DBMS- LAB

Database Management System Lab Manual

DEPT OF ISE

PL / SQL

Ramaiah Institute Of Technology
Bengaluru

Introduction

PL/SQL is a combination of SQL along with the procedural features of programming languages. It was developed by Oracle Corporation in the early 90's to enhance the capabilities of SQL. PL/SQL is one of three key programming languages embedded in the Oracle Database, along with SQL itself and Java.

Syllabus

Consider the following database for a **BANK** system:

BRANCH (Code, Name, Assets)
CUSTOMER (SSN, Name, Place)
ACCOUNT (AccNo, SSN, Code, Balance)

Consider the following database for **EMPLOYEE** system:

EMPLOYEE (SSN,Name,Salary,DeptNo)

i) Create the above tables by stating the primary and foreign keys
ii) Insert the following tuples to the tables (As in page no. = 2)

Problem Statements :

1. Write a PL/SQL program to display the contents of the above tables and then update the balance of a few accounts.

 2. Write a program that gives all employees in department 10 a 15% pay increase. Display a message displaying how many employees were awarded the increase.

3. Write a PL/SQL program to check whether a given number is prime or not

 4. Using cursors demonstrate the process of copying the contents of one table to a new table

5. Write a PL/SQL program to print the first 8 fibonacci numbers

6. Write a PL/SQL procedure to find the factorial of a given number and a program to call the same

7. Write a PL/SQL program to check whether a given number is palindrome or not

8. Consider the following EMPLOYEE relation schema.Write a trigger to raise an error if the table is modified on a specific day (Eg., Saturday or Sunday) of the week

Database Schema with data

BANK

BRANCH		
CODE	NAME	ASSETS
B1	MSR	10000
B2	RNR	20000
B3	SMR	15000
B4	SKR	25000

ACCOUNT			
ACCNO	SSN	CODE	BALANCE
A1	1	B1	100000
A2	1	B1	200000
A3	2	B2	100000
A4	3	B2	100000
A5	3	B2	100000
A6	3	B2	100000
A7	4	B2	200000

CUSTOMER		
SSN	NAME	PLACE
1	RAM	BNG
2	ASHA	MNG
3	USHA	MYS
4	SRI	DEL

EMPLOYEE

EMPLOYEE			
SSN	NAME	SALARY	DEPTNO
111	RAM	10000	10
121	SAM	20000	10
131	TIM	30000	6
141	TOM	40000	5
151	JIM	50000	9

SQL Queries to build the schema and insert data

```
CREATE TABLE BRANCH_DETAIL (

    CODE VARCHAR(2) PRIMARY KEY,

    NAME VARCHAR(3),

    ASSETS NUMBER(6)  );

CREATE TABLE CUSTOMER_DETAIL(

    SSN NUMBER(1) PRIMARY KEY,

    NAME VARCHAR(20) ,

    PLACE VARCHAR(3)  );

CREATE TABLE ACCOUNT_DETAIL(

    ACCNO VARCHAR(2) PRIMARY KEY,

    SSN   NUMBER(1) REFERENCES CUSTOMER_DETAIL(SSN) ON DELETE CASCADE,

    CODE  VARCHAR(2) REFERENCES BRANCH_DETAIL(CODE) ON DELETE CASCADE,

    BALANCE NUMBER(7) );

CREATE TABLE EMPLOYEE_DETAIL (

    SSN NUMBER(3) PRIMARY KEY,

    NAME VARCHAR(20),

    SALARY NUMBER(6),

    DEPTNO NUMBER(3)  );


INSERT INTO BRANCH_DETAIL VALUES ('&CODE', '&NAME', &ASSETS);

INSERT INTO CUSTOMER_DETAIL VALUES (&SSN,'&NAME','&PLACE');

INSERT INTO ACCOUNT_DETAIL VALUES ('&ACCNO',&SSN,'&CODE',&BALANCE);

INSERT INTO EMPLOYEE_DETAIL VALUES (&SSN,'&NAME',&SALARY,&DEPTNO);
```

PL/SQL PROGRAMS :

