why? Because when i am creating a table i am going to log in to the database again as the created user to create the tables that are used by my application in the newly created user's table space. You can stay as an admin user and still create the new tables in the newly created users table space. But for this scope we are not going to cover how to do that and to make life simple let's go and create another database connection to the new user in the Oracle Sql Developer using the same cloud wallet file. Now the two connections would look like in the below screenshot. From now onward we will be using this database connection for all our database operations.


Oracle Autonomous Database restuser
Oracle Autonomous Databse

Creating a table and enable Oracle Rest Data Services and restrict the access to outsiders

Open up a new sql worksheet on the new connection and let's create a table called restdata which we are going to use to expose the data as a rest endpoint.



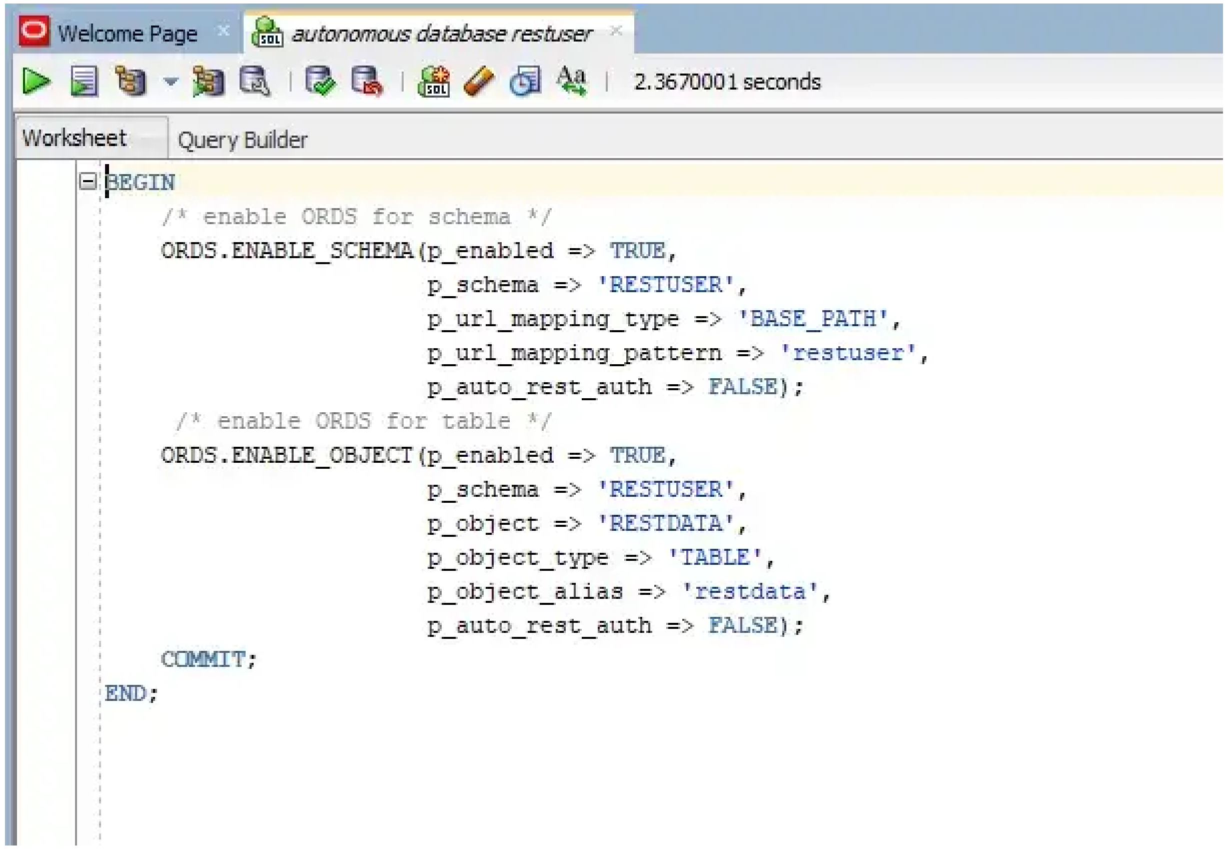
```
CREATE TABLE restdata(
    "ID" VARCHAR2(100) DEFAULT ON NULL SYS_GUID(),
    "FIRST_NAME" VARCHAR2(100)  NOT NULL ENABLE,
    "LAST_NAME" VARCHAR2(100)  NOT NULL ENABLE,
    "USERNAME" VARCHAR2(50)  NOT NULL ENABLE,
    "CREATED_ON" TIMESTAMP (6) DEFAULT ON NULL CURRENT_TIMESTAMP,
     CONSTRAINT "USER_PK" PRIMARY KEY ("ID")
);
```

Above we created the table called restdata by logging in as the new user we created (restuser)

By doing that we created that table in the new users table space. In simple terms a table space is the allocated quota to a user in a database to create tables. We have granted unlimited tablespace for the restuser when we were creating the user in the beginning executing the below statement.

