

PL/SQL Assignment 5

ANUBHAV SINGH

IC2K1654

PL/SQL ONLINE CLASS ASSIGNMENT 5

True or False:

1. When you use DML in a PL/SQL block, Oracle uses explicit cursors to track the data changes.

TRUE

2. **EXPLICIT** cursors are created by the programmer.

3. **IMPLICIT** cursors are created by the Oracle server.

4. The following code is supposed to display the lowest and highest elevations for a country name entered by the user. However, the code does not work. Fix the code by following the guidelines for retrieving data that you learned in this lesson.

```
DECLARE
v_country_name wf_countries.country_name%TYPE
:= 'United States of America';
v_lowest_elevation wf_countries.lowest_elevation%TYPE;
v_highest_elevation wf_countries.highest_elevation%TYPE;
BEGIN
SELECT lowest_elevation, highest_elevation
INTO v_lowest_elevation, v_highest_elevation
FROM wf_countries;
DBMS_OUTPUT.PUT_LINE('The lowest elevation in
'||country_name||' is '||v_lowest_elevation
||' and the highest elevation is '||
v_highest_elevation||'.');
END;
```

```
SQL> create table countries(country_name varchar2(30),lowest_elevation varchar2(30),highest_elevation varchar2(30));
Table created.

SQL> insert into countries(country_name,lowest_elevation,highest_elevation)
  2  values
  3  ('USA','BOSTON','CHICAGO');
1 row created.
```

PL/SQL Assignment 5

```
SQL> DECLARE
2 v_country_name countries.country_name%TYPE := 'USA';
3 v_highest_elevation countries.highest_elevation%TYPE;
4 v_lowest_elevation countries.lowest_elevation%TYPE;
5 BEGIN
6 SELECT lowest_elevation, highest_elevation INTO v_lowest_elevation, v_highest_elevation FROM countries;
7 DBMS_OUTPUT.PUT_LINE('The lowest elevation in '||v_country_name||' is '||v_lowest_elevation||' and the highest elevation is '||v_highest_elevation||'.');
8 end;
9 /
The lowest elevation in USA is BOSTON and the highest elevation is CHICAGO.
PL/SQL procedure successfully completed.
```

5. How many transactions are shown in the following code? Explain your reasoning.

```
BEGIN
INSERT INTO my_savings (account_id, amount)
VALUES (10377,200);
INSERT INTO my_checking(account_id, amount)
VALUES (10378,100);
END;
```

2 transaction states

1st one is to insert data into my_saving and

2nd one is to insert data into my_checking.

6. Examine the following block. If you were to run this block, what data do you think would be saved in the database?

```
BEGIN
INSERT INTO endangered_species
VALUES (100, 'Polar Bear','Ursus maritimus');
SAVEPOINT sp_100;
INSERT INTO endangered_species
VALUES (200, 'Spotted Owl','Strix occidentalis');
SAVEPOINT sp_200;
INSERT INTO endangered_species
VALUES (300, 'Asiatic Black Bear','Ursus thibetanus');
ROLLBACK TO sp_100;
COMMIT;
END;
```

/

```
SQL> create table endangered_species(specie_id int,specie_name varchar(20),specie_bio varchar(20));
Table created.

SQL> BEGIN
2 INSERT INTO endangered_species VALUES (100, 'Polar Bear','Ursus maritimus');
3 SAVEPOINT sp_100;
4 INSERT INTO endangered_species VALUES (200, 'Spotted Owl','Strix occidentalis');
5 SAVEPOINT sp_200;
6 INSERT INTO endangered_species VALUES (300, 'Asiatic Black Bear','Ursus thibetanus');
7 ROLLBACK TO sp_100;
8 COMMIT;
9 END;
10 /

PL/SQL procedure successfully completed.

SQL> select * from endangered_species;

SPECIE_ID SPECIE_NAME      SPECIE_BIO
-----
100 Polar Bear             Ursus maritimus
```

PL/SQL Assignment 5

7. List the three categories of control structures in PL/SQL with example.

**Transaction control structure,
Conditional control structure,
Iterative control structure**

1. IF ELSE.

```
DECLARE
a number(2) := 10;
BEGIN
a:= 10;
IF( a < 20 )
THEN
dbms_output.put_line('a is less than 20 ');
END IF;
dbms_output.put_line('value of a is : ' || a);
END;
/
```

```
SQL> @ifelse
a is less than 20
value of a is : 10

PL/SQL procedure successfully completed.
```

