Table of Contents

[CHAPTER 16: STANDARD AND ORACLE SUPPLIED PACKAGES 2](#_Toc414361171)

[Theory 2](#_Toc414361172)

[AIM 3](#_Toc414361173)

[Lab Exercise 16: STANDARD AND ORACLE SUPPLIED PACKAGES 4](#_Toc414361174)

[1. STANDARD Package 5](#_Toc414361175)

[2. DBMS\_OUTPUT Package 10](#_Toc414361176)

[3. UTL\_FILE Package 17](#_Toc414361177)

[4. UTL\_MAIL Package 26](#_Toc414361178)

[SUMMARY 33](#_Toc414361179)

[REFERENCES 34](#_Toc414361180)

[INDEX 35](#_Toc414361181)

# CHAPTER 16: STANDARD AND ORACLE SUPPLIED PACKAGES

## Theory

Oracle Supplied Packages are the packages created by Oracle to extend the functionality of PL/SQL. This includes most commonly required tasks that most developer might need, for example, accessing files, sending email, dealing with websites and creating dynamic SQL. Most of Oracle Supplied Packages are installed by default in Oracle Database. The other packages require you to follow the instructions of installation. You already see some Oracle Supplied Packages such as DBMS\_OUTPUT and DBMS\_RANDOM. In this tutorial, you will see other packages like UTL\_FILE and UTL\_MAIL.

A special Oracle Supplied Package named STANDARD defines the PL/SQL environment and is implicitly included in all PL/SQL block or subprograms. For examples, STANDARD package defines NUMBER, INTEGER, VARCHAR2 and DATE data types. It also defines some common functions like TO\_CHAR, TO\_NUMBER, TO\_DATE, ABS and many other functions. You don't need to refer the package name to use types, variables, and functions defined in a STANDARD package.

## AIM

The AIM of the following exercise is to demonstrate the use of some Oracle Supplied Packages in PL/SQL.

The steps involved will include:

* STANDARD package
* DBMS\_OUTPUT package
* UTL\_FILE Package
* UTL\_MAIL Package

In general, lab exercises are done in sequential order. Thus, it is assumed that you successfully completed the previous labs. However, not all previous labs are required. Please be sure to run the following lab before proceeding:

* Installing Oracle Database 12c.
* Package

Estimated Completion Time:

30 minutes

# Lab Exercise 16: STANDARD AND ORACLE SUPPLIED PACKAGES

|  |
| --- |
|  |

## STANDARD Package

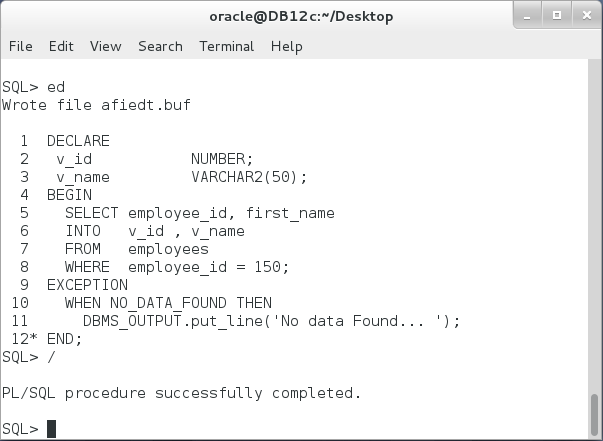
**Step 1:** Open the Terminal, open SQL\*Plus console and connect to hr schema.

|  |  |
| --- | --- |
| Command | Description |
| sqlplus | Open SQL\*Plus console. |
| hr/oracle | connect to **hr** schema. |

****

**Step 2:** Execute the following block:

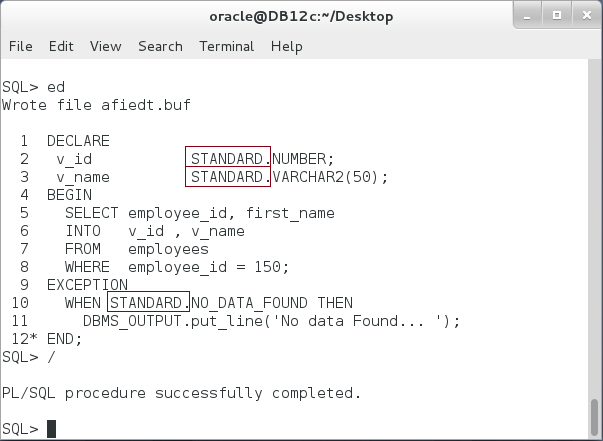
|  |  |
| --- | --- |
| Command | Description |
| DECLARE |  |
| v\_id NUMBER; |  |
| v\_name VARCHAR2(50); |
| BEGIN |
| SELECT employee\_id, first\_name |
| INTO v\_id , v\_name |
| FROM employees |
| WHERE employee\_id = 150; |
| EXCEPTION |
| WHEN NO\_DATA\_FOUND THEN |
| DBMS\_OUTPUT.put\_line('No data Found... '); |  |
| END; |  |
| / |  |