GRANT UNLIMITED TABLESPACE TO username ;

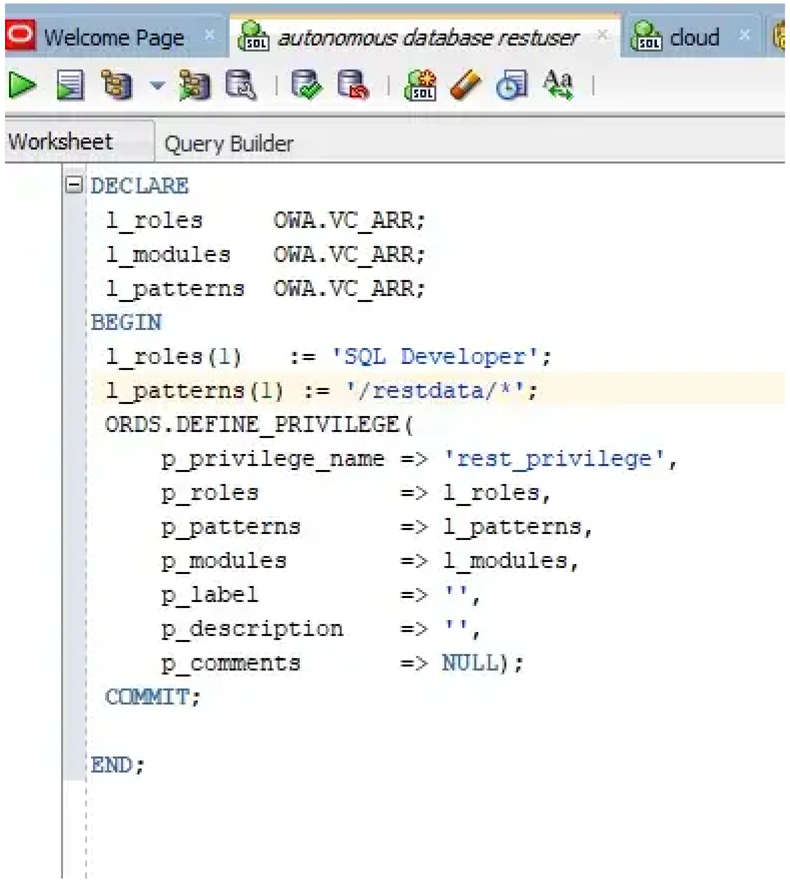
Okay. Now we need to tell Oracle that this created table and the users namespace should be enabled to work with the rest services. For that we need to **restenable** the table and the namespace. Run the following statement for that.



```
BEGIN
    /* enable ORDS for schema */
    ORDS.ENABLE_SCHEMA(p_enabled => TRUE,
                       p_schema => 'RESTUSER',
                       p_url_mapping_type => 'BASE_PATH',
                       p_url_mapping_pattern => 'restuser',
                       p_auto_rest_auth => FALSE);
     /* enable ORDS for table */
    ORDS.ENABLE_OBJECT(p_enabled => TRUE,
                       p_schema => 'RESTUSER',
                       p_object => 'RESTDATA',
                       p_object_type => 'TABLE',
                       p_object_alias => 'restdata',
                       p_auto_rest_auth => FALSE);
    COMMIT;
END;
```

We are setting **p_auto_rest_auth** to FALSE in each of the two statements above. This is done to make sure that an unauthenticated client calls will not be answered by our endpoint. E 4 ıser needs to send a rest call the the endpoint, user needs to send credentials and authenticate the rest call before performing any operation with the database. Otherwise the server will responded with 401 Unauthenticated error.

Although above statements have restenabled the object and schema but we need to create a privilege to the table before we can do the rest calls.Execute the below statement to register that privilege for the table.

