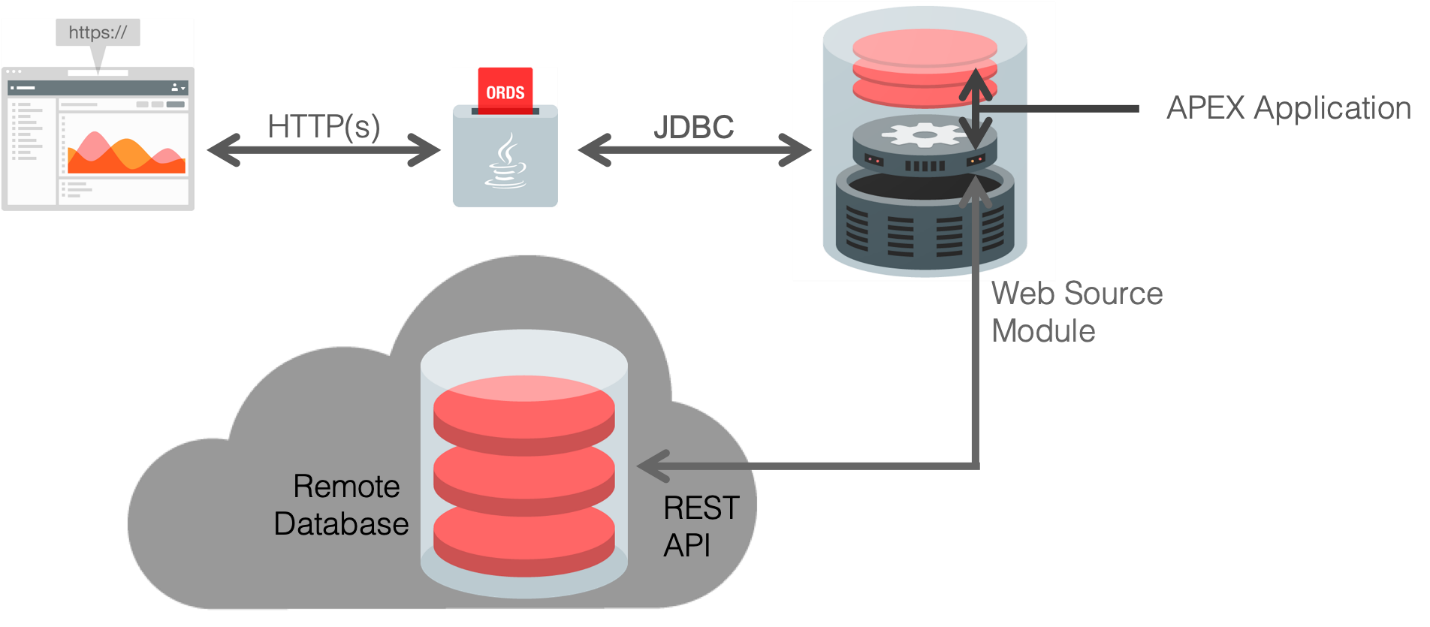
**REST Enabling Database Objects**

**Introduction**

These notes are about how to REST enable a schema and database objects.

**We** will be creating the regular EMP and DEPT tables used in Oracle, REST enabling them, and then building an application all in the same schema. Normally, you would REST enable database objects in a remote database schema, and then utilize those REST endpoints to then define Web Source modules in an application, where APEX is installed, within the local database, as outlined below.

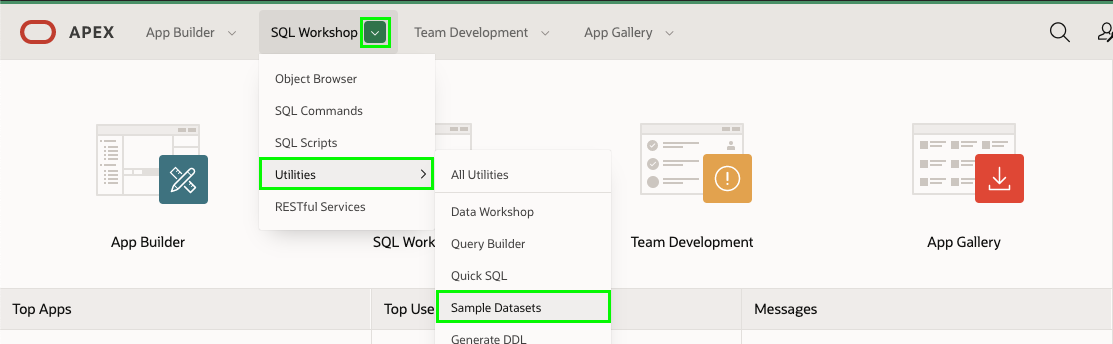


Once the Web Source module is defined in an APEX application, rather than calling a local database object, the APEX Engine calls the REST endpoint, to retrieve the data from the remote database. If the PUT, POST, and DELETE handlers have also been defined, then APEX applications can also perform updates, inserts, and deletes on the remote database objects.