2. IF THEN ELSE

```
DECLARE
a number(3) := 100;
BEGIN
IF( a < 20 )
THEN
dbms_output.put_line('a is less than 20 ');
ELSE
dbms_output.put_line('a is not less than 20 ');
END IF;
dbms_output.put_line('value of a is : ' || a);
END;
/
```

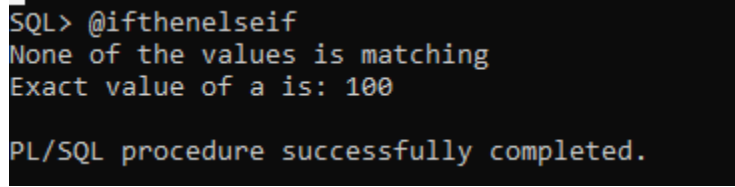
```
SQL> @ifthenelse
a is not less than 20
value of a is : 100

PL/SQL procedure successfully completed.
```

PL/SQL Assignment 5

3. IF THEN ELSEIF

```
DECLARE
a number(3) := 100;
BEGIN
IF ( a = 10 )
THEN
dbms_output.put_line('Value of a is 10' );
ELSIF ( a = 20 )
THEN
dbms_output.put_line('Value of a is 20' );
ELSIF ( a = 30 )
THEN
dbms_output.put_line('Value of a is 30' );
ELSE
dbms_output.put_line('None of the values is matching');
END IF;
dbms_output.put_line('Exact value of a is: ' || a );
END;
/
```



```
SQL> @ifthenelseif
None of the values is matching
Exact value of a is: 100

PL/SQL procedure successfully completed.
```

4. CASE Statement.

```
DECLARE
grade char(1) := 'A';
BEGIN
CASE grade
when 'A' then
dbms_output.put_line('Excellent');
when 'B' then
dbms_output.put_line('Very good');
when 'C' then
dbms_output.put_line('Well done');
when 'D' then
dbms_output.put_line('You passed');
when 'F' then
dbms_output.put_line('Better try again');
else
dbms_output.put_line('No such grade');
END CASE;
END;
/
```

PL/SQL Assignment 5

```
SQL> @case
Excellent

PL/SQL procedure successfully completed.
```

5. Searched CASE Statement.

```
DECLARE
grade char(1) := 'B';
BEGIN
case
when grade = 'A'
then
dbms_output.put_line('Excellent');
when grade = 'B'
then
dbms_output.put_line('Very good');
when grade = 'C'
then
dbms_output.put_line('Well done');
when grade = 'D'
then
dbms_output.put_line('You passed');
when grade = 'F' then
dbms_output.put_line('Better try again');
else
dbms_output.put_line('No such grade');
end case;
end;
/
```

```
SQL> @searchedcase
Very good

PL/SQL procedure successfully completed.
```

6. LOOPS

```
DECLARE
x number := 10;
BEGIN
LOOP
dbms_output.put_line(x);
x := x + 10;
IF x > 50
THEN exit;
END IF;
END LOOP;
```

PL/SQL Assignment 5

```
dbms_output.put_line('After Exit x is: ' || x);  
END;  
/
```

```
SQL> @loops  
10  
20  
30  
40  
50  
After Exit x is: 60  
  
PL/SQL procedure successfully completed.
```

8. List the keywords that can be part of an IF statement with one example each.

IF, THEN, ELSE, ELSIF, END IF

```
DECLARE  
a number(3) := 100;  
b number(3) := 200;  
BEGIN  
IF( a = 100 )  
THEN  
IF( b = 200 )  
THEN  
dbms_output.put_line('Value of a is 100 and b is 200' );  
END IF;  
END IF;  
dbms_output.put_line('Exact value of a is : ' || a );  
dbms_output.put_line('Exact value of b is : ' || b );  
END;  
/
```

```
SQL> @nestedifthenelse  
Value of a is 100 and b is 200  
Exact value of a is : 100  
Exact value of b is : 200  
  
PL/SQL procedure successfully completed.
```