1. Write a PL/SQL program to display the contents of the above tables and then update the balance of a few accounts.

PL/SQL PROGRAM :

```
SET SERVEROUTPUT ON

BEGIN

 FOR rec IN (SELECT * FROM BRANCH_DETAIL)

  LOOP

    dbms_output.put_line('CODE : ' || rec.code || ' NAME : '|| rec.name ||
' ASSETS : '|| rec.assets);

  END LOOP;

 FOR rec IN (SELECT * FROM CUSTOMER_DETAIL)

  LOOP

    dbms_output.put_line('SSN : ' || rec.ssn || ' NAME : '|| rec.name ||
' PLACE : '|| rec.place);

  END LOOP;

 FOR rec IN (SELECT * FROM ACCOUNT_DETAIL)

  LOOP

    dbms_output.put_line('ACCNo : ' || rec.accno || ' SSN : '|| rec.ssn
|| ' CODE : '|| rec.code || ' BALANCE : '|| rec.balance);

  END LOOP;

UPDATE ACCOUNT_DETAIL

SET BALANCE=120000

WHERE SSN=1;

dbms_output.put_line('SOME ROWS ARE UPDATED' );

END;

/
```

OUTPUT :

CODE : B1 NAME : MSR ASSETS : 10000
CODE : B2 NAME : RNR ASSETS : 20000
CODE : B3 NAME : SMR ASSETS : 15000
CODE : B4 NAME : SKR ASSETS : 25000

SSN : 1 NAME : Ram PLACE : BNG
SSN : 2 NAME : Asha PLACE : MNG
SSN : 3 NAME : Usha PLACE : MYS
SSN : 4 NAME : Sri PLACE : DEL

ACCNo : A1 SSN : 1 CODE : B1 BALANCE : 120000
ACCNo : A2 SSN : 1 CODE : B1 BALANCE : 120000
ACCNo : A3 SSN : 2 CODE : B2 BALANCE : 100000
ACCNo : A4 SSN : 3 CODE : B2 BALANCE : 100000
ACCNo : A5 SSN : 3 CODE : B2 BALANCE : 100000
ACCNo : A6 SSN : 3 CODE : B2 BALANCE : 100000
ACCNo : A7 SSN : 4 CODE : B2 BALANCE : 200000

SOME ROWS ARE UPDATED

SYNTAX :

FOR LOOP SYNTAX

```
FOR EACH_REC IN  (SQL QUERY) LOOP
      Sequence_of_statements ;
END LOOP;
```
- EACH_REC IS A ROW RETURNED BY THE SQL QUERY. IT IS A CURSOR

2. Write a program that gives all employees in department 10 a 15% pay increase. Display a message displaying how many employees were awarded the increase.

PL/SQL :

SET SERVEROUTPUT **ON**

BEGIN

UPDATE EMPLOYEE_DETAIL

SET SALARY = **CASE**

   WHEN DEPTNO = 10 **THEN** salary+(salary * 0.15)

   ELSE salary   -- *not strictly necessary. just to make sure.*

   END

WHERE DEPTNO IN (10);

`dbms_output.put_line(`**TO_Char**`(`**SQL%ROWCOUN**T)||' rows affected.') ;

END ;

/

OUTPUT :

2 rows affected.

SYNTAX :

CASE STATEMENT :

> **CASE** selector
> > **WHEN** 'value1' **THEN** S1;
> > **WHEN** 'value2' **THEN** S2;
> > **ELSE** Sn; *-- default case*
> **END**;

%ROWCOUNT
- It is an IMPLICIT CURSOR which returns the number of rows affected by an INSERT, UPDATE, or DELETE statement, or returned by a SELECT INTO statement.

3. Write a PL/SQL program to check whether a given number is prime or not

PL/SQL:

```
SET SERVEROUTPUT ON
DECLARE
    n number;
    i number;
    flag number;
BEGIN
    i:=2;
    flag:=1;
    n:=12;
    FOR i in 2..n/2
     LOOP
       IF MOD(n,i)=0 THEN
            flag:=0;
            exit;
         END IF ;
    END LOOP ;
    IF flag=1 THEN
      dbms_output.put_line('PRIME');
    ELSE
      dbms_output.put_line('NOT PRIME');
    END IF;
END;
/
```

OUTPUT :

Enter Value for n : 12

NOT PRIME

SYNTAX :

IF CONDITION :

```
IF condition THEN   S1;
ELSE
        S2;
END IF;
```

MOD FUNCTION :

- The Oracle PL/SQL **MOD** (short for *modulus*) function returns the remainder when one argument is divided by the second argument.

