DBMS-PL SQL Exercise

```
DECLARE
  incentive   NUMBER(8,2);
salary   NUMBER(8,2);
incentivepct NUMBER(8,2);
BEGIN
  SELECT SALARY INTO salary
  FROM employees
  WHERE EMPLOYEE_ID = 110;
  SELECT COMMISSION_PCT INTO incentivepct
   FROM employees
   WHERE EMPLOYEE_ID=110;
Incentive := salary * incentivepct;
DBMS_OUTPUT.PUT_LINE('Incentive  = ' || TO_CHAR(incentive));
END;
```

Q1.
```
DECLARE
  "WELCOME" varchar2(10) := 'welcome';
BEGIN
  DBMS_Output.Put_Line("Welcome");
END;
DECLARE
  WELCOME varchar2(10) := 'welcome';
BEGIN
  DBMS_Output.Put_Line("Welcome");
END;
```

```
Q2.DECLARE
  "NUMBER" varchar2(25) := 'UPPERCASE';
  "Number" varchar2(25) := 'Proper Case';
  "number" varchar2(25) := 'lowercase';
BEGIN
  DBMS_Output.Put_Line("NUMBER");
  DBMS_Output.Put_Line("Number");
  DBMS_Output.Put_Line("number");
END;
```

```
Q3. DECLARE
  "WORLD" varchar2(20) := 'world';
  "DECLARE" varchar2(20) := 'declare';
BEGIN
  DBMS_Output.Put_Line(World);
  DBMS_Output.Put_Line(DECLARE);
end;
```

```
Q4.DECLARE
  "WORLD" varchar2(10) := 'world';
  "DECLARE" varchar2(10) := 'declare';
BEGIN
  DBMS_Output.Put_Line(World);
  DBMS_Output.Put_Line("Declare");
end;
```

```
Q5.DECLARE
 Pi_Value             NUMBER := 3.1415926; -- pi is set to 3.1415926 : this is single line comment
BEGIN

  /* PI is initialized above.
PI Value is printed here: : this is multi line comment*/
 DBMS_OUTPUT.PUT_LINE('The value of pi  is: ' || Pi_Value);
END;

Q6.DECLARE
  item_number       NUMBER(5);
  item_name         VARCHAR2(20);
  stock_yn          BOOLEAN;
  item_rate        NUMBER(8,2);
  item_description  VARCHAR2(40);
  maximum_deposit     CONSTANT REAL    := 25000.00;
  min_no_of_days CONSTANT INTEGER := 75;
  nominee_yn     CONSTANT BOOLEAN := FALSE;
  employee_no          INTEGER := 0;
  pi     CONSTANT REAL := 3.14159;
  radius          REAL := 10;
BEGIN
  NULL;
END;
```

```
Q7.DECLARE
  var_a INTEGER;
  var_b REAL;
BEGIN
  var_a:=5;
  var_b:=10.25;
  DBMS_OUTPUT.PUT_LINE('In the Outer Block');
  DBMS_OUTPUT.PUT_LINE('var_a = ' || var_a); -- var_a  is INTEGER
  DBMS_OUTPUT.PUT_LINE('var_b = ' || var_b); -- var_b is REAL
               DECLARE
               var_a CHAR;  -- Scope of var_a have changed into CHAR and beginning from here
               var_c REAL;       -- Scope of var_c is REAL
              BEGIN
                var_a:='C';
                var_c:=15.50;
              DBMS_OUTPUT.PUT_LINE('In the First sub-Block');
              DBMS_OUTPUT.PUT_LINE('var_a = ' || var_a);
              DBMS_OUTPUT.PUT_LINE('var_b = ' || var_b);
             DBMS_OUTPUT.PUT_LINE('var_c = ' || var_c);
              END;
DBMS_OUTPUT.PUT_LINE('At the end in the  Outer-Block');
DBMS_OUTPUT.PUT_LINE('var_a = ' || var_a); -- var_a  is INTEGER
DBMS_OUTPUT.PUT_LINE('var_b = ' || var_b); -- var_b is REAL
END;
```

```
Q8.DECLARE "WELCOME" varchar2(10) := 'welcome';
 BEGIN DBMS_Output.Put_Line(Welcome);