9. List the keywords that are a required part of an IF statement.

IF condition

THEN statement

END IF

PL/SQL Assignment 5

10. Write a PL/SQL block to find the total monthly salary paid by the company for a given department number from the employees table. Display a message indicating whether the total salary is greater than or less than \$19,000. Test your block twice using the Administration department (department_id = 10) and the IT department (department_id = 60). The IT department should be greater than \$19,000, while the Administration department's total should be less than \$19,000.

```
SQL> CREATE TABLE employees(SALARY INTEGER NOT NULL,department_id INTEGER NOT NULL PRIMARY KEY);
Table created.
```

```
SQL> INSERT INTO employees VALUES (230000,80);
```

```
1 row created.
```

```
SQL> INSERT INTO employees VALUES (13000,40);
```

```
1 row created.
```

```
SQL> select * from employees;
```

SALARY	DEPARTMENT_ID
230000	80
13000	40

DECLARE

v_salary employees.salary%TYPE;

BEGIN

SELECT SUM(salary)

INTO v_salary FROM employees WHERE department_id = 60;

DBMS_OUTPUT.PUT_LINE('Total salary is : '||v_salary);

IF v_salary < 19000 THEN DBMS_OUTPUT.PUT_LINE('Total salary in IT department is smaller than \$19K ');

ELSE DBMS_OUTPUT.PUT_LINE('Total salary in IT department is greater than \$19K ');

END IF;

END;

```
SQL> edit dept1;
```

```
SQL> @dept1;
```

```
11 /
```

```
Total salary is : 230000
```

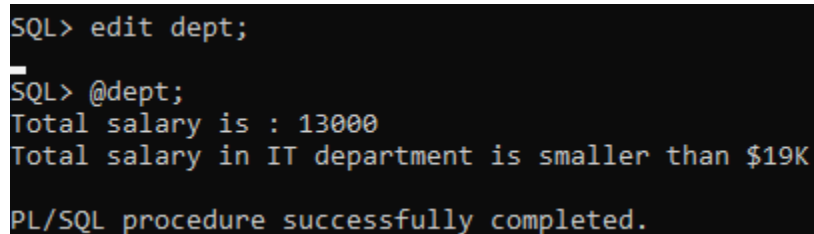
```
Total salary in IT department is greater than $19K
```

```
PL/SQL procedure successfully completed.
```

PL/SQL Assignment 5

11. What happens if we use the Marketing department (department_id=20) in the previous script?

```
DECLARE
v_salary employees.salary%TYPE;
BEGIN
SELECT SUM(salary)
INTO v_salary FROM employees WHERE department_id = 40;
DBMS_OUTPUT.PUT_LINE('Total salary is : '||v_salary);
IF v_salary < 19000 THEN DBMS_OUTPUT.PUT_LINE('Total salary in IT department is
smaller than $19K ');
ELSE DBMS_OUTPUT.PUT_LINE('Total salary in IT department is greater than $19K
');
END IF;
END;
/
```



```
SQL> edit dept;
_
SQL> @dept;
Total salary is : 13000
Total salary in IT department is smaller than $19K

PL/SQL procedure successfully completed.
```

12. Alter the PL/SQL code to include an ELSIF to handle this situation.

```
>DECLARE
v_salary employees.salary%TYPE;
BEGIN
SELECT SUM(salary)
INTO v_salary FROM employees WHERE department_id = 40;
DBMS_OUTPUT.PUT_LINE('Total salary is : '||v_salary);
IF v_salary < 19000 THEN
DBMS_OUTPUT.PUT_LINE('Total salary in Marketing department is smaller than $19K
');
ELSIF v_salary = 1900 THEN
DBMS_OUTPUT.PUT_LINE('Total salary in Marketing department is equal $19K ');
ELSE
DBMS_OUTPUT.PUT_LINE('Total salary in Marketing department is greater than $19K
');
END IF;
END;
```


PL/SQL Assignment 5

```
SQL> edit deptelseif;
_
SQL> @deptelseif;
Total salary is : 13000
Total salary in Marketing department is smaller than $19K

PL/SQL procedure successfully completed.
```

