Practice: Creating and Running Notebooks in Oracle Machine Learning

Get your free cloud account: click here.

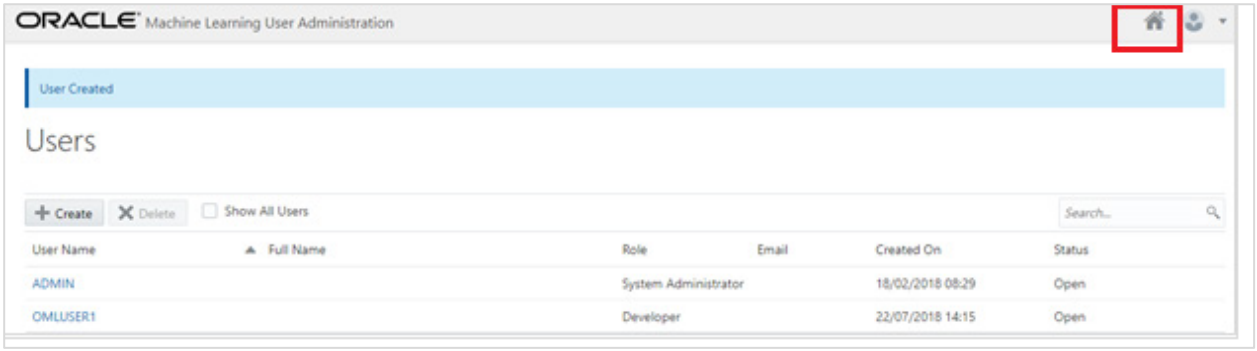
Overview

In this practice, you create a new Oracle Machine Learning notebook application provided with ADW.

Tasks

1. Log into your Oracle Cloud Free Tier Account
   a. Navigate to the **Oracle Autonomous Databases** home page where your instance is listed.
   b. Click your Data Warehouse instance name.
   c. Click the **Service Console** option 

.
   d. If required, log in as the **admin** user. Here, you are signing in to the Oracle Autonomous Data warehouse console.
   e. On the left hand side menu, click **Administration**.
   f. Click **Manage Oracle ML Users**.
   g. If required, log in as the **admin** user. Here, you are signing in to the Oracle Machine Learning console.
   h. Click **Home** in the top-right corner, highlighted in the following screenshot:

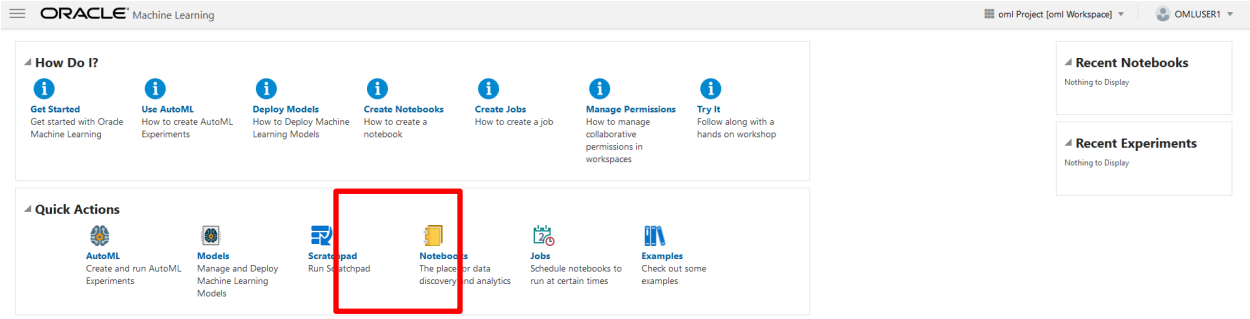


   i. Log in as the **OMLUSER1** user. This time, you are signing in to the Oracle Machine Learning console as the **OMLUSER1** user.

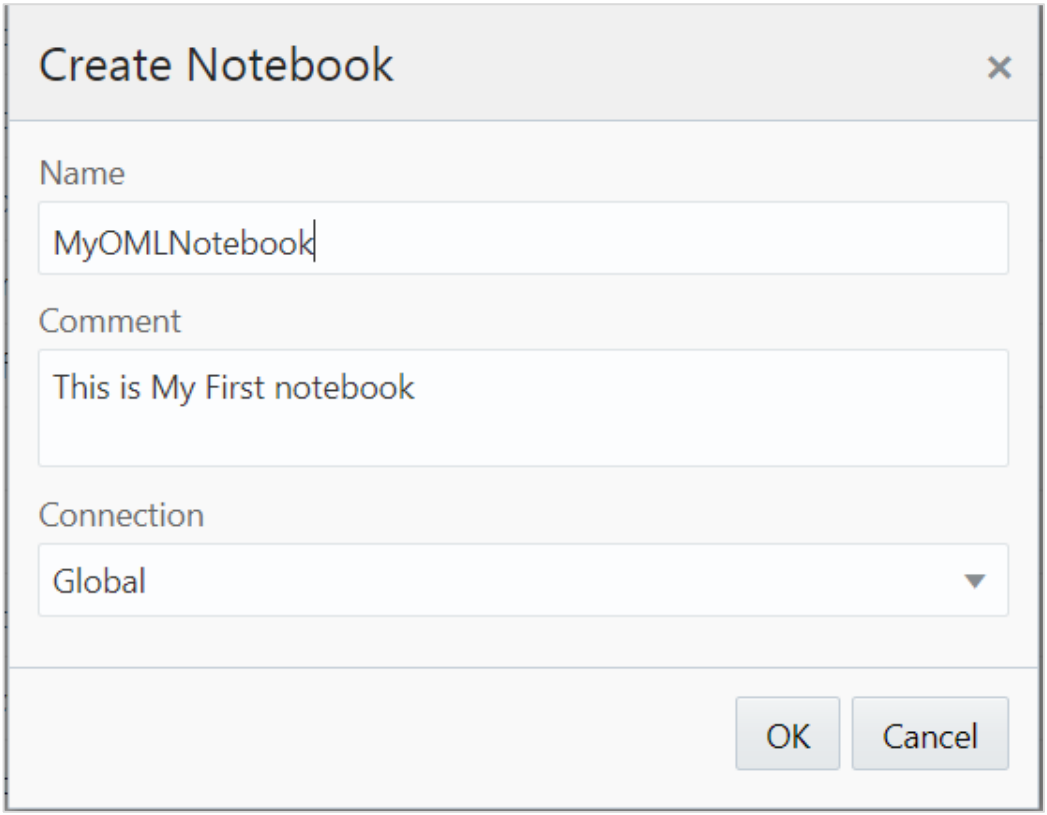
      After you have successfully signed in to OML, the application home page will be displayed.