 4. Using cursors demonstrate the process of copying the contents of one table to a new table

PL/SQL PROGRAM :

```
SET SERVEROUTPUT ON

DECLARE
   c_id       employee_detail.ssn%type;
   c_name     employee_detail.Name%type;
   c_salary   employee_detail.salary%type;
   c_deptno   employee_detail.deptno%type;
   CURSOR c1
      IS  SELECT SSN ,NAME,SALARY,DEPTNO
         FROM  EMPLOYEE_DETAIL;
BEGIN
 OPEN c1;
 LOOP
   FETCH  c1  INTO c_id, c_name, c_salary,c_deptno;
   EXIT WHEN   c1%notfound;
INSERT INTO EMPLOYEE_DETAIL_COPYVALUES(c_id,c_name,c_salary,c_deptno);
END LOOP;
   CLOSE c1;
dbms_output.put_line('SUCCESSFULLY COPIED TO NEW TABLE');
END;
/
```

NOTE :

CREATE DUPLICATE TABLE BEFORE EXECUTING.

CREATE TABLE EMPLOYEE_DETAIL_COPY(

SSN **NUMBER**(3) **PRIMARY KEY**,

NAME **VARCHAR**(20),

SALARY **NUMBER**(6),

DEPTNO **NUMBER**(3));

OUTPUT :

SUCCESSFULLY COPIED TO NEW TABLE

SYNTAX :

CURSOR :

- A **cursor** is a pointer to this context area. PL/SQL controls the context area through a cursor. A cursor holds the rows(one or more) returned by a SQL statement.

OPEN cursor ;

- Opening the cursor allocates the memory for the cursor and makes it ready for fetching the rows returned by the SQL statement into it.

FETCH cursor INTO [VARIABLES] ;

- Fetching the cursor involves accessing one row at a time.

CLOSE cursor;

- Closing the cursor means releasing the allocated memory.

5. Write a PL/SQL program to print the first 8 fibonacci numbers

PL/SQL PROGRAM :

```
SET SERVEROUTPUT ON
DECLARE
    first      number :=0;
    Second     number:=1;
    third      number;
    n          number:=8;
    i          number;
 BEGIN
    dbms_output.put_line('Fibonacci series is:');
    dbms_output.put_line(first);
    dbms_output.put_line(second);
    FOR i iN  2..n
  LOOP
        third := first + second ;
        first:= second ;
        second:= third;
        dbms_output.put_line(third);
    END LOOP;
END;
/
```

OUTPUT :

Fibonacci series is:

0

1

1

2

3

5

8

13

21

6. Write a PL/SQL procedure to find the factorial of a given number and a program to call the same

PL/SQL PROCEDURE :

```
SET SERVEROUTPUT ON

CREATE OR REPLACE PROCEDURE findFactorial

AS

 n number;

 fac number:=1;

 i number;

 BEGIN

   n:=&n;

   FOR i IN 1..n

    LOOP

        fac:= fac * i ;

    END LOOP;

    dbms_output.put_line('Factorial='||fac);

END;

/

EXECUTE findFactorial;                  -- EXECUTING THE PROCEDURE

SHOW ERROR PROCEDURE findFactorial;  --debugging query to see the errors
```

OUTPUT :

Enter value of n : 5

Factorial = 120

7. Write a PL/SQL program to check whether a given number is palindrome or not

PL/SQL PROGRAM :

```
SET SERVEROUTPUT ON
DECLARE
    str1 varchar(50):='&n';
    str2 varchar(50);
    len number;
    i number;
 BEGIN
    len: = length(str1);
    FOR i IN REVERSE 1..len
    LOOP
        str2:=str2 || substr(str1,i,1);
    END LOOP;
    IF str1=str2 THEN
        dbms_output.put_line('IT'S PALINDROME');
    ELSE
        dbms_output.put_line('IT'S NOT PALINDROME');
    END IF ;
END;
/
```

OUTPUT :

Enter value of n : AAABBAAA

IT'S NOT PALINDROME

SYNTAX :

REVERSE FOR LOOP :

      FOR var **IN REVERSE** l.. 20

        LOOP S1;

        END LOOP ;

LENGTH FUNCTION :

    LENGTH (string1);

- LENGTH function returns the length of the specified string.