****

**Please note:** The previous package is a regular PL/SQL block with DECLARE, BEGIN, EXCEPTION and END clauses.

**Step 3:** Modify the previous block as shown below:

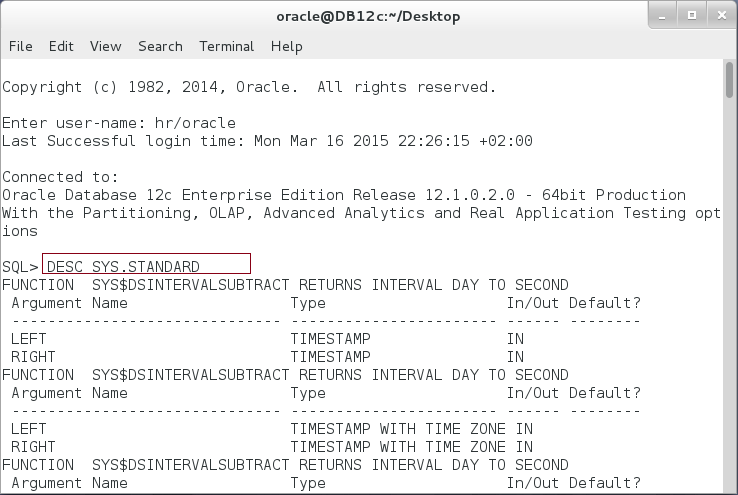
|  |  |
| --- | --- |
| Command | Description |
| DECLARE | STANDARD package defines all predefined types and subtypes. It defines exceptions and subprograms. |
| v\_id **STANDARD**.NUMBER; |
| v\_name **STANDARD**.VARCHAR2(50); |
| BEGIN |
| SELECT employee\_id, first\_name |
| INTO v\_id , v\_name |
| FROM employees |
| WHERE employee\_id = 150; |
| EXCEPTION |
| WHEN **STANDARD**.NO\_DATA\_FOUND THEN |
| DBMS\_OUTPUT.put\_line('No data Found... '); |
| END; |
| / |  |



**Please note:** STANDARD package is implicitly defined in all PL/SQL blocks. You don't have to specify STANDARD in front of variable's type.

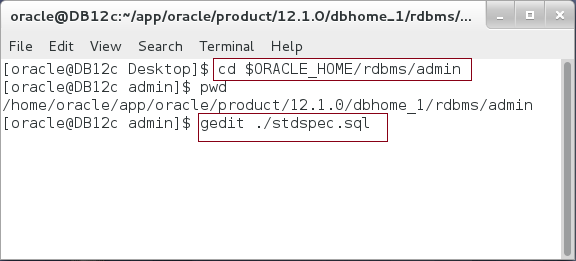
**Step 4:** You can explore the content of UTL\_MAIL package specification using "DESC" command as shown:

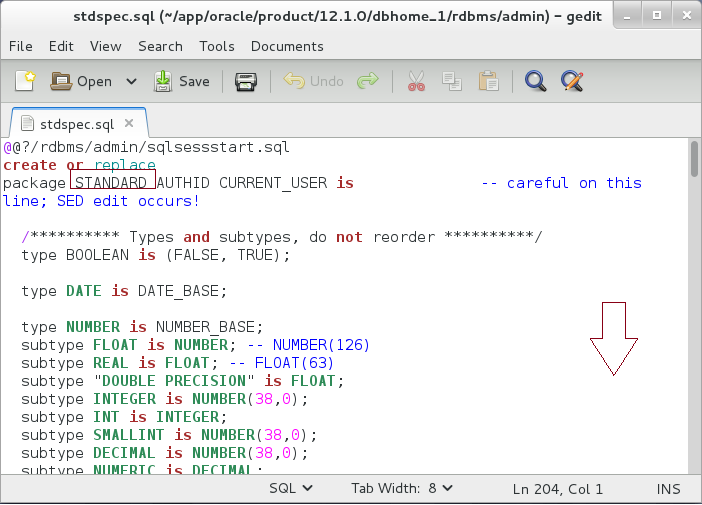
|  |  |
| --- | --- |
| Command | Description |
| DESC SYS.STANDARD | Describe the "SYS.STANDARD" |



**Step 5:** You may also use the package specification file. Open new terminal and execute the following:

|  |  |
| --- | --- |
| Command | Description |
| cd $ORACLE\_HOME/rdbms/admin | Open ORACLE\_HOME/rbdms/admin/stdspec.sql file |
| pwd |
| gedit ./stdspec.sql | . |

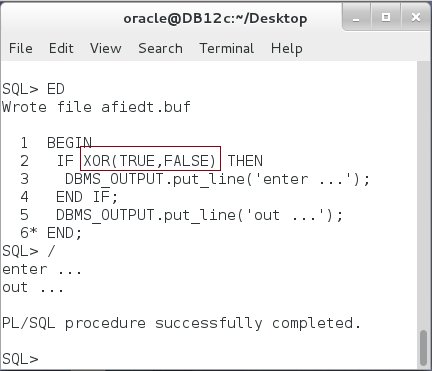


****

**Please note:** Take 5 minutes to explore STANDARD file.

**Step 6:** Find any new function in STANDARD and execute it in PL/SQL block as show below:

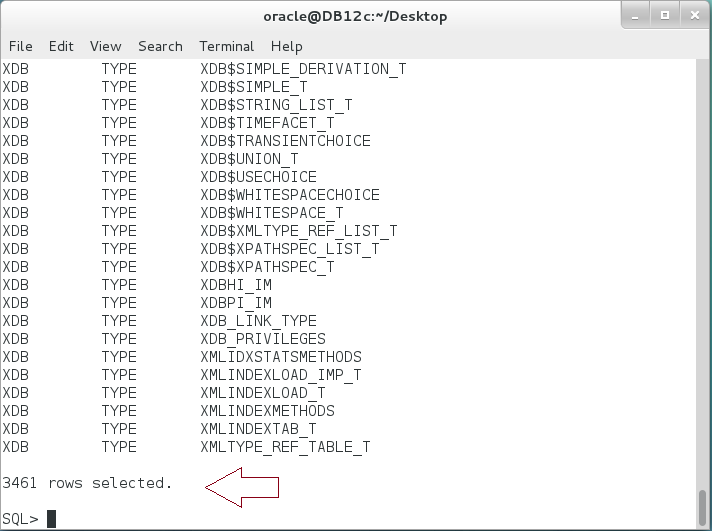
|  |  |
| --- | --- |
| Command | Description |
| BEGIN |  |
| IF **XOR(TRUE,FALSE)** THEN | XOR function |
| DBMS\_OUTPUT.put\_line('enter ...'); |
| END IF; |
| DBMS\_OUTPUT.put\_line('out ...'); |
| END; |
| / |