j. Ensure you are in the **MyFirstproject[OMLUSER1SAMPLE]** project and workspace, which you created in the previous practice.

2. On the Oracle Machine Learning home page, click **Notebooks**.



3. In the **Notebooks** action item, click **Create**.
4. In the **Create Notebook** window, enter **Name**, **Comment,** and **Global** as a connection string. After you click the **OK** button, you will get your notebook ready, as in the following screenshot. In this case, MyOMLNotebook is the notebook name.



5. In the newly created notebook, you can enter SQL commands and run them.

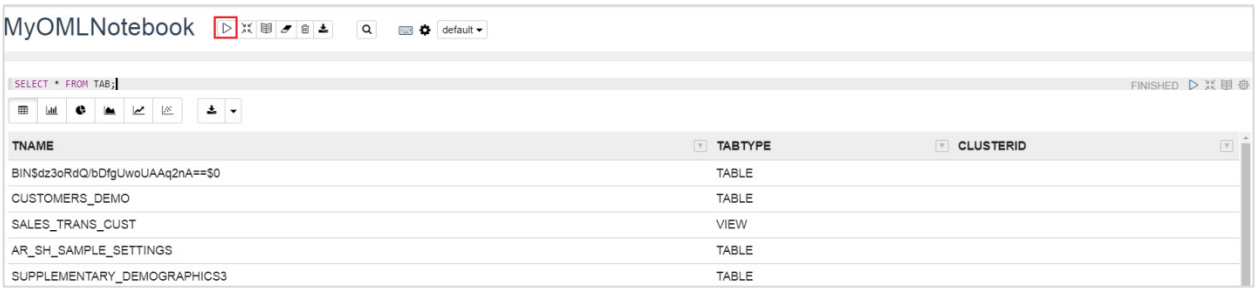
   a. In the notebook editor type, use the following SQL statements to fetch data from an Oracle Database:

```
%sql
select * from tab;
```

   b. We are using the table name **`tab`** in this example as we do not have any tables yet in this database. Ensure you have **`%sql`** as the first line of the editor. Click **Run** next to the **READY** sign when you are ready.

6. You can run a notebook by clicking the **Run** icon next to the **READY** sign. This is a single paragraph run. Alternatively, you can press **Shift+Enter**.

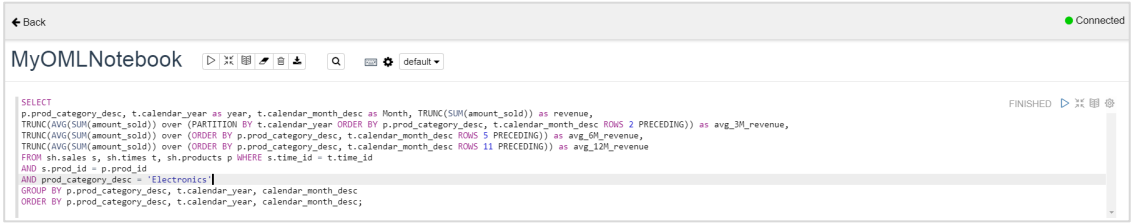
   For a multiscript or multiparagraph run, which is also called an all paragraph run, click **Run** next to the notebook name.



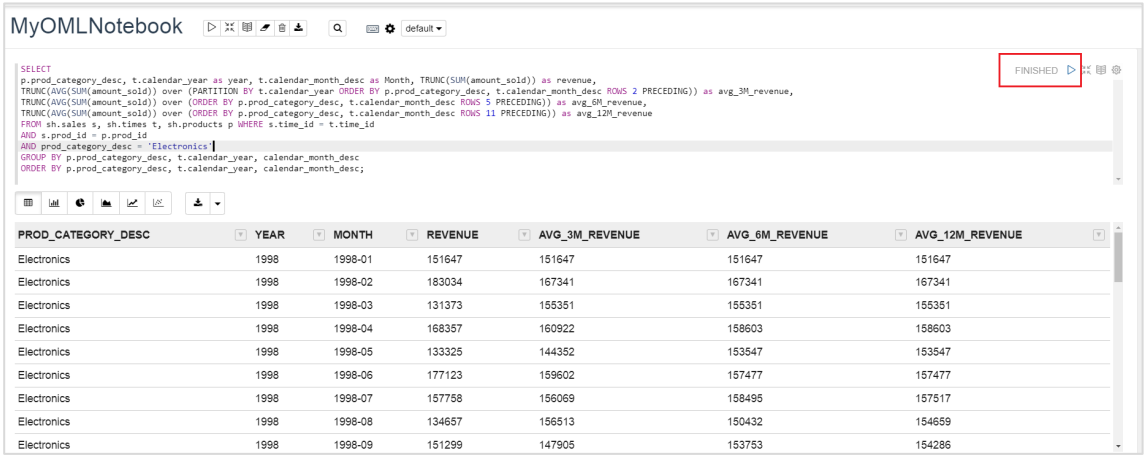
a. In the **MyOMLNotebook** area, copy and paste the following code:

```
SELECT
 p.prod_category_desc, t.calendar_year as
 year, t.calendar_month_desc as Month,
 TRUNC(SUM(amount_sold)) as revenue,
 TRUNC(AVG(SUM(amount_sold)) over (PARTITION BY t.calendar_year ORDER BY
p.prod_category_desc, t.calendar_month_desc ROWS 2 PRECEDING)) as
avg_3M_revenue,
 TRUNC(AVG(SUM(amount_sold)) over (ORDER BY
p.prod_category_desc, t.calendar_month_desc ROWS 5 PRECEDING))
as avg_6M_revenue,
 TRUNC(AVG(SUM(amount_sold)) over (ORDER BY
p.prod_category_desc, t.calendar_month_desc ROWS 11 PRECEDING))
as avg_12M_revenue
FROM sh.sales s, sh.times t, sh.products p WHERE
s.time_id = t.time_id
AND s.prod_id = p.prod_id
AND prod_category_desc = 'Electronics'
GROUP BY p.prod_category_desc, t.calendar_year,
calendar_month_desc
ORDER BY p.prod_category_desc, t.calendar_year,
calendar_month_desc;
```