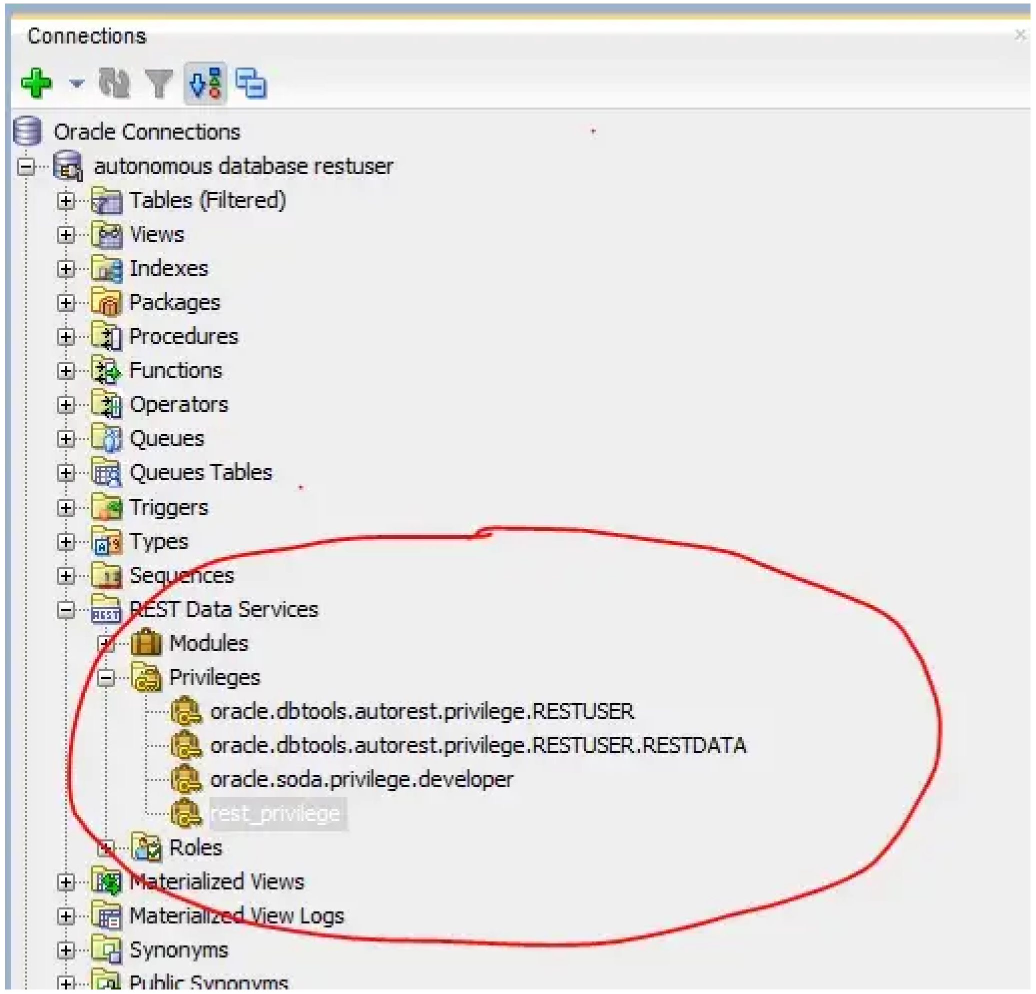


```
DECLARE
 l_roles     OWA.VC_ARR;
 l_modules   OWA.VC_ARR;
 l_patterns  OWA.VC_ARR;
BEGIN
 l_roles(1)   := 'SQL Developer';
 l_patterns(1) := '/restdata/*';
 ORDS.DEFINE_PRIVILEGE(
     p_privilege_name => 'rest_privilege',
     p_roles          => l_roles,
     p_patterns       => l_patterns,
     p_modules        => l_modules,
     p_label          => '',
     p_description    => '',
     p_comments       => NULL);
 COMMIT;

END;
```

Replace the value for the l_patterns(1) with the table name like /restdata/* as above to be able for the rest privilege to identify the pattern when we are calling using our url. Later we will see that our rest api call will have the table name in the url and we are doing this here for the endpoint to understand if this table name comes in the url you have a request that makes sense and you can serve that request with this table's data. Don't worry you will get to know about what this url and the format of the url when we talk about doing the rest call.

Now if you go and check the left panel and expand the Rest services section and expand privileges you should be able to see something like below. If you see the same you can assume that everything so far is done correctly.



Now we have all the privileges and restrictions set for the table and the table space (user).

We are done with server side configurations now and the rest is related to the Authentication and client registration. In my next article we will lean about what is Authentication and how we are going to use it to send an authenticated REST call to this configured OAD.

Ords Oracle Autonomous Db Rest Api Oauth Oracel Sql Developer