****

## DBMS\_OUTPUT Package

**Step 1:** DBMS\_OUTPUT is just one of many Oracle Supplied Packages. You should familiarize yourself with these packages. You can use the query below to list all Oracle Supplied Packages:

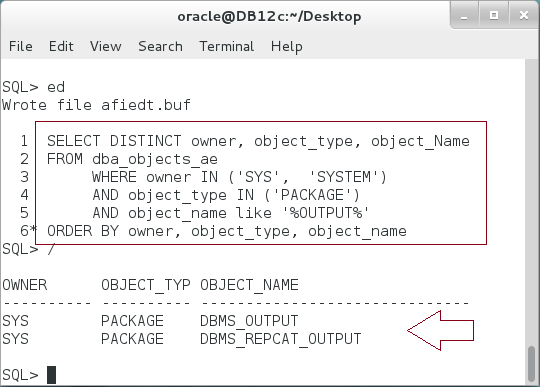
|  |  |
| --- | --- |
| Command | Description |
| SELECT DISTINCT owner, object\_Type, object\_Name | Query all Oracle Supplied Packages and Types. |
| FROM dba\_objects\_ae |
| WHERE owner IN ( |
| 'SYS', 'OUTLN', 'SYSTEM', 'CTXSYS', 'DBSNMP', |
| 'LOGSTDBY\_ADMINISTRATOR', 'ORDSYS', |
| 'ORDPLUGINS', 'OEM\_MONITOR', 'MDSYS', 'LBACSYS', |
| 'DMSYS', 'WMSYS', 'OLAPDBA', 'OLAPSVR', 'OLAP\_USER', |
| 'OLAPSYS', 'EXFSYS', 'SYSMAN', 'MDDATA', |
| 'SI\_INFORMTN\_SCHEMA', 'XDB', 'ODM') |
| AND object\_type IN ('PACKAGE', 'TYPE') |
| ORDER BY owner, object\_type, object\_name |
| / |

****

**Please note:** There are about 3461 different packages and types! It is not easy to know about them all but you can narrow down the search to find a relative packages.

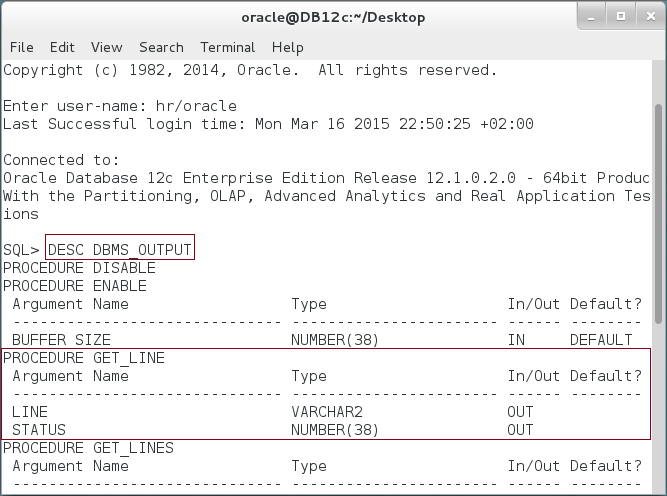
**Step 2:** You can query all packages in "SYS" or "SYSTEM" schemas that may concern about printing "OUTPUT" as shown below:

|  |  |
| --- | --- |
| Line | Description |
| SELECT DISTINCT owner, object\_type, object\_Name | Search for all Oracle Supplied Packages that may concern about printing "OUTPUT". |
| FROM dba\_objects\_ae |
| WHERE owner IN ('SYS', 'SYSTEM') |
| AND object\_type IN ('PACKAGE') |
| AND object\_name like '%OUTPUT%' |
| ORDER BY owner, object\_type, object\_name |
| / |



**Step 3:** To know more about a selected package, first you may use a description command:

|  |
| --- |
| Line |
| DESC DBMS\_OUTPUT |

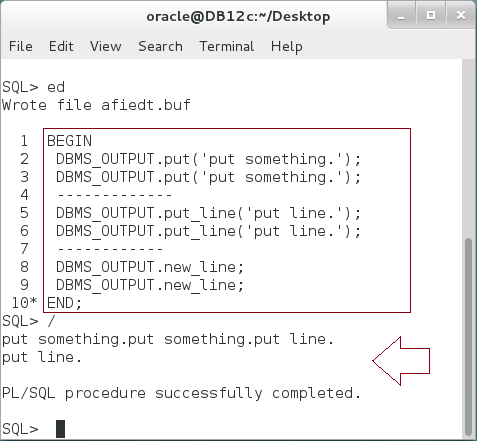


**Step 4:** You also need to review Oracle Documentation about the selected package, for example, to know about DBMS\_OUTPUT package visit this link:

|  |
| --- |
| Line |
| https://docs.oracle.com/database/121/ARPLS/d\_output.htm#ARPLS67314 |

**Step 4:** You should be familiar with DBMS\_OUTPUT package. You use it several times to print output to the SQL\*Plus screen. DBMS\_OUTPUT has many other options to put string to the output buffer. Execute the following block:

|  |  |
| --- | --- |
| Line | Description |
| BEGIN | PUT procedure |
| DBMS\_OUTPUT.put('put something.'); |
| DBMS\_OUTPUT.put('put something.'); |
| ------------- |  |
| DBMS\_OUTPUT.put\_line('put line.'); | PUT\_LINE procedure |
| DBMS\_OUTPUT.put\_line('put line.'); |
| ------------ |
| DBMS\_OUTPUT.new\_line; | NEW\_LINE procedure |
| DBMS\_OUTPUT.new\_line; |
| END; |
| / |  |