SUBSTR FUNCTION :

    SUBSTR(string , start_position , length);

- SUBSTR functions allows you to extract a substring from a string.

8. Consider the following EMPLOYEE relation schema.Write a trigger to raise an error if the table is modified on a specific day (Eg., Thursday or Wednesday) of the week

PL/SQL TRIGGER :

```
SET SERVEROUTPUT ON

CREATE OR REPLACE TRIGGER tri_employee

BEFORE insert or update

ON EMPLOYEE_DETAIL

FOR EACH ROW

DECLARE

    rec varchar2(10) ;

BEGIN

    SELECT to_char(sysdate,'Dy') INTO rec FROM dual;

    IF rec = 'Thu' OR rec='Wed' THEN

        dbms_output.put_line(rec);

        raise_application_error(-20343, 'NOT ALLOWED TO ENTER');

    END IF ;

END ;

/

show error

INSERT INTO EMPLOYEE_DETAIL VALUES (499,'RAM',10000,10);
```

OUTPUT :

Trigger created.

No errors.

Thu

INSERT INTO EMPLOYEE_DETAIL_COPY VALUES (499,'RAM',10000,10)

ERROR at line 1:
ORA-20343: **NOT ALLOWED TO ENTER**

SYNTAX :

TRIGGER :

- Triggers are stored programs, which are automatically executed or fired when some events occur.

to_char(**sysdate** , 'Dy') **FUNCTION:**

- Strips first three letters of the day of the week from current date, current date is returned by **sysdate** and converts it into varchar.

RAISE APPLICATION ERROR :

```
raise_application_error(error number, 'error message');
```

- It raises application error with given error message and number.

NOT IN SYLLABUS OF LAB

- BUT IN THEORY

1. Write a PL/SQL FUNCTION to find the factorial of a given number and a program to call the same

```
SET SERVEROUTPUT ON
DECLARE
    a number;
    b number;
    fac number :=1;
    i number;
FUNCTION findFactorial(x IN number)
RETURN number IS z number;
BEGIN
    FOR i IN 1..x
     LOOP
         fac:= fac * i ;
     END LOOP;
     z:=fac;
    RETURN z;
END;
BEGIN
    a:= 7;
    b:= findFactorial(a);
    dbms_output.put_line(' Factorial of 7 is ' || b);
END;
/
```

Additional Source

Install MongoDB Enterprise in Ubuntu :

1 . Import Public Key (RSA) for Package Mgmt.

dbms@dbmslab $ sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 0C49F3730359A14518585931BC711F9BA15703C6

2. Install according the system requirements

**For Ubuntu Xenial(16.04) please refer your OS and Package **

dbms@dbmslab $ echo "deb [arch=amd64,arm64,ppc64el,s390x] http://repo.mongodb.com/apt/ubuntu xenial/mongodb-enterprise/3.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-enterprise.list

**For Ubuntu Trusty (14.04) please refer your OS and Package **

dbms@dbmslab $ echo "deb [arch=amd64] http://repo.mongodb.com/apt/ubuntu trusty/mongodb-enterprise/3.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-enterprise.list

dbms@dbmslab $ sudo apt-get install mongodb-enterprise

3. Now after Successful installation . to start MongoDB Server issue the following cmd

dbms@dbmslab $ sudo service mongod start

4. Now to get MongoShell ,issue the following cmd

dbms@dbmslab $ mongo

You will have Mongo Shell Running in the terminal

5.Now to stop MongoDB Server , issue the following cmd

dbms@dbmslab $ sudo service mongod stop

Other Useful References :

Tutorial For Oracle SQL and PL/SQL

https://www.tutorialspoint.com/plsql/index.htm

https://www.tutorialspoint.com/oracle_sql/index.asp

Tutorial For MongoDB

www.tutorialspoint.com/mongodb/

Documentation Of MongoDb and Oracle

https://docs.mongodb.com/manual/

https://docs.oracle.com/cd/E11882_01/nav/portal_4.htm

(Be Great if You find the above link helpful)

Online Execution

https://www.tutorialspoint.com/oracle_terminal_online.php

HAPPY CODING !