PL/SQL

Control Structures

In addition to SQL commands,PL/SQL can also process data usin flow of statements.the flow of control statements are classified into the following categories.

- Conditional control -Branching
- Iterative control - looping
- Sequential control

BRANCHING in PL/SQL:

Sequence of statements can be executed on satisfying certain condition .

If statements are being used and different forms of if are:

1.Simple IF

2.ELSIF

3.ELSE IF

SIMPLE IF:

Syntax:

```
IF condition THEN
   statement1;
   statement2;
END IF;
```

IF-THEN-ELSE STATEMENT:

Syntax:

```
IF condition THEN
   statement1;
ELSE
   statement2;
END IF;
```

ELSIF STATEMENTS:

Syntax:

```
IF condition1 THEN
  statement1;
```

```
ELSIF condition2 THEN
  statement2;
ELSIF condition3 THEN
  statement3;
ELSE
  statementn;
END IF;
```

NESTED IF :

Syntax:

```
IF condition THEN
   statement1;
ELSE
 IF condition THEN
   statement2;
 ELSE
   statement3;
 END IF;
END IF;
ELSE
 statement3;
END IF;
```

SELECTION IN PL/SQL(Sequential Controls)

SIMPLE CASE

Syntax:

```
CASE SELECTOR
   WHEN Expr1 THEN statement1;
   WHEN Expr2 THEN statement2;
:
```

FOR LOOP

Syntax:

```
FOR counter IN [REVERSE]
   LowerBound..UpperBound
LOOP
statement1;
statement2;
END LOOP;
```

PROGRAM 1

Write a PL/SQL block to calculate the incentive of an employee whose ID is 110.

```
Set Serveroutput ON;
Declare
   v_salary NUMBER;
   v_incentive NUMBER;
BEGIN
   Select salary INTO v_salary From employees where
employee_id = 110;
   IF v_salary >= 50000 Then
      v_incentive := v_salary * 0.10;
   ELSEIF v_salary >= 30000 Then
      v_incentive := v_salary * 0.07;
   ELSE
      v_incentive := v_salary * 0.05;
   END IF;
   DBMS_OUTPUT.PUT_LINE ('Incentive : ' ||v_incentive);
END;
/
```

PROGRAM 2

Write a PL/SQL block to show an invalid case-insensitive reference to a quoted and without quoted user-defined identifier.

```
Set Serveroutput ON;
Declare
   "Name" Varchar2(20) := 'Kyrah';
Begin
   DBMS_OUTPUT.PUT_LINE(Name);
End;
/
```

PROGRAM 3

Write a PL/SQL block to adjust the salary of the employee whose ID 122.

Sample table: employees

```
Set server output ON;

Declare
    v_old_salary employees.salary%Type;
BEGIN
    Select salary into v_old_salary
    From employees
    Where employee_id = 122;

UPDATE employees
Set salary = v_old_salary * 1.10
where employee_id = 122;
DBMS_OUTPUT.PUT_LINE('Salary adjusted successfully for
employee ID 122');
DBMS_OUTPUT.PUT_LINE('Old Salary: ' || v_old_salary);
DBMS_OUTPUT.PUT_LINE('New Salary: ' || (v.old_salary*1.10));

EXCEPTION
    WHEN No_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Employee ID 122
                not found.');

END;
```

PROGRAM 4

Write a PL/SQL block to create a procedure using the "IS [NOT] NULL Operator" and show AND operator returns TRUE if and only if both operands are TRUE.

```
Set Server output ON ;
Create or Replace Procedure check_employee_status IS
    v_name Varchar2(30) := 'Vignesh';
    v_bonus Number := 1000;
BEGIN
    IF (v_name IS NOT NULL) AND (v_bonus > 500) THEN
    DBMS_OUTPUT.PUT_LINE ('Both conditions are True - AND
returns True.');
    ELSE
        DBMS_OUTPUT.PUT_LINE ('Atleast one condition is false
-AND returns FALSE.');
        END IF ;
    END;
```

PROGRAM 5

Write a PL/SQL block to describe the usage of LIKE operator including wildcard characters and escape character.

```
Set server output on;
DECLARE
    v_name Varchar2(70);
BEGIN
    v_name:='S_Kumar';
  IF v_name Like 'S%' THEN
      Dbms_Output.Put_Line('Name starts with S');
  END IF;
  IF v_name Like 'S_%' THEN
    Dbms_Output.Put_Line('Second character can be anything
                          after S_');
    END IF;
  IF vname LIKE 'S\_K%' Escape '\' THEN
    Dbms_Output.Put_Line('Method name containing literal
        underscore (_) using Escape');
    END IF;
END;
```

20

PROGRAM 6

Write a PL/SQL program to arrange the number of two variable in such a way that the small number will store in num_small variable and large number will store in num_large variable.

```
Set server output on;
Declare a Number:= 30; b Number := 60 ; sNumber, l Number;
Begin
   If a< b Then s:=a ; l:=b; Else s:=b; l:=a; End If;
   Dbms_Output.Put_Line ('Small = '||s|| ' Large = '||l);
End;
/
```