b. Your screen should now look like this:



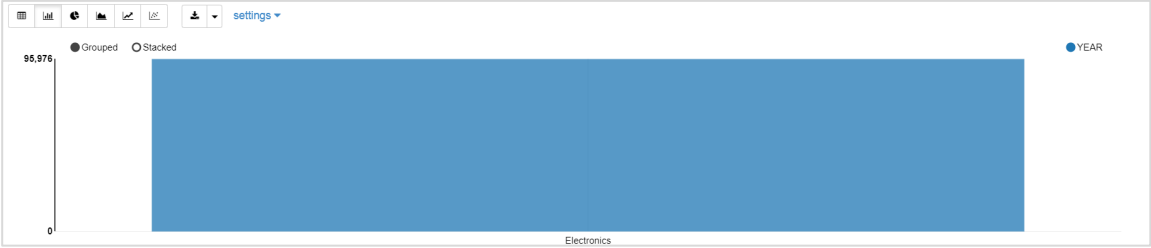
c. Click the **Run this paragraph** icon shown in the following screenshot to execute the SQL statement and display the results in a tabular format.



PROD_CATEGORY_DESC	YEAR	MONTH	REVENUE	AVG_3M_REVENUE	AVG_6M_REVENUE	AVG_12M_REVENUE
Electronics	1998	1998-01	151647	151647	151647	151647
Electronics	1998	1998-02	183034	167341	167341	167341
Electronics	1998	1998-03	131373	155351	155351	155351
Electronics	1998	1998-04	168357	160922	158603	158603
Electronics	1998	1998-05	133325	144352	153547	153547
Electronics	1998	1998-06	177123	159602	157477	157477
Electronics	1998	1998-07	157758	156069	158495	157517
Electronics	1998	1998-08	134657	156513	150432	154659
Electronics	1998	1998-09	151299	147905	153753	154286

7. Changing the report type
   a. Using the **report menu bar**, you can change the table to a graph and/or export the result set to a CSV or TSV file.

   b. Click the **bar graph** icon to change the output to a bar graph.

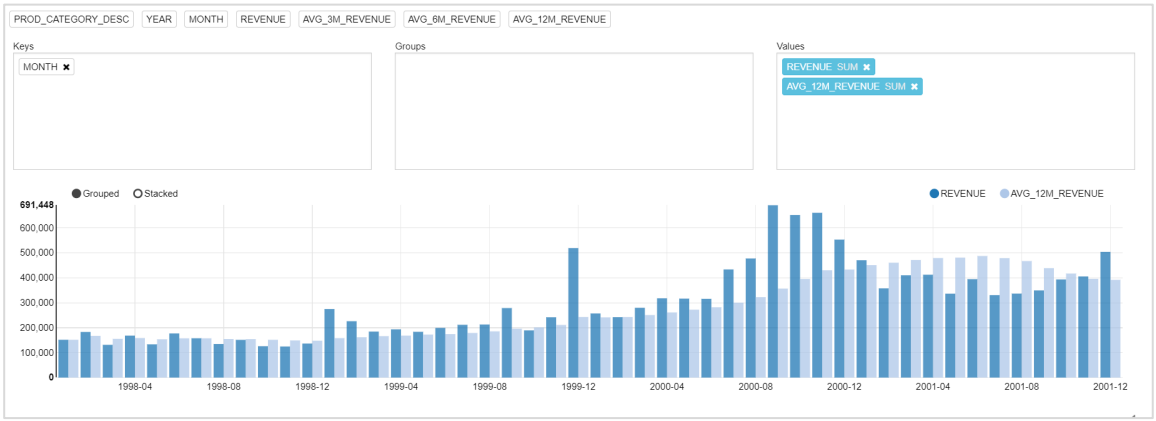


   c. Click the **Settings** link to unfold the settings panel for the graph.

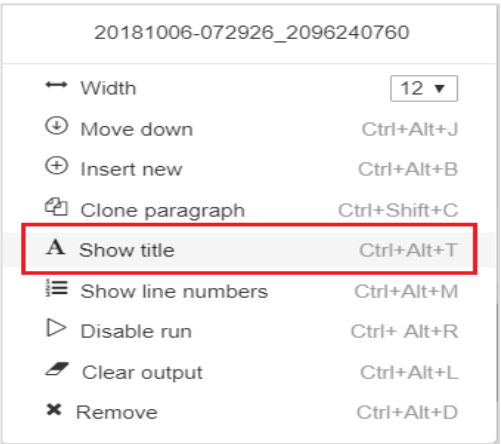
   Note: To add a column to one of the **Keys**, **Groups,** or **Values** panels, just drag and drop the column name into the required panel. To remove a column from the Keys, Groups, or Values panel, just click the **x** next to the column name displayed in the relevant panel.