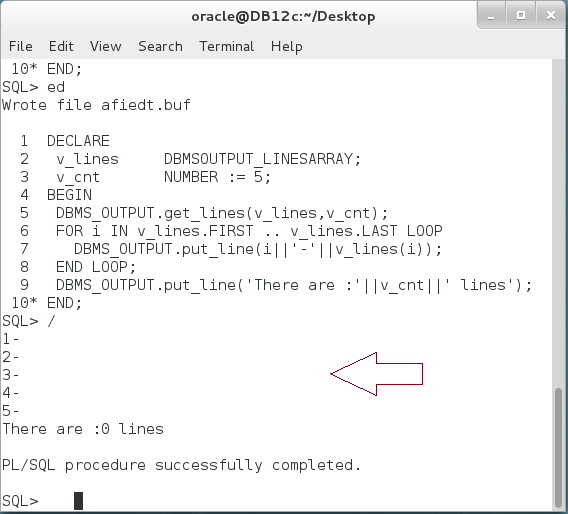
**Please note:** There are three different procedures that can be used to put strings to the output buffer: PUT, PUT\_LINE, and NEW\_LINE. **Did you recognize the difference between them?**

**Step 5:** Instead of displaying the output in the SQL\*Plus console, you can get/receive the output using GET\_LINES/GET\_LINE procedures. For example, GET\_LINES procedure takes two parameters:

1. OUT parameter "LINES" of DBMSOUTPUT\_LINESARRAY type.
2. IN/OUT parameter "NUMLINES" of NUMBER type -- IN the number of lines you want to fetch and OUT the real number of lines being fetched

Execute the following block:

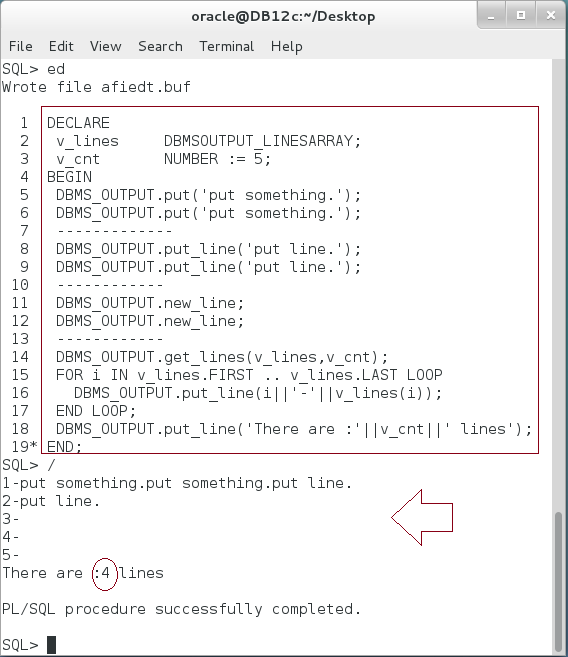
|  |  |
| --- | --- |
| Line | Description |
| DECLARE |  |
| v\_lines DBMSOUTPUT\_LINESARRAY; |  |
| v\_cnt NUMBER := 5; |  |
| BEGIN |  |
| DBMS\_OUTPUT.get\_lines(v\_lines,v\_cnt); | GET\_LINES procedure |
| FOR i IN v\_lines.FIRST .. v\_lines.LAST LOOP |
| DBMS\_OUTPUT.put\_line(i||'-'||v\_lines(i)); |
| END LOOP; |  |
| DBMS\_OUTPUT.put\_line('There are :'||v\_cnt||' lines'); |  |
| END; |  |
| / |  |



**Please note:** The number of lines required is 5 but the real lines number is 0. Note also that we already executed previous blocks that generate output. Thus, GET\_LINES does not get the buffer in the session level, it gets buffer generated in a working unit level.

**Step 5:** Modify the previous block as shown below:

|  |  |
| --- | --- |
| Line | Description |
| DECLARE |  |
| v\_lines DBMSOUTPUT\_LINESARRAY; |  |
| v\_cnt NUMBER := 5; |  |
| BEGIN |  |
| DBMS\_OUTPUT.put('put something.'); | Generate output |
| DBMS\_OUTPUT.put('put something.'); |
| ------------- |
| DBMS\_OUTPUT.put\_line('put line.'); |
| DBMS\_OUTPUT.put\_line('put line.'); |
| ------------ |
| DBMS\_OUTPUT.new\_line; |
| DBMS\_OUTPUT.new\_line; |
| ------------ |  |
| DBMS\_OUTPUT.get\_lines(v\_lines,v\_cnt); | Get the output generated previously. |
| FOR i IN v\_lines.FIRST .. v\_lines.LAST LOOP |
| DBMS\_OUTPUT.put\_line(i||'-'||v\_lines(i)); |
| END LOOP; |
| DBMS\_OUTPUT.put\_line('There are :'||v\_cnt||' lines'); |
| END; |  |
| / |  |

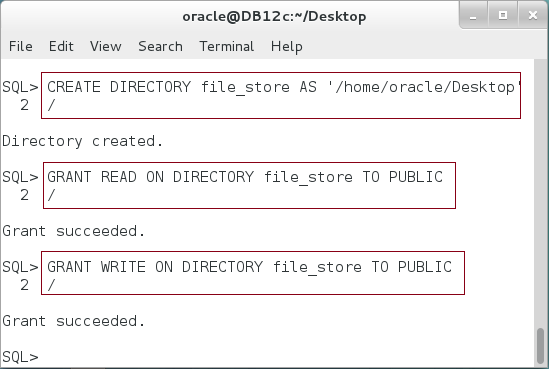


**Please explain the output.**

## UTL\_FILE Package

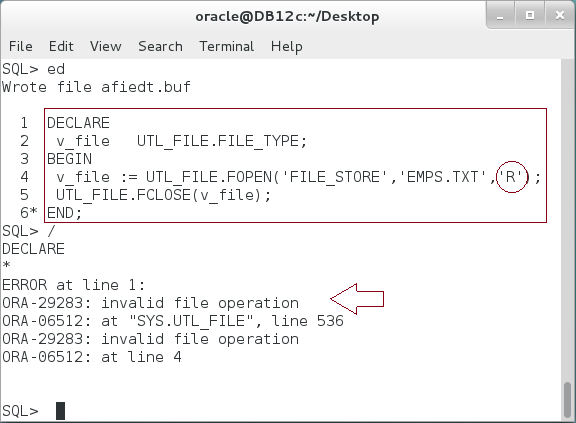
**Step 1:** UTL\_FILE allows you to manipulate files, either on the server side or on the client side. So, you can create, write, and read external files using UTL\_FILE. In order to locate the files, UTL\_FILE uses the Oracle DIRECTORY. Thus, you don't need to modify your code if the location of your files changed. Create a DIRECTORY as shown below:

|  |  |
| --- | --- |
| Command | Description |
| CREATE DIRECTORY file\_store AS '/home/oracle/Desktop' | Create a directory to point to your files location |
| / |
| GRANT READ ON DIRECTORY file\_store TO PUBLIC | Grant READ privilege to all users. |
| / |
| GRANT WRITE ON DIRECTORY file\_store TO PUBLIC | Grant WRITE privilege to all users. |
| / |

****

**Step 2:** We will try to create a file like the following:

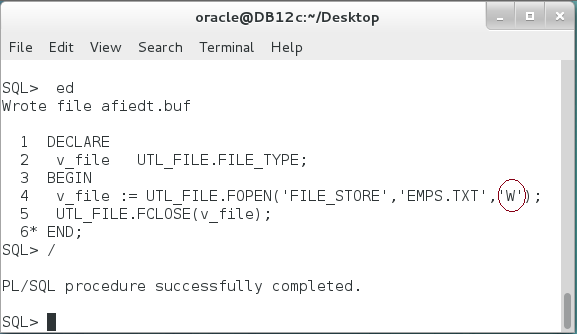
|  |  |
| --- | --- |
| Command | Description |
| DECLARE | FILE\_TYPE is a record defined inside UTL\_FILE package to store the file handler |
| v\_file UTL\_FILE.FILE\_TYPE; |
| BEGIN | FOPEN (Directory , File name, open mode) |
| v\_file := **UTL\_FILE.FOPEN**('FILE\_STORE','EMPS.TXT','R'); |
| UTL\_FILE.FCLOSE(v\_file); | Close the file and flush the changes to the desk, if any. |
| END; |
| / |  |

****

**Please note:** The error message indicates that the file is not exists.

**Step 3:** Modify the previous block, so it open the file for write not for read:

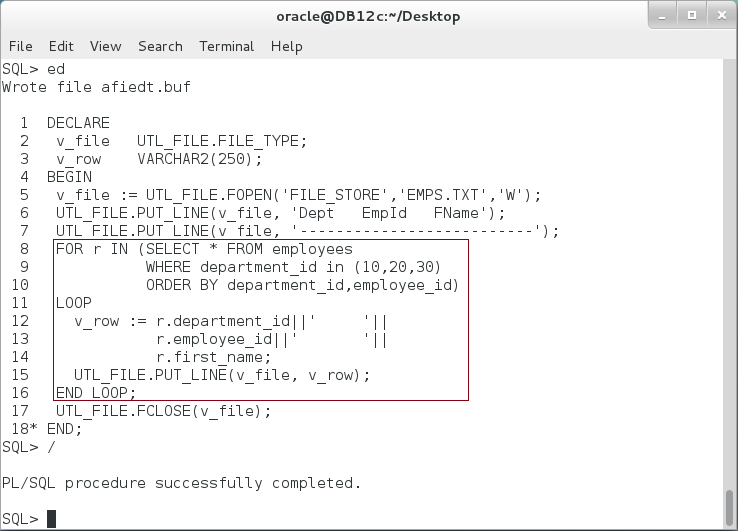
|  |  |
| --- | --- |
| Command | Description |
| DECLARE | Open the file for write. |
| v\_file UTL\_FILE.FILE\_TYPE; |
| BEGIN |
| v\_file := UTL\_FILE.FOPEN('FILE\_STORE','EMPS.TXT',**'W'**); |
| UTL\_FILE.FCLOSE(v\_file); |
| END; |
| / |

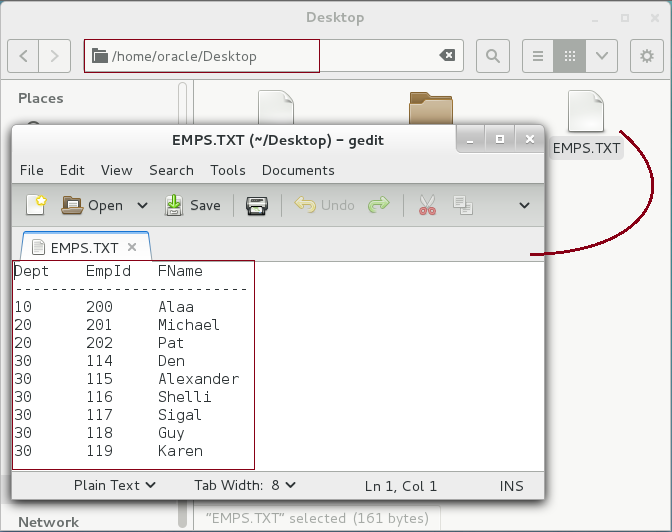
****

**Please note:** when you open a file for "write", it will create a file if the file does not exist. Please visit your directory to find the file.