PROGRAM 7

Write a PL/SQL procedure to calculate the incentive on a target achieved and display the message either the record updated or not.

```
Set Server output ON;
Create or replace Procedure inc (p_id Number, p_t Number) Is
BEGIN
    Update employees SET Salary = salary + (p_t * 0.05) where
    employee_id = p_id;
    IF SQL%Rowcount > 0 Then DBMS_OUTPUT.PUT_LINE ('Record
    Updated');
    ELSE DBMS_OUTPUT.PUT_LINE ('No record updated');
    ENDIF;
END;
/
BEGIN inc (110, 20000);
END;
/
```

PROGRAM 8

Write a PL/SQL procedure to calculate incentive achieved according to the specific sale limit.

```
Create or replace procedure calc_incentive (Sales Number) IS
  incentive number;
BEGIN
  If sales >= 100000 Then
      incentive := sales *0.10;
  ELSE IF  sales >= 50000 Then
      incentive := sales * 0.05;
  ELSE
      incentive := 0;
  END IF;
  DBMS_OUTPUT.PUT_LINE ('Incentive:' || incentive);
END;
```

PROGRAM 9

Write a PL/SQL program to count number of employees in department 50 and check whether this department have any vacancies or not. There are 45 vacancies in this department.

```
DECLARE
    emp_count NUMBER ;
    vacancies NUMBER := 45;
BEGIN
    Select Count (*) INTO emp_count FROM employees
                    Where department_id = 50;
    DBMS_OUTPUT.PUT_LINE ('Employees' || emp_count);
IF emp_count < vacancies THEN
    DBMS_OUTPUT.PUT_LINE ('Vacancies: ' || vacancies - emp_count);
ELSE
    DBMS_OUTPUT.PUT_LINE ('No vacancies');
END IF;
END;
```

(17)

PROGRAM 10

Write a PL/SQL program to count number of employees in a specific department and check whether this department have any vacancies or not. If any vacancies, how many vacancies are in that department.

```
DECLARE
    dept_id Number := &dept_id;
    emp_count number;
    vacancies number;
Begin
    Select count(*) into emp_count from employees where
                department_id = dept_id;

    Select vacancies into vacancies from departments where
                department_id = dept_id;

    DBMS_OUTPUT.PUT_LINE ('Employees: ' || emp_count);
    IF emp_count < vacancies Then
        DBMS_OUTPUT.PUT_LINE ('Vacancies : ' || vacancies-emp_count);
    ELSE
        DBMS_OUTPUT.PUT_LINE ('No vacancies');
    END IF;
END;
```

PROGRAM 11

Write a PL/SQL program to display the employee IDs, names, job titles, hire dates, and salaries of all employees.

```
BEGIN
 FOR emp IN (
    SELECT employee_id, employee_name, job_title, hire_date, salary
    FROM employees
 ) LOOP
 DBMS_OUTPUT.PUT_LINE (emp.employee_id || ' ' || emp.employee_name || ' ' ||
   emp.job_title || ' ' || emp.hire_date || ' ' || emp.salary);
  END LOOP;
END;
```

PROGRAM 12

Write a PL/SQL program to display the employee IDs, names, and department names of all employees.

```
BEGIN
  FOR emp IN (
    SELECT employee_id, employee_name, department_name
    FROM employees
  ) LOOP
  DBMS_OUTPUT.PUT_LINE (emp.employee_id || ' ' || employee_name || ' ' ||
                        emp.department_name);

  END LOOP;
END;
```

PROGRAM 13

Write a PL/SQL program to display the job IDs, titles, and minimum salaries of all jobs.

```
BEGIN
  For job IN (
      Select job_id, job_title, min_salary
      From jobs
  )Loop
      Dbms_Output.Put_Line (job.job_id || '' ||job_title || '' ||
      job.min_salary);

  END Loop;

  END;
```

PROGRAM 14

Write a PL/SQL program to display the employee IDs, names, and job history start dates of all employees.

```
BEGIN
  FOR emp IN (
      Select employee_id ,employee_name, job_start_date
      From employees
  ) LOOP
    DBMS_OUTPUT.PUT_LINE (emp.employee_id || ' ' || emp.employee_name || ' '
  || emp.job_start_date );
  END LOOP;
END;
```

PROGRAM 15

Write a PL/SQL program to display the employee IDs, names, and job history end dates of all employees.

```
Begin
  For emp IN (
      Select employee_id, employee_name, job_end_date
      From employees
  ) Loop
  DBMS_Output.Put_Line (emp.employee_id || '' || emp employee_name || '' ||
  emp.job_end_date);
  End Loop;
End;
```

Evaluation Procedure	Marks awarded
PL/SQL Procedure(5)	5
Program/Execution (5)	5
Viva(5)	5
Total (15)	15
Faculty Signature	

(20)