13. Write a PL/SQL block to select the number of countries using a supplied currency name. If the number of countries is greater than 20, display “More than 20 countries”. If the number of countries is between 10 and 20, display “Between 10 and 20 countries”. If the number of countries is less than 10, display “Fewer than 10 countries”. Use a CASE statement.

```
DECLARE
v_no_of_countries countries.currency_code%TYPE;
BEGIN
SELECT COUNT(currency_code) INTO v_no_of_countries
FROM countries WHERE currency_code =001;
CASE
WHEN v_no_of_countries <10 THEN
DBMS_OUTPUT.PUT_LINE('Fewer than 10 countries');
WHEN v_no_of_countries BETWEEN 10 AND 20
THEN DBMS_OUTPUT.PUT_LINE('Between 10 and 20 countries');
ELSE DBMS_OUTPUT.PUT_LINE('More than 20 countries');
END CASE;
DBMS_OUTPUT.PUT_LINE('No of countries is : '||v_no_of_countries);
END;
/
```

PL/SQL Assignment 5

```
SQL> create table countries(country_name varchar(3),country_code int,currency_code int);
```

```
Table created.
```

```
SQL> insert into countries(country_name,country_code,currency_code)
2 values('a',10,001);
```

```
1 row created.
```

```
SQL> insert into countries(country_name,country_code,currency_code)
2 values('b',11,001);
```

```
1 row created.
```

```
SQL> insert into countries(country_name,country_code,currency_code)
2 values('c',12,001);
```

```
1 row created.
```

```
SQL> insert into countries(country_name,country_code,currency_code)
2 values('d',13,001);
```

```
1 row created.
```

```
SQL> insert into countries(country_name,country_code,currency_code)
2 values('e',14,001);
```

```
1 row created.
```

```
SQL> insert into countries(country_name,country_code,currency_code)
2 values('f',15,001);
```

```
1 row created.
```

```
SQL> select * from countries;
```

COU	COUNTRY_CODE	CURRENCY_CODE
a	10	1
b	11	1
c	12	1
d	13	1
e	14	1
f	15	1

```
SQL> edit countrycode;
```

```
SQL> @countrycode;
Fewer than 10 countries
No of countries is : 6
```

```
PL/SQL procedure successfully completed.
```

PL/SQL Assignment 5

14. Write a PL/SQL block to display the country_id and country_name values from the WF_COUNTRIES table for country_id whose values range from 1 through 3. Use a basic loop. Increment a variable from 1 through 3. Use an IF statement to test your variable and EXIT the loop after you have displayed the first 3 countries.

```
DECLARE
v_countryid countries.country_code%TYPE := 10;
v_countryname countries.country_name%TYPE;
v_counter NUMBER(2) := 10;
BEGIN
LOOP
SELECT country_code INTO v_countryid FROM countries
WHERE country_code = v_counter;
v_counter := v_counter + 1;
DBMS_OUTPUT.PUT_LINE('Countries are: '||v_countryid);
EXIT
WHEN v_counter > 13;
END LOOP;
END;
/
```

```
SQL> edit countrycount;
```

```
SQL> @countrycount;
```

```
Countries are: 10
```

```
Countries are: 11
```

```
Countries are: 12
```

```
Countries are: 13
```

```
PL/SQL procedure successfully completed.
```

PL/SQL Assignment 5

15. Write a PL/SQL block to produce a list of available vehicle license plate numbers. These numbers must be in the following format: NN-MMM, where NN is between 60 and 65, and MMM is between 100 and 110. Use nested FOR loops. The outer loop should choose numbers between 60 and 65. The inner loop should choose numbers between 100 and 110, and concatenate the two numbers together.

```
SQL> begin
  2  FOR v_out IN 60..65 LOOP
  3  FOR v_inn IN 100..110 LOOP
  4  DBMS_OUTPUT.PUT_LINE(v_out||'-'||v_inn);
  5  END LOOP;
  6  END LOOP;
  7  END;
  8  /
60->100
60->101
60->102
60->103
60->104
60->105
60->106
60->107
60->108
60->109
60->110
61->100
61->101
61->102
61->103
61->104
61->105
61->106
61->107
61->108
61->109
61->110
62->100
62->101
62->102
62->103
62->104
62->105
62->106
62->107
62->108
62->109
62->110
63->100
63->101
63->102
63->103
63->104
63->105
63->106
63->107
63->108
63->109
63->110
64->100
64->101
64->102
64->103
64->104
64->105
64->106
64->107
64->108
64->109
64->110
65->100
65->101
65->102
65->103
65->104
65->105
65->106
65->107
65->108
65->109
65->110
PL/SQL procedure successfully completed.
```