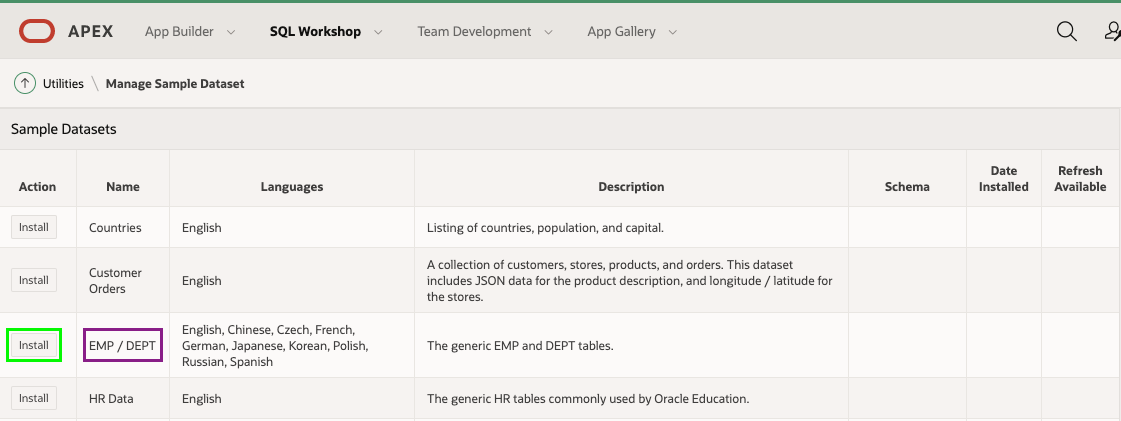
**STEP 1 – Creating the Sample Tables**

This lab utilizes the common EMP and DEPT tables. These can easily be installed using Sample Datasets.

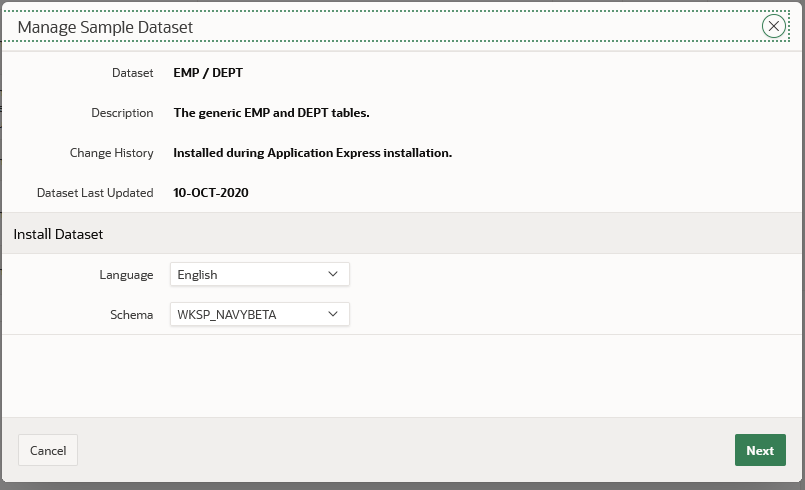
1. Log into your workspace.
2. In the main menu, select **SQL Workshop**, select **Utilities**, click **Sample Datasets**.



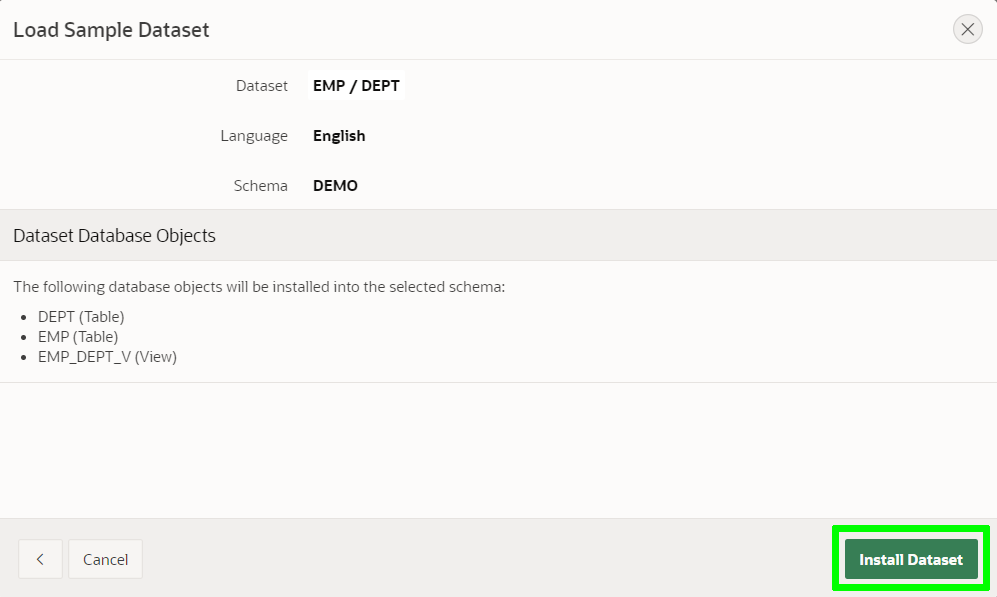
1. In the list of Sample Datasets, on the EMP / DEPT row, click **Install**.