**Step 4:** Modify the previous block to fill in the file with employees' information as shown below:

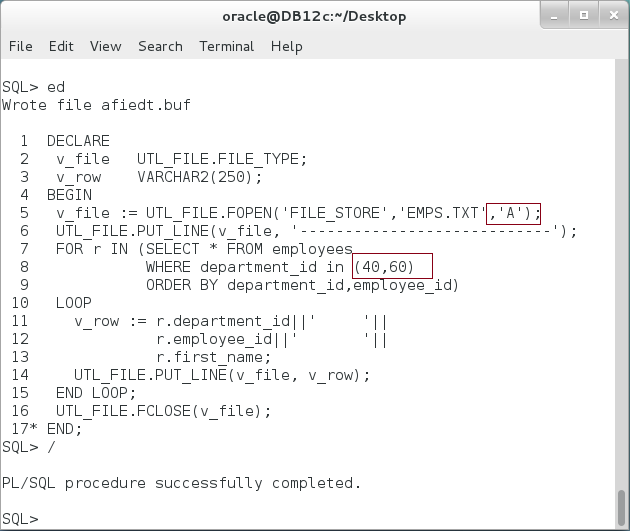
|  |  |
| --- | --- |
| Command | Description |
| DECLARE |  |
| v\_file UTL\_FILE.FILE\_TYPE; |  |
| v\_row VARCHAR2(250); |  |
| BEGIN |  |
| v\_file := UTL\_FILE.FOPEN('FILE\_STORE','EMPS.TXT','W'); |  |
| UTL\_FILE.PUT\_LINE(v\_file, 'Dept EmpId FName'); |  |
| UTL\_FILE.PUT\_LINE(v\_file, '--------------------------'); |  |
| **FOR r IN** (SELECT \* FROM employees | Write the employees' information to "EMPS.TXT" file. |
| WHERE department\_id in (10,20,30) |
| ORDER BY department\_id,employee\_id) |
| **LOOP** |
| v\_row := r.department\_id||' '|| |
| r.employee\_id||' '|| |
| r.first\_name; |
| **UTL\_FILE.PUT\_LINE**(v\_file, v\_row); |
| **END LOOP;** |  |
| UTL\_FILE.FCLOSE(v\_file); |  |
| END; |  |
| / |  |

****

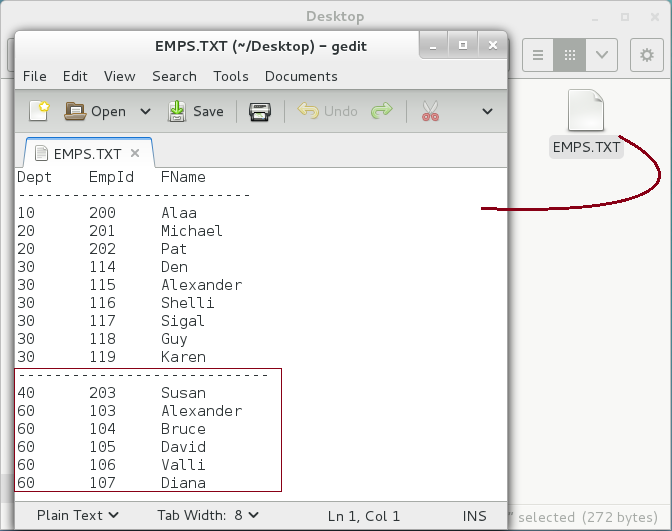
**Please note:** check the file and find the employees information

**Step 5:** To add new employees to the file, use append mode "A" as shown below:

|  |  |
| --- | --- |
| Command | Description |
| DECLARE |  |
| v\_file UTL\_FILE.FILE\_TYPE; |  |
| v\_row VARCHAR2(250); |  |
| BEGIN | Open file to **append** more data |
| v\_file := UTL\_FILE.FOPEN('FILE\_STORE','EMPS.TXT',**'A'**); |
| UTL\_FILE.PUT\_LINE(v\_file, '--------------------------------------'); |  |
| **FOR r IN** (SELECT \* FROM employees |  |
| WHERE department\_id in **(40,60)** | Add department "**40**" and "**60**" |
| ORDER BY department\_id,employee\_id) |
| **LOOP** |  |
| v\_row := r.department\_id||' '|| |  |
| r.employee\_id||' '|| |  |
| r.first\_name; |  |
| UTL\_FILE.PUT\_LINE(v\_file, v\_row); |  |
| **END LOOP;** |  |
| UTL\_FILE.FCLOSE(v\_file); |  |
| END; |  |
| / |