8. Changing the layout of the graph
   a. With the graph settings panel visible, remove all columns from both the Keys and Values panels.
   b. Drag and drop MONTH into the Keys panel.
   c. Drag and drop REVENUE into the Values panel.
   d. Drag and drop AVG_12M_REVENUE into the Values panel.

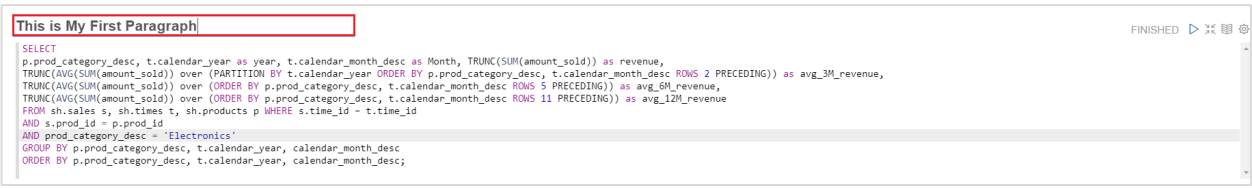
e. The report should now look like the one below.



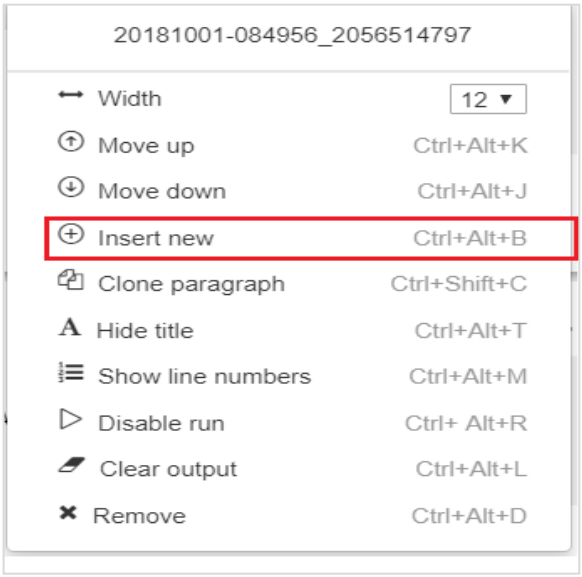
9. To show the title for the paragraph, click in the right corner and select **Show title**.

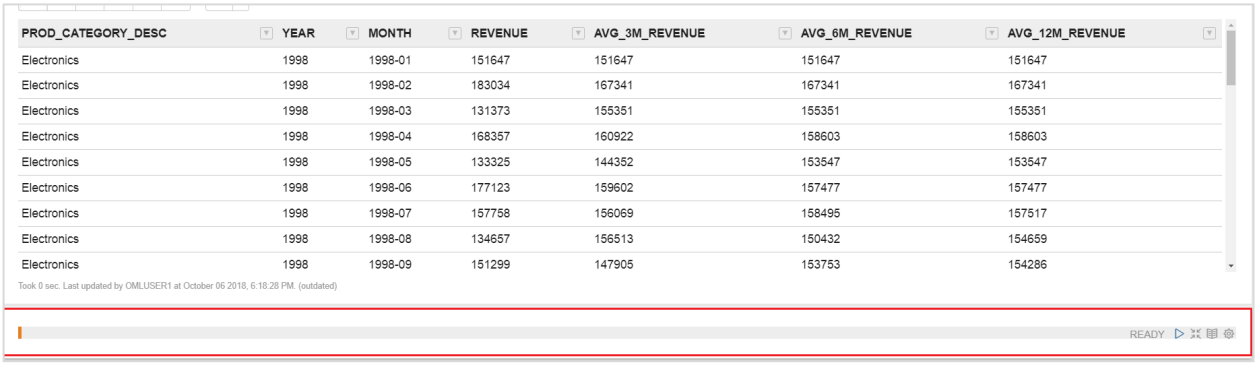


10. Name the title **This is My First Paragraph** as shown below.



11. To add one more paragraph in the notebook, click ⚙ and select **Insert new**. A new paragraph will be created.





PROD_CATEGORY_DESC	YEAR	MONTH	REVENUE	AVG_3M_REVENUE	AVG_6M_REVENUE	AVG_12M_REVENUE
Electronics	1998	1998-01	151647	151647	151647	151647
Electronics	1998	1998-02	183034	167341	167341	167341
Electronics	1998	1998-03	131373	155351	155351	155351
Electronics	1998	1998-04	168357	160922	158603	158603
Electronics	1998	1998-05	133325	144352	153547	153547
Electronics	1998	1998-06	177123	159602	157477	157477
Electronics	1998	1998-07	157758	156069	158495	157517
Electronics	1998	1998-08	134657	156513	150432	154659
Electronics	1998	1998-09	151299	147905	153753	154286

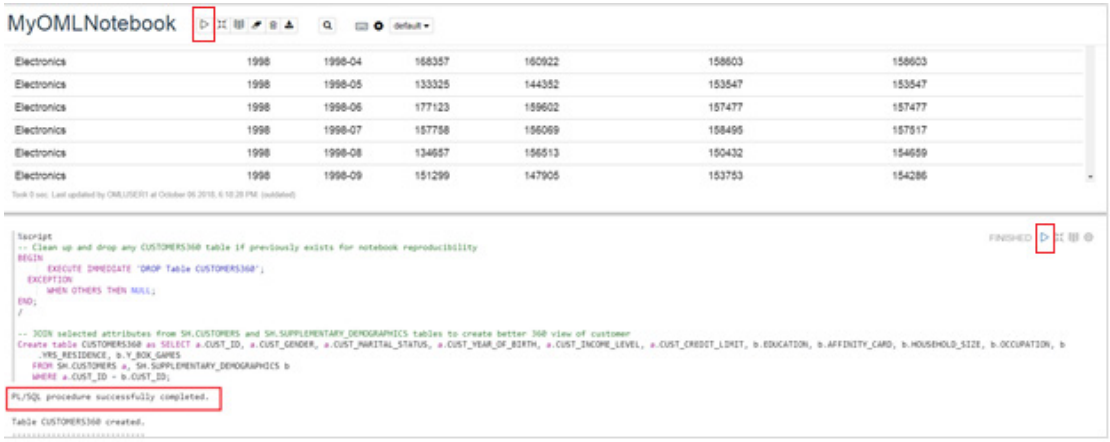
12. Title the paragraph "Clean Up Previously Existing Tables and Create a Data Table" with the help of previous steps. Copy and paste the following query in the second paragraph.

```
%script
-- Clean up and drop any CUSTOMERS360 table if previously
exists for notebook reproducibility
BEGIN
      EXECUTE IMMEDIATE 'DROP Table CUSTOMERS360';
  EXCEPTION
      WHEN OTHERS THEN NULL;
END;
/
```