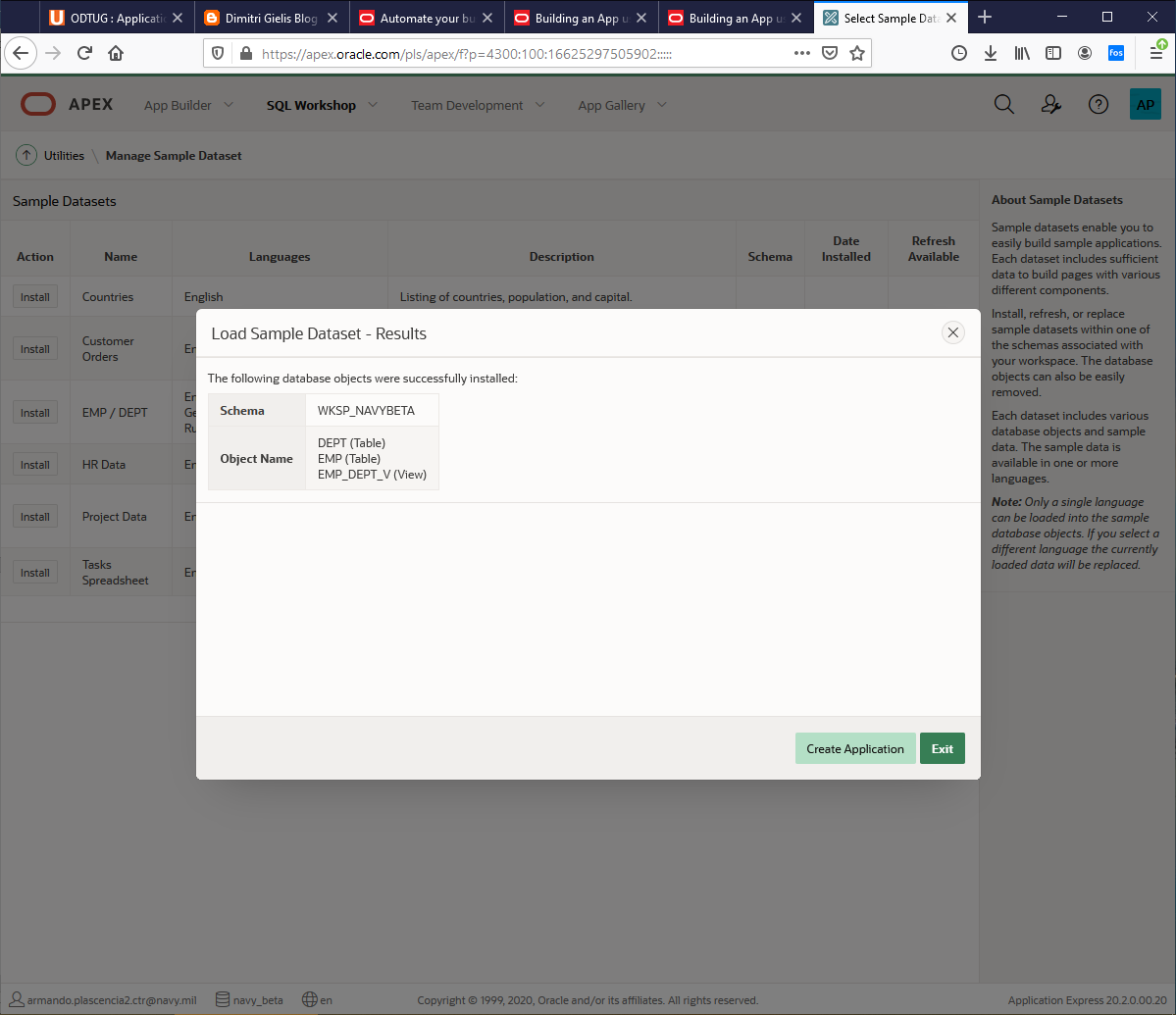


1. On the Manage Sample Dataset dialog, click **Next**.



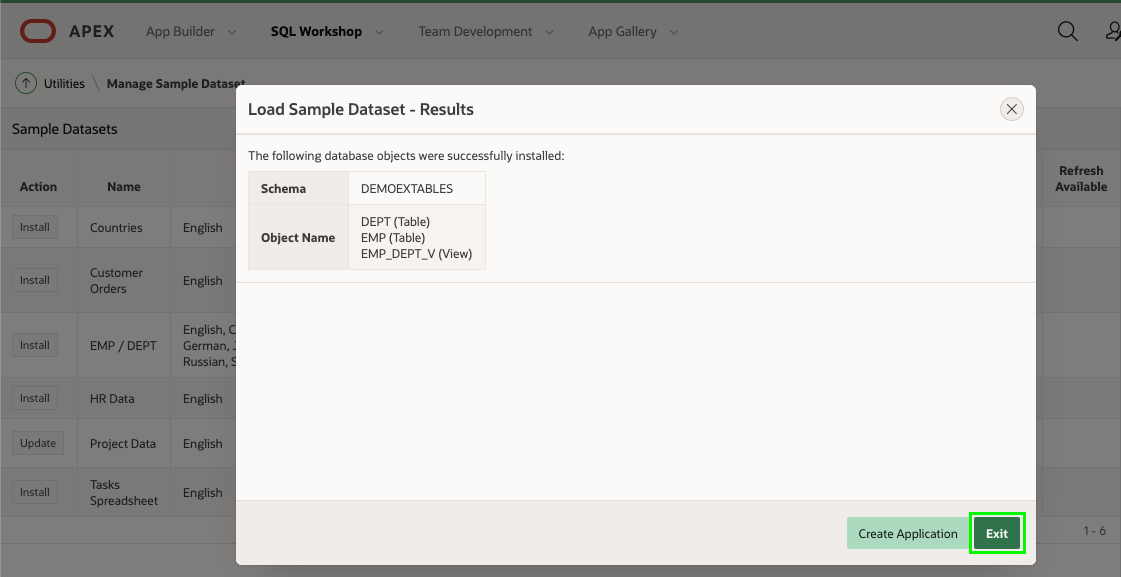
1. On the Load Sample Dataset dialog, click **Install Dataset**.





1. On the Load Sample Dataset - Results dialog, click **Exit**.

*Note: We do not want to create an application directly on the tables.*

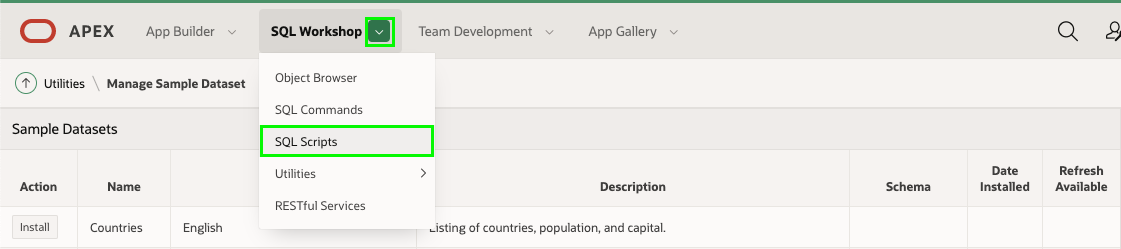


**STEP 2 – REST Enabling the Database Objects**

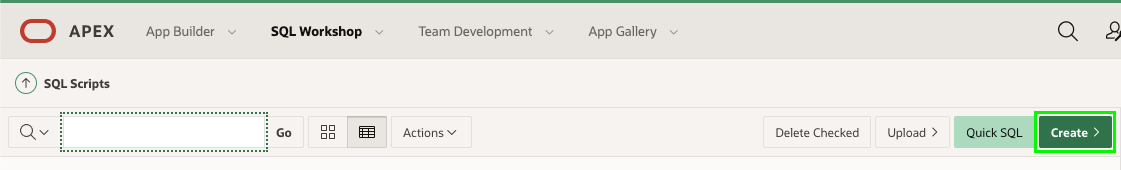
Running the SQL Script below will REST Enable the schema and also create modules for the EMP and DEPT tables and the EMP\_DEPT\_V view, together with the appropriate handlers.

Alternatively, you could go to SQL Workshop > RESTful Services and REST enabled the schema. Then go to SQL Workshop > Object Browser, clicked on each table/view and then selected REST to define the REST services. However, the handlers created using the SQL Workshop > Object Browser will utilize Data Dictionary lookups each time they are called. This is less efficient then the manually created services you will create, especially on services, such as apex.oracle.com, which have an extremely large data dictionary. On apex.oracle.com there are over 20,000 schemas.