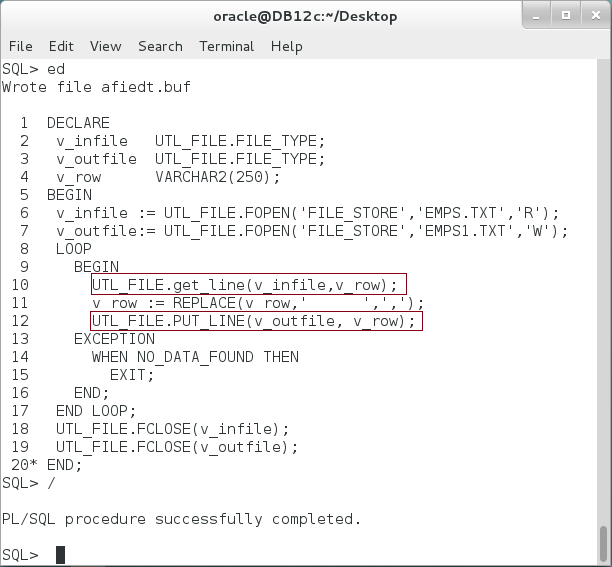
****

**Please note:**  check the file content to see the changes.

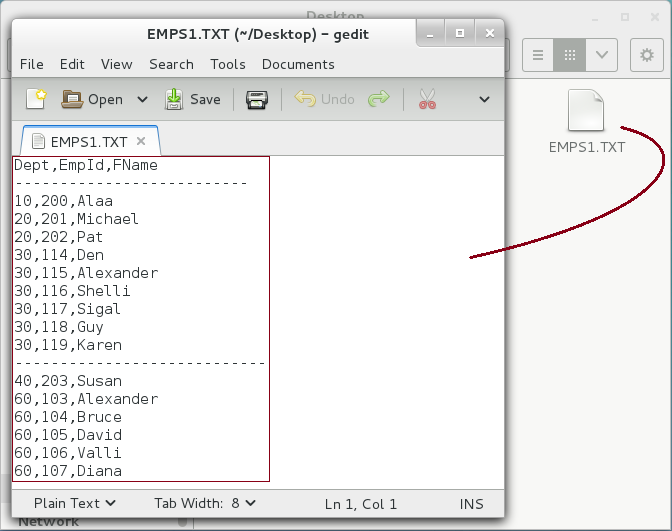


**Step 6:** You can modify the file as shown below:

|  |  |
| --- | --- |
| Command | Description |
| DECLARE |  |
| v\_infile UTL\_FILE.FILE\_TYPE; |  |
| v\_outfile UTL\_FILE.FILE\_TYPE; |  |
| v\_row VARCHAR2(250); |  |
| BEGIN |
| v\_infile := UTL\_FILE.FOPEN('FILE\_STORE','EMPS.TXT','R'); |  |
| v\_outfile:= UTL\_FILE.FOPEN('FILE\_STORE','EMPS1.TXT','W'); |  |
| LOOP |  |
| BEGIN |
| **UTL\_FILE.get\_line(**v\_infile,v\_row); | **Read** information |
| v\_row := REPLACE(v\_row,' ',','); |  |
| **UTL\_FILE.PUT\_LINE**(v\_outfile, v\_row); | **Write** information |
| EXCEPTION |  |
| WHEN NO\_DATA\_FOUND THEN |  |
| EXIT; |  |
| END; |  |
| END LOOP; |  |
| UTL\_FILE.FCLOSE(v\_infile); |
| UTL\_FILE.FCLOSE(v\_outfile); |  |
| END; |  |
| / |  |



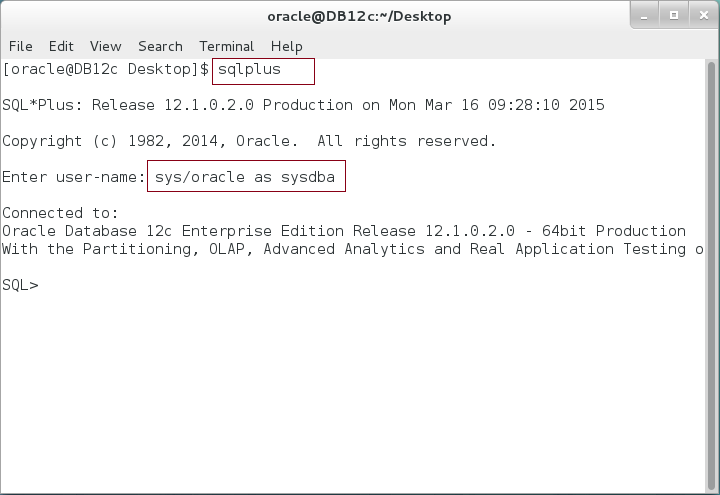
**Please note:**  check the new file's content.



## UTL\_MAIL Package

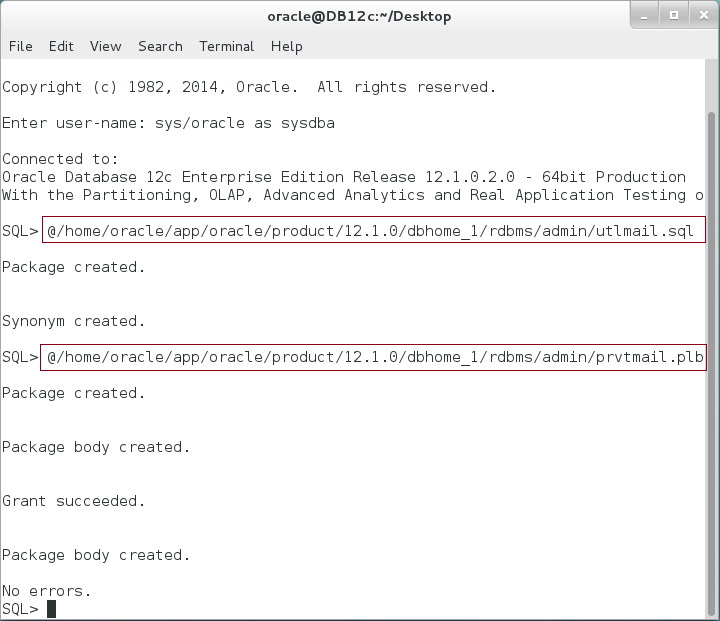
**Step 1:** UTL\_MAIL is not installed by default. You must installed it first. Connect using **sys** account.

|  |  |
| --- | --- |
| Command | Description |
| sqlplus | Open SQL\*Plus console. |
| sys/oracle as sysdba | connect to **sys** schema. |

****

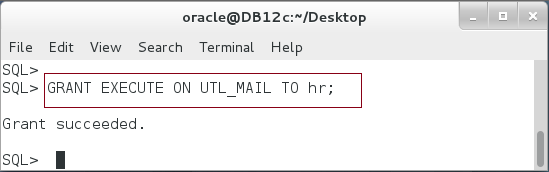
**Step 2:** Run the required scripts : "**utl\_mail.sql**" and "**prvtmail.plb**" as shown below:

|  |
| --- |
| Command |
| @/home/oracle/app/oracle/product/12.1.0/dbhome\_1/rdbms/admin/utlmail.sql |
| @/home/oracle/app/oracle/product/12.1.0/dbhome\_1/rdbms/admin/prvtmail.plb |



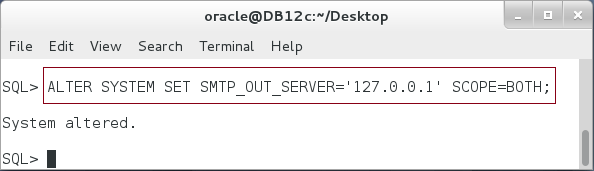
**Step 3:** Grant "hr" user an "EXECUTE" privilege on UTL\_MAIL package as shown:

|  |  |
| --- | --- |
| Command | Description |
| GRANT EXECUTE ON UTL\_MAIL TO hr; | Grant execute privilege to "hr". |