JOIN selected attributes from the SH.CUSTOMERS and SH.SUPPLEMENTARY_DEMOGRAPHICS tables to create a better 360 degree view of the customer.

```
Create table CUSTOMERS360 as SELECT a.CUST_ID, a.CUST_GENDER,
a.CUST_MARITAL_STATUS, a.CUST_YEAR_OF_BIRTH,
a.CUST_INCOME_LEVEL, a.CUST_CREDIT_LIMIT, b.EDUCATION,
b.AFFINITY_CARD, b.HOUSEHOLD_SIZE, b.OCCUPATION,
b.YRS_RESIDENCE, b.Y_BOX_GAMES
   FROM SH.CUSTOMERS a,
SH.SUPPLEMENTARY_DEMOGRAPHICS b
   WHERE a.CUST_ID = b.CUST_ID;
```

a. Click the **Run this paragraph** icon shown below to execute the SQL statement and display the execution message.



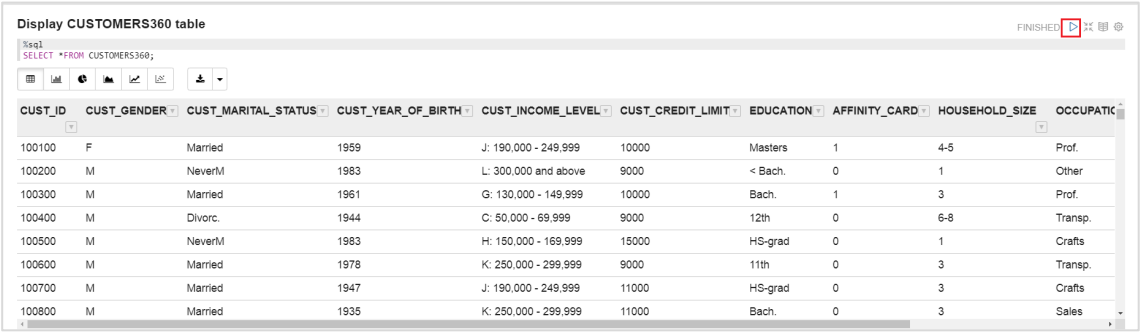
b. Create the paragraphs (in **MyOMLNotebook**) one by one with the tile name, copy and paste the following SQL statements, and run them one by one to see the execution messages.

Paragraph 3:

Tile Name: Display the CUSTOMERS360 Table

Query: Display the CUSTOMERS360 Table

```
%sql
SELECT *FROM CUSTOMERS360;
```



Display CUSTOMERS360 table

```
%sql
SELECT *FROM CUSTOMERS360;
```

CUST_ID	CUST_GENDER	CUST_MARITAL_STATUS	CUST_YEAR_OF_BIRTH	CUST_INCOME_LEVEL	CUST_CREDIT_LIMIT	EDUCATION	AFFINITY_CARD	HOUSEHOLD_SIZE	OCCUPATIO
100100	F	Married	1959	J: 190,000 - 249,999	10000	Masters	1	4-5	Prof.
100200	M	NeverM	1983	L: 300,000 and above	9000	< Bach.	0	1	Other
100300	M	Married	1961	G: 130,000 - 149,999	10000	Bach.	1	3	Prof.
100400	M	Divorc.	1944	C: 50,000 - 69,999	9000	12th	0	6-8	Transp.
100500	M	NeverM	1983	H: 150,000 - 169,999	15000	HS-grad	0	1	Crafts
100600	M	Married	1978	K: 250,000 - 299,999	9000	11th	0	3	Transp.
100700	M	Married	1947	J: 190,000 - 249,999	11000	HS-grad	0	3	Crafts
100800	M	Married	1935	K: 250,000 - 299,999	11000	Bach.	0	3	Sales

Paragraph 4:

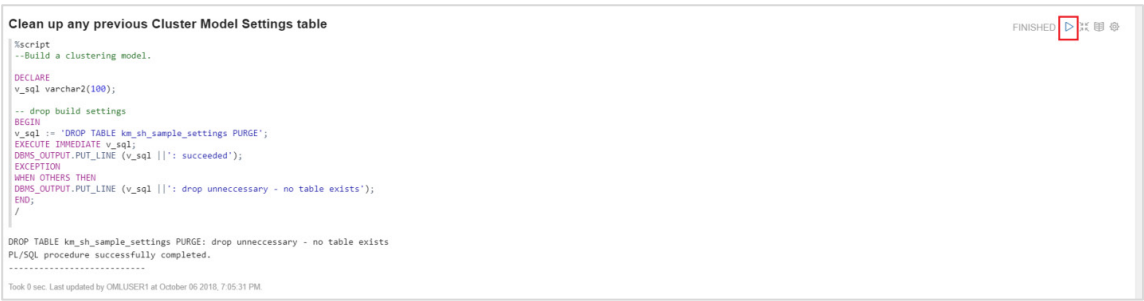
Title Name: Clean Up Any Previous Cluster Model Settings Table