1. In the main menu, select **SQL Workshop**, click **SQL Scripts**.



1. On the SQL Scripts page, click **Create**.



1. In the Script Editor, enter the following.
   * **Script Name:** Enter **Define REST Endpoints for EMP and DEPT**.
   * **Script Editor:** Cut and paste the following.

begin

ords.enable\_schema;

end;

/

begin

ords.delete\_module(

p\_module\_name => 'emp.rest' );

ords.define\_module(

p\_module\_name => 'emp.rest',

p\_base\_path => '/emp/' );

ords.define\_template(

p\_module\_name => 'emp.rest',

p\_pattern => 'hol/' );

ords.define\_template(

p\_module\_name => 'emp.rest',

p\_pattern => 'hol/:empno' );

ords.define\_handler(

p\_module\_name => 'emp.rest',

p\_pattern => 'hol/',

p\_method => 'GET',

p\_source\_type => ords.source\_type\_collection\_feed,

p\_source => 'select \* from emp' );

ords.define\_handler(

p\_module\_name => 'emp.rest',

p\_pattern => 'hol/:empno',

p\_method => 'GET',

p\_source\_type => ords.source\_type\_collection\_item,

p\_source => 'select \* from emp where empno = :empno' );

ords.define\_handler(

p\_module\_name => 'emp.rest',

p\_pattern => 'hol/',

p\_method => 'POST',

p\_source\_type => ords.source\_type\_plsql,

p\_source => 'begin insert into emp (empno, ename, job, mgr, hiredate, sal, comm, deptno) values (:empno,:ename,:job,:mgr,:hiredate,:sal,:comm,:deptno);:forward\_location:=:empno;:status\_code:=201;end;' );

ords.define\_handler(

p\_module\_name => 'emp.rest',

p\_pattern => 'hol/:empno',

p\_method => 'PUT',

p\_source\_type => ords.source\_type\_plsql,

p\_source => 'begin update emp set ename=:ename,job=:job,mgr=:mgr,hiredate=:hiredate,sal=:sal,comm=:comm,deptno=:deptno where empno=:empno; :forward\_location:=:empno;:status\_code:=200;end;' );

ords.define\_handler(

p\_module\_name => 'emp.rest',

p\_pattern => 'hol/:empno',

p\_method => 'DELETE',

p\_source\_type => ords.source\_type\_plsql,

p\_source => 'begin delete emp where empno=:empno;:status\_code:=200;end;' );

end;

/

begin

ords.delete\_module(

p\_module\_name => 'dept.rest' );

ords.define\_module(

p\_module\_name => 'dept.rest',

p\_base\_path => '/dept/' );

ords.define\_template(

p\_module\_name => 'dept.rest',

p\_pattern => 'hol/' );

ords.define\_template(

p\_module\_name => 'dept.rest',

p\_pattern => 'hol/:deptno' );

ords.define\_handler(

p\_module\_name => 'dept.rest',

p\_pattern => 'hol/',

p\_method => 'GET',

p\_source\_type => ords.source\_type\_collection\_feed,

p\_source => 'select \* from dept' );

ords.define\_handler(

p\_module\_name => 'dept.rest',

p\_pattern => 'hol/:deptno',

p\_method => 'GET',

p\_source\_type => ords.source\_type\_collection\_item,

p\_source => 'select \* from dept where deptno = :deptno' );

ords.define\_handler(

p\_module\_name => 'dept.rest',

p\_pattern => 'hol/',

p\_method => 'POST',

p\_source\_type => ords.source\_type\_plsql,

p\_source => 'begin insert into dept (deptno, dname, loc) values (:deptno,:dname,:loc);:forward\_location:=:deptno;:status\_code:=201;end;' );

ords.define\_handler(

p\_module\_name => 'dept.rest',

p\_pattern => 'hol/:deptno',

p\_method => 'PUT',

p\_source\_type => ords.source\_type\_plsql,

p\_source => 'begin update dept set dname=:dname,loc=:loc where deptno=:deptno; :forward\_location:=:deptno;:status\_code:=200;end;' );

ords.define\_handler(

p\_module\_name => 'dept.rest',

p\_pattern => 'hol/:deptno',

p\_method => 'DELETE',

p\_source\_type => ords.source\_type\_plsql,

p\_source => 'begin delete dept where deptno=:deptno;:status\_code:=200;end;' );

end;