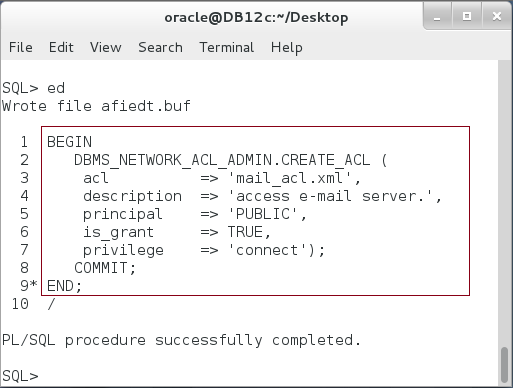
**Step 4:** Sending email is done using an outgoing SMTP mail server. You should make right your SMTP mail server instead of "127.0.0.1" in the next command. If you intend to install and configure the mail server in your host, please follow the suitable instructions according to your selected mail server and environment. In my host, I used "Postfix" which installed and almost completely configured in Oracle Enterprise Linux OEL7. Before proceeding, make sure your host can reach the SMTP mail host using "ping" command. After that, execute the following command in SQL\*Plus:

|  |
| --- |
| Command |
| ALTER SYSTEM SET SMTP\_OUT\_SERVER='127.0.0.1' SCOPE=BOTH; |

****

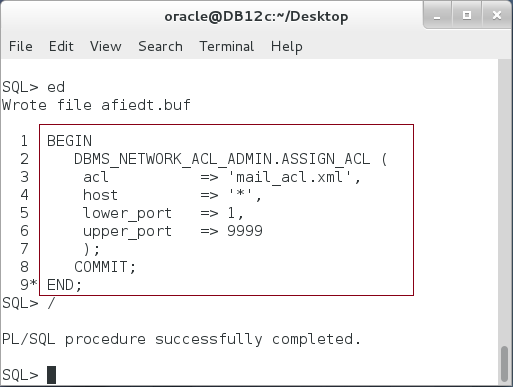
**Step 5:** Since Oracle 11g, it is not enough to grant execute on the UTL\_MAIL. You have to give a fine-grained privilege and specify for which the users are allowed to send email to. For tutorial purpose only, we will grant all users the privilege to send email to any mail server. It is highly recommended, however, to not give this "open" access in a real production environment. Fine-grained privileges should be done using XML files which you can set using "DBMS\_NETWORK\_ACL\_ADMIN" package. The first step is to create an Active Control List (ACL) as shown below:

|  |  |
| --- | --- |
| Command | Description |
| BEGIN |  |
| DBMS\_NETWORK\_ACL\_ADMIN.CREATE\_ACL ( |  |
| acl => 'mail\_acl.xml', | ACL file name |
| description => 'access e-mail server.', | file description |
| principal => 'PUBLIC', | Grant access to "PUBLIC" |
| is\_grant => TRUE, | Grant=TRUE, Revoke=FALSE |
| privilege => 'connect'); | Case sensitive "connect" privilege |
| COMMIT; |
| END; |  |
| BEGIN |  |



**Step 6:** After creating the ACL, you should assign the hosts and port numbers for which the ACL users are allowed to send mail. In our case, we use "\*" to refer to all hosts without restrictions.

|  |  |
| --- | --- |
| Command | Description |
| BEGIN |  |
| DBMS\_NETWORK\_ACL\_ADMIN.ASSIGN\_ACL ( |  |
| acl => 'mail\_acl.xml', | ACL file name |
| host => '\*', | all hosts |
| lower\_port => 1, | host port no. start from 1 |
| upper\_port => 9999 | host port no. ends with 9999 |
| ); |  |
| COMMIT; |  |
| END; |  |
| / |  |



**Step 7:** Finally, you can use a UTL\_MAIL to send the email as shown below:

|  |  |  |
| --- | --- | --- |
| Command | Description | |
| BEGIN | |  |
| UTL\_MAIL.SEND  (sender => 'elee@ConsultantNetwork.com', | | Sender email. |
| recipients => 'elee@ConsultantNetwork.com', | | Recipient email |
| subject => 'PL/SQL course', | | Mail subject |
| message => 'Hi,, | | Mail body |
| This is to explain how to | |
| send email using UTL\_MAIL', | |
| mime\_type => 'text/plain; charset=us-ascii' | | Send text. |
| ); | |  |
| END; | |  |



# SUMMARY

STANDARD package defines the PL/SQL environment. It defines most of the PL/SQL data types and exceptions in addition to a set of useful functions and procedures. STANDARD package is implicitly declared in all PL/SQL block; there is no need to refer its name before its elements. In addition to STANDARD package, you may use Oracle Supplied Packages and type instead of re-program them by yourself. There are more than 3000 Oracle Supplied Packages and types. Use a dictionary table to search for specific package and DESC command to know about its functions and procedures. You may also need to visit Oracle Documentation to view the usage details. Among most Oracle Supplied Packages used, there are DBMS\_OUTPUT, UTL\_FILE, and UTL\_MAIL. DBMS\_OUTPUT is most used to send and receive string output and used mostly for debugging purposes. UTL\_FILE is used to deal with external files. UTL\_MAIL is used to send emails from the database.

After completing this lab exercise, you should be aware about some of the most Oracle Supplied Packages.

# REFERENCES

* https://docs.oracle.com/database/121/ARPLS/intro.htm#ARPLS65020
* http://docs.oracle.com/database/121/LNPLS/packages.htm#LNPLS00907
* https://docs.oracle.com/database/121/ARPLS/d\_output.htm#ARPLS036
* https://docs.oracle.com/database/121/ARPLS/u\_file.htm#ARPLS069
* https://docs.oracle.com/database/121/ARPLS/u\_mail.htm#ARPLS384

# INDEX

Oracle Supplied Packages 2, 3, 10, 11, 33

STANDARD 2, 3, 4, 5, 6, 7, 9, 33