Query:

```
%script
--Build a clustering model.

DECLARE
v_sql varchar2(100);

-- drop build settings
BEGIN
v_sql := 'DROP TABLE km_sh_sample_settings
PURGE';
EXECUTE IMMEDIATE v_sql;
DBMS_OUTPUT.PUT_LINE (v_sql ||': succeeded');
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE (v_sql ||': drop unnecessary
- no table exists');
END;
/
```



Clean up any previous Cluster Model Settings table

```
%script
--Build a clustering model.

DECLARE
v_sql varchar2(100);

-- drop build settings
BEGIN
v_sql := 'DROP TABLE km_sh_sample_settings PURGE';
EXECUTE IMMEDIATE v_sql;
DBMS_OUTPUT.PUT_LINE (v_sql ||': succeeded');
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE (v_sql ||': drop unneccessary - no table exists');
END;
/
```

```
DROP TABLE km_sh_sample_settings PURGE: drop unneccessary - no table exists
PL/SQL procedure successfully completed.
--------------------------
```

Took 0 sec. Last updated by OMLUSER1 at October 06 2018, 7:05:31 PM.

Paragraph 5:

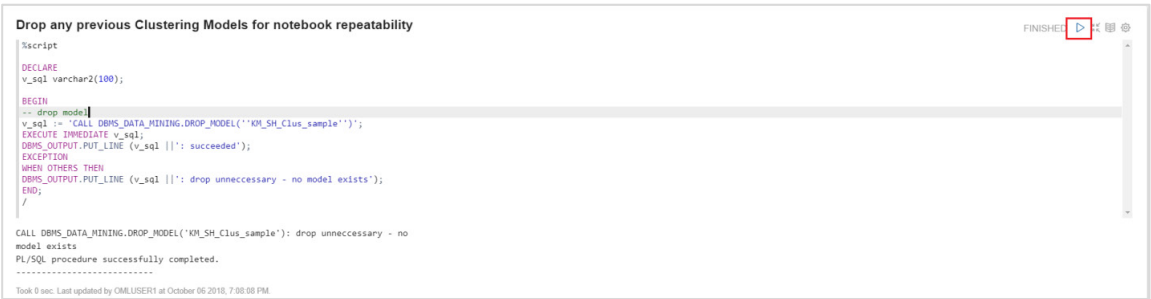
Title Name: Drop Any Previous Clustering Models for Notebook Repeatability

Query:

```
%script

DECLARE
v_sql varchar2(100);

BEGIN
-- drop model
v_sql := 'CALL
DBMS_DATA_MINING.DROP_MODEL(''KM_SH_Clus_sample'')
';
EXECUTE IMMEDIATE v_sql;
DBMS_OUTPUT.PUT_LINE (v_sql ||': succeeded');
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE (v_sql ||': drop unnecessary
- no model exists');
END;
/
```



Paragraph 6:

Title Name: Create a Model Settings Table

Query:

```
%script
DECLARE
v_sql varchar2(100);

-- Create a Build Setting (DT) for K-Means Model
Build

BEGIN
v_sql := 'CREATE TABLE km_sh_sample_settings
(setting_name  VARCHAR2(30),setting_value
VARCHAR2(4000))';
EXECUTE IMMEDIATE v_sql;
DBMS_OUTPUT.PUT_LINE (v_sql ||': succeeded');
EXCEPTION
WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE (v_sql ||': drop unnecessary
- no table exists');
END;
/
```

Paragraph 7

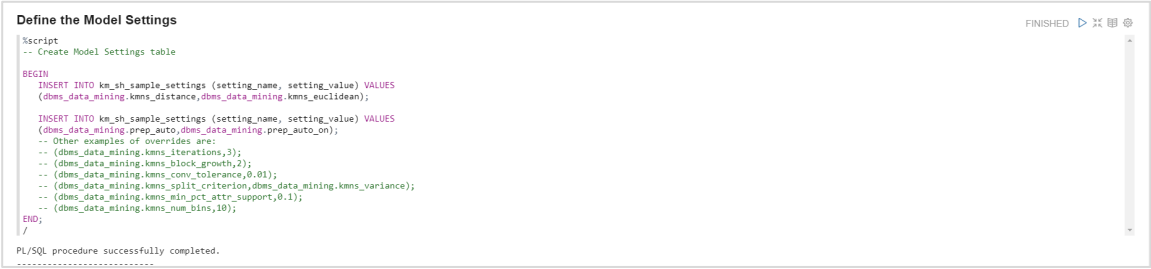
Title Name: Define the Model Settings

Query:

```
%script
-- Create Model Settings table
BEGIN
   INSERT INTO km_sh_sample_settings
(setting_name, setting_value) VALUES

(dbms_data_mining.kmns_distance,dbms_data_mining.
kmns_euclidean);
   INSERT INTO km_sh_sample_settings
(setting_name, setting_value) VALUES

(dbms_data_mining.prep_auto,dbms_data_mining.prep
_auto_on);
   -- Other examples of overrides are:
   -- (dbms_data_mining.kmns_iterations,3);
   -- (dbms_data_mining.kmns_block_growth,2);
   --
(dbms_data_mining.kmns_conv_tolerance,0.01);
   --
(dbms_data_mining.kmns_split_criterion,dbms_data_
mining.kmns_variance);
   --
(dbms_data_mining.kmns_min_pct_attr_support,0.1);
   -- (dbms_data_mining.kmns_num_bins,10);
END;
/
```