/

begin

ords.delete\_module(

p\_module\_name => 'empdeptv.rest' );

ords.define\_module(

p\_module\_name => 'empdeptv.rest',

p\_base\_path => '/empdeptv/' );

ords.define\_template(

p\_module\_name => 'empdeptv.rest',

p\_pattern => 'hol/' );

ords.define\_template(

p\_module\_name => 'empdeptv.rest',

p\_pattern => 'hol/:empno' );

ords.define\_handler(

p\_module\_name => 'empdeptv.rest',

p\_pattern => 'hol/',

p\_method => 'GET',

p\_source\_type => ords.source\_type\_collection\_feed,

p\_source => 'select \* from emp\_dept\_v' );

ords.define\_handler(

p\_module\_name => 'empdeptv.rest',

p\_pattern => 'hol/:empno',

p\_method => 'GET',

p\_source\_type => ords.source\_type\_collection\_item,

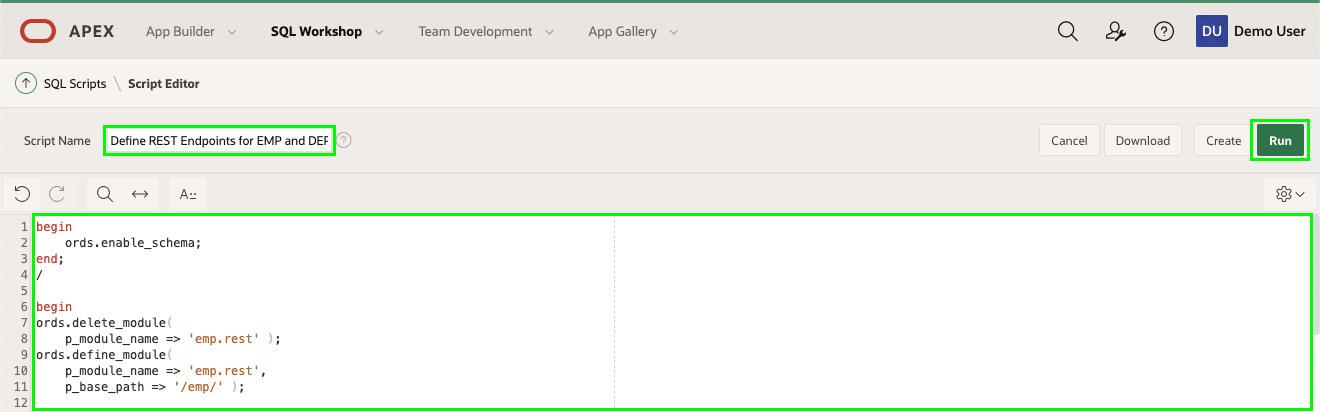
p\_source => 'select \* from emp\_dept\_v where empno = :empno' );

end;

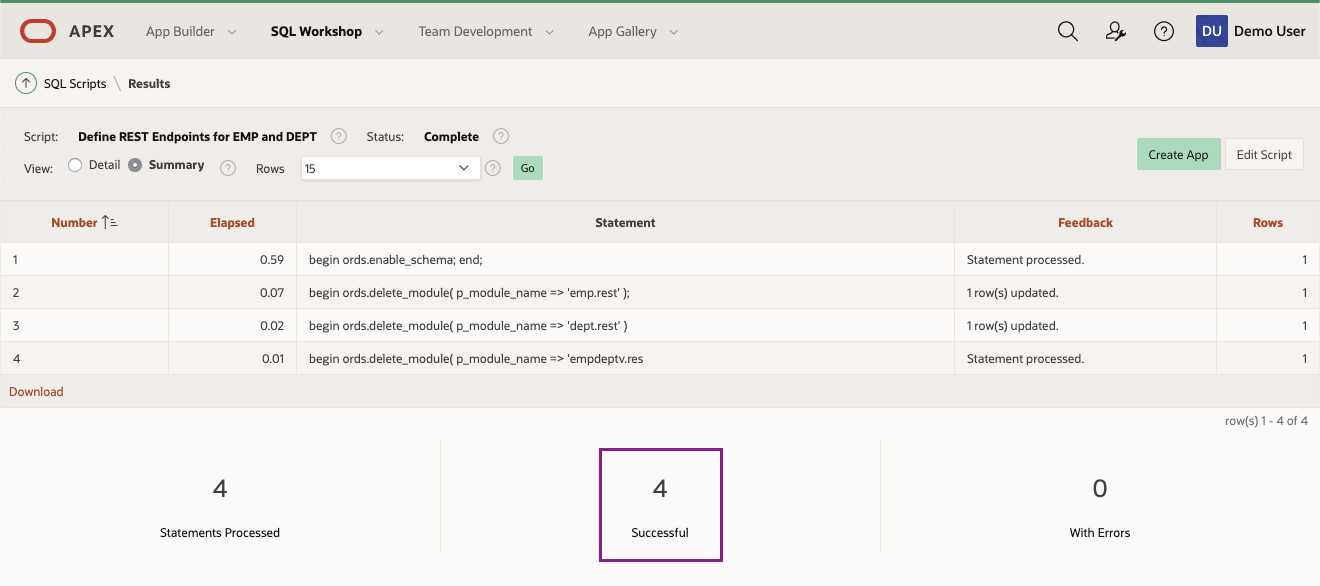
/

1. *Note: You can also cut and paste the script from the following URL -* <https://www.oracle.com/technetwork/developer-tools/apex/application-express/apex-hol-rest-enable-5478504.txt>

Click **Run**.

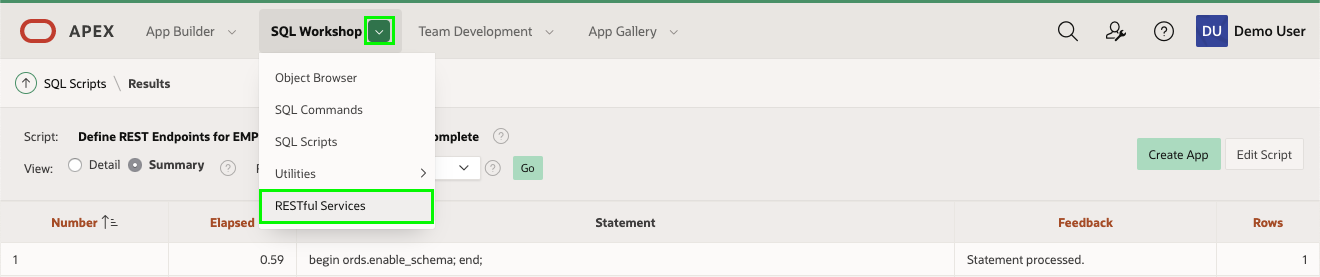


1. On the Run Script page, click **Run Now**.
2. Results should show 4 statements processed successfully.



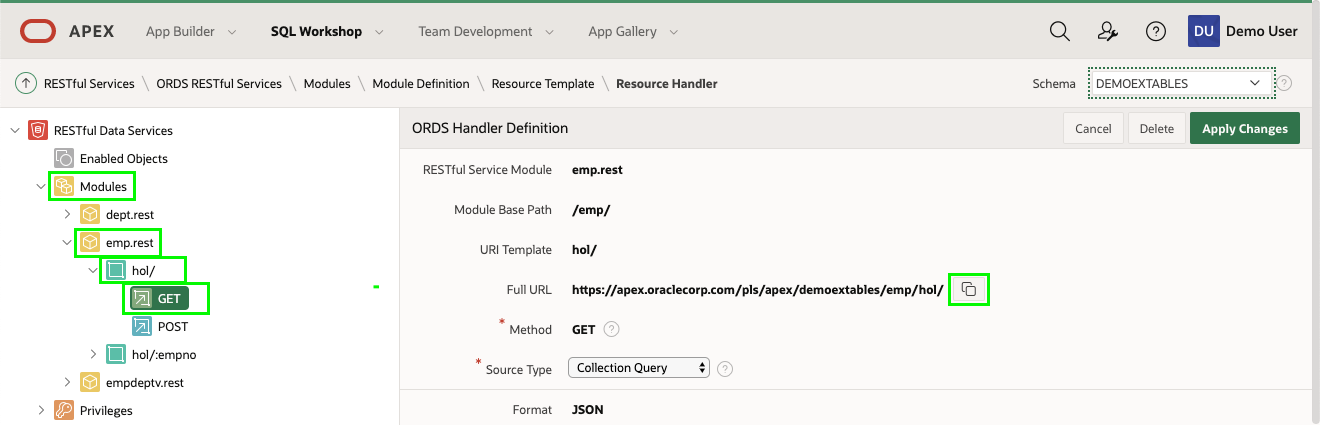
**STEP 3 – Reviewing the REST Services**