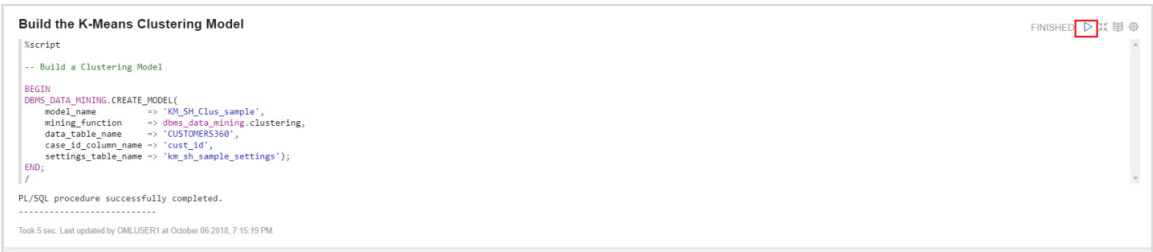


Paragraph 8

Title Name: Build the K-Means Clustering Model

Query:

```
%script
-- Build a Clustering Model
BEGIN
DBMS_DATA_MINING.CREATE_MODEL(
    model_name          => 'KM_SH_Clus_sample',
    mining_function     =>
dbms_data_mining.clustering,
    data_table_name     => 'CUSTOMERS360',
    case_id_column_name => 'cust_id',
    settings_table_name =>
'km_sh_sample_settings');
END;
/
```

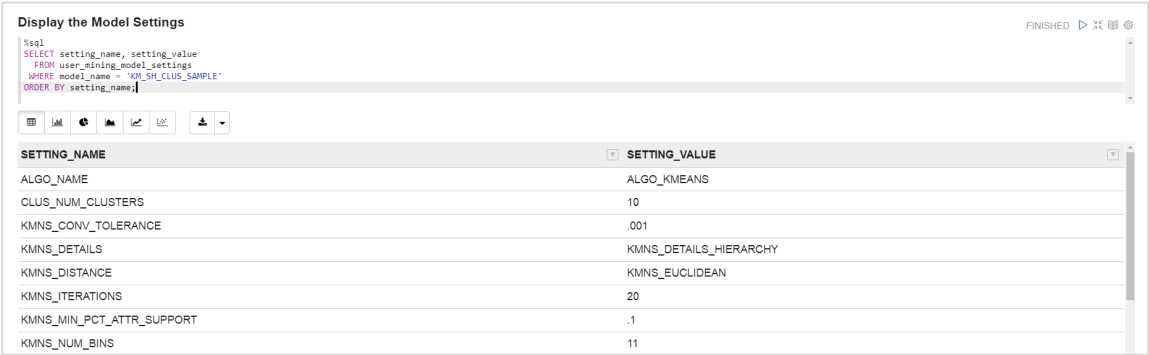


Paragraph 9

Title Name: Display the Model Settings

Query:

```
%sql
SELECT setting_name, setting_value
  FROM user_mining_model_settings
 WHERE model_name = 'KM_SH_CLUS_SAMPLE'
ORDER BY setting_name;
```

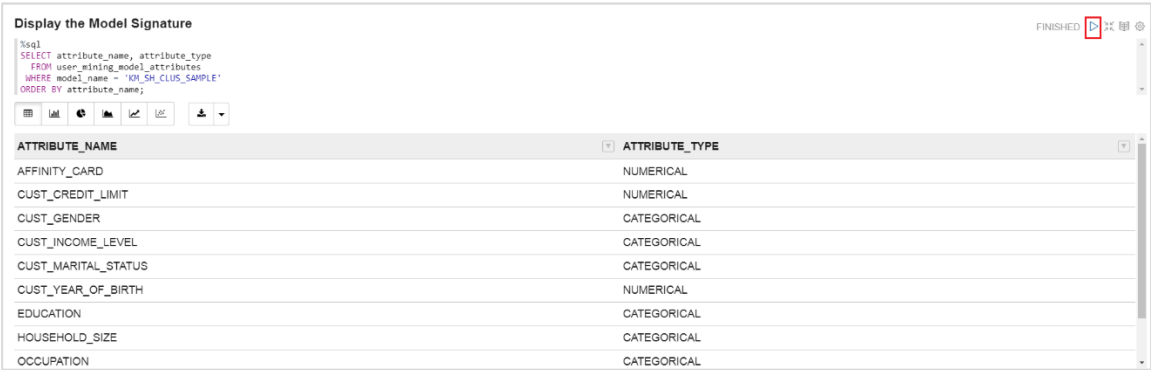


Paragraph 10:

Title Name: Display the Model Signature

Query:

```
%sql
SELECT attribute_name, attribute_type
  FROM user_mining_model_attributes
 WHERE model_name = 'KM_SH_CLUS_SAMPLE'
ORDER BY attribute_name;
```

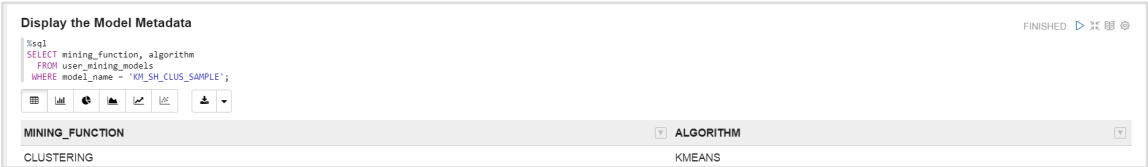


Paragraph 11:

Title Name: Display the Model Metadata

Query:

```
%sql
SELECT mining_function, algorithm
  FROM user_mining_models
 WHERE model_name = 'KM_SH_CLUS_SAMPLE';
```

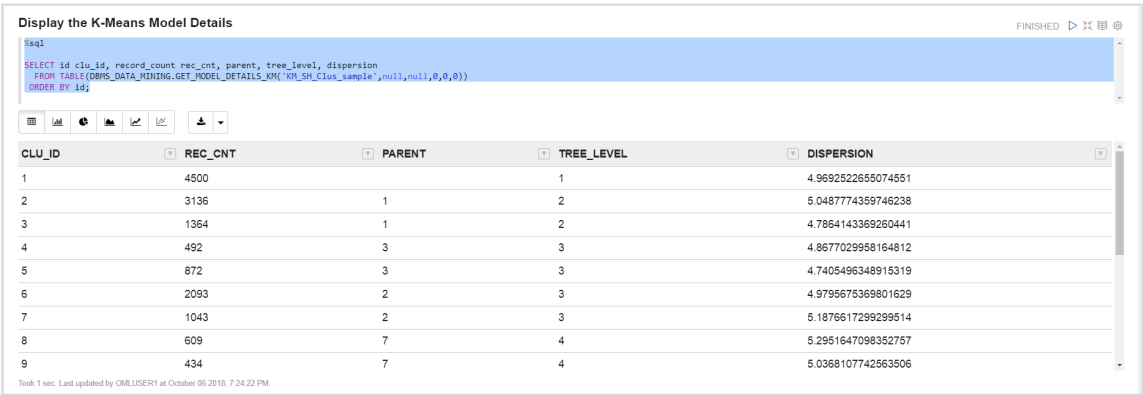


Paragraph 12:

Title Name: Display the K-Means Model Details

Query:

```
%sql
SELECT id clu_id, record_count rec_cnt, parent, tree_level,
dispersion
  FROM
TABLE(DBMS_DATA_MINING.GET_MODEL_DETAILS_KM('KM_SH_Clus_sample'
,null,null,0,0,0))
 ORDER BY id;
```



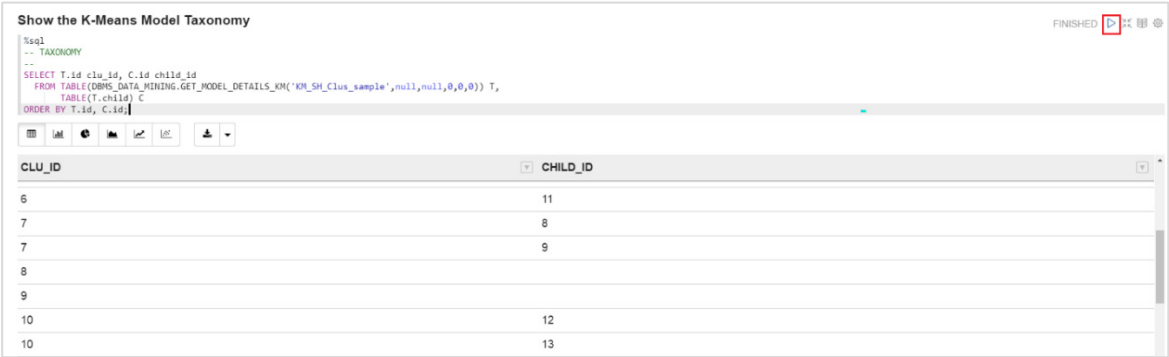
CLU_ID	REC_CNT	PARENT	TREE_LEVEL	DISPERSION
1	4500		1	4.9692522655074551
2	3136	1	2	5.0487774359746238
3	1364	1	2	4.7864143369260441
4	492	3	3	4.8677029958164812
5	872	3	3	4.7405496348915319
6	2093	2	3	4.9795675369801629
7	1043	2	3	5.1876617299299514
8	609	7	4	5.2951647098352757
9	434	7	4	5.0368107742563506

Paragraph 13:

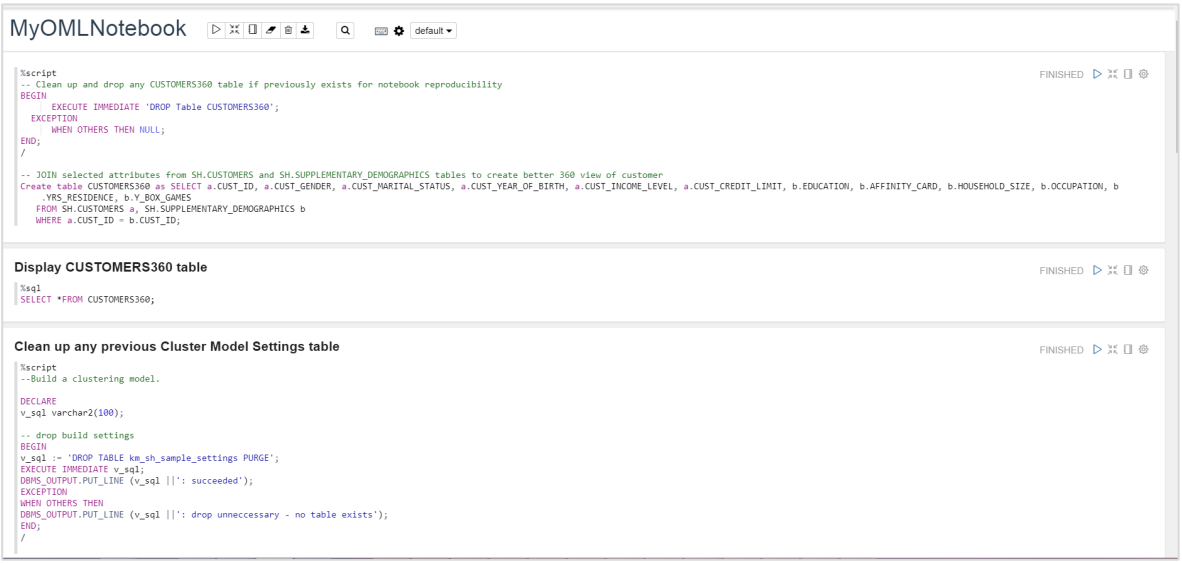
Title Name: Show the K-Means Model Taxonomy

Query:

```
%sql
SELECT T.id clu_id, C.id child_id
  FROM
TABLE(DBMS_DATA_MINING.GET_MODEL_DETAILS_KM('KM_SH_Clus_sample'
,null,null,0,0,0)) T,
       TABLE(T.child) C
ORDER BY T.id, C.id;
```



To show or hide the output, click  . It hides the output and displays the paragraphs.



It automatically saves all the paragraphs in the notebook as there is no Save option.

You have successfully created and run the notebook with all the paragraphs.

This completes the practice for creating and running Notebooks in Oracle Machine Learning.