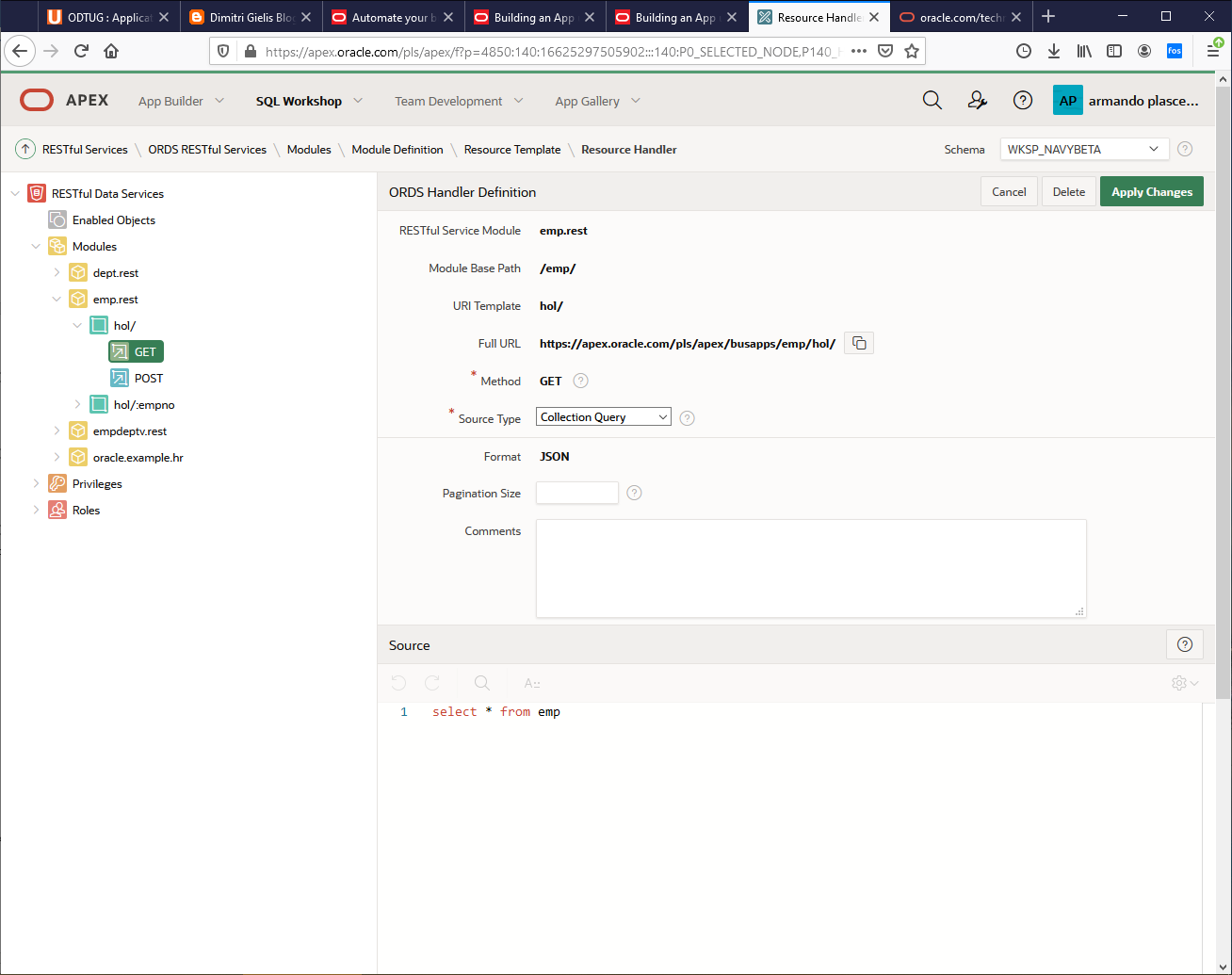
1. In the main menu, select **SQL Workshop**, click **RESTful Services**.



1. In the ORDS RESTful Services page, expand **Modules**, expand **emp.rest**, and then expand **hol/**.  
   Click on **GET**.

Click on the Copy to Clipboard icon (or manually copy the Full URL).





1. Open a new tab/window in your browser and paste the Full URL. Keep this tab open or make a note of this URL because you will need it later.



In our case <https://apex.oracle.com/pls/apex/busapps/emp/hol/>

*Note: Depending on your browser you may just see the raw JSON document rather than the formatted output displayed above.*

**Summary**

This completes Lab 2. You now know how to REST enable database objects include all of the necessary handlers. [Click here to navigate to Lab 3](https://oracle.github.io/learning-library/developer-library/apex/rest-web-source/?lab=lab-3-defining-web-sources).

**Acknowledgements**

* **Author/Contributors** - Salim Hlayel, Principle Product Manager
* **Contributors** - Oracle LiveLabs Team (Arabella Yao, Product Manager Intern | Jaden McElvey, Technical Lead | Jeffrey Malcolm Jr, Intern)
* **Last Updated By/Date** - Tom McGinn, Database Innovation Architect, Product Management, July 2020

**Need Help?**

Please submit feedback or ask for help using our [LiveLabs Support Forum](https://community.oracle.com/tech/developers/categories/oracle-apex-development-workshops). Please click the **Log In** button and login using your Oracle Account. Click the **Ask A Question** button to the left to start a *New Discussion* or *Ask a Question*. Please include your workshop name and lab name. You can also include screenshots and attach files. Engage directly with the author of the workshop.

If you do not have an Oracle Account, click [here](https://profile.oracle.com/myprofile/account/create-account.jspx) to create one.