END;

DECLARE WELCOME varchar2(10) := 'welcome';
BEGIN DBMS_Output.Put_Line(Welcome);
END;

Q9.DECLARE
  salary_of_emp  NUMBER(8,2);
   BEGIN
  SELECT salary INTO salary_of_emp
  FROM employees
  WHERE employee_id = 122;
  Empsal:= 100;
   empsal := empsal + addless;
  DBMS_OUTPUT.PUT_LINE('New Salary: ' || empsal);
END;

Q10. DECLARE
  salary_of_emp  NUMBER(8,2);
PROCEDURE approx_salary ( emp   NUMBER, empsal IN OUT NUMBER, addless   NUMBER)
IS
  BEGIN
    empsal := empsal + addless;
  END;
```

```
BEGIN
  SELECT salary INTO salary_of_emp
  FROM employees
  WHERE employee_id = 122;
  DBMS_OUTPUT.PUT_LINE
   ('Before invoking procedure, salary_of_emp: ' || salary_of_emp);
  approx_salary (100, salary_of_emp, 1000);
  DBMS_OUTPUT.PUT_LINE
   ('After invoking procedure, salary_of_emp: ' || salary_of_emp);
END;
```

```
Q11.DECLARE
  salary      NUMBER := 40000;
  commission  NUMBER := 0.15;
BEGIN
   DBMS_OUTPUT.PUT_LINE('8 + 20 / 4 = ' || (8 + 20 / 4));
  DBMS_OUTPUT.PUT_LINE('20 / 4 + 8 = ' || (20 / 4 + 8));
  DBMS_OUTPUT.PUT_LINE('7 + 9 / 3 = ' || (7 + 9 / 3));
  DBMS_OUTPUT.PUT_LINE('(7 + 9) / 3 = ' || ((7 + 9) / 3));
  DBMS_OUTPUT.PUT_LINE('30 + (30 / 6 + (15 - 8)) = '|| (30 + (30 / 6 + (15 - 8))));
  DBMS_OUTPUT.PUT_LINE('(salary*0.08)+(commission*0.12) = '||((salary * 0.08) + (commission*0.12)));
  DBMS_OUTPUT.PUT_LINE('salary * 0.08 + commission * 0.12 = '|| (salary * 0.08 + commission * 0.12));
END;
```

Q12.
```
CREATE OR REPLACE PROCEDURE pri_bool(boo_name VARCHAR2,boo_val BOOLEAN) IS
BEGIN
  IF boo_val IS NULL THEN
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = NULL');
  ELSIF boo_val = TRUE THEN
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = TRUE');
  ELSE
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = FALSE');
  END IF;
END;

DECLARE
  PROCEDURE pri_m_and_n (m  BOOLEAN,n  BOOLEAN) IS
  BEGIN
   pri_bool ('m', m);
   pri_bool ('n', n);
   pri_bool ('m AND n', m AND n);
END pri_m_and_n;

BEGIN
DBMS_OUTPUT.PUT_LINE('------------- FOR m and n both FALSE ---------------------');
pri_m_and_n (FALSE, FALSE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m TRUE AND n FALSE ---------------------');
pri_m_and_n (TRUE, FALSE);
```

```
DBMS_OUTPUT.PUT_LINE('------------- FOR m FALSE AND n TRUE ---------------------');
pri_m_and_n (FALSE, TRUE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m TRUE AND n TRUE ---------------------');
pri_m_and_n (TRUE, TRUE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m TRUE AND n NULL ---------------------');
pri_m_and_n (TRUE, NULL);
DBMS_OUTPUT.PUT_LINE('------------- FOR m FALSE AND n NULL---------------------');
pri_m_and_n (FALSE, NULL);
DBMS_OUTPUT.PUT_LINE('------------- FOR m NULL AND n TRUE ---------------------');
pri_m_and_n (NULL, TRUE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m NULL AND n FALSE ---------------------');
pri_m_and_n (NULL, FALSE);
END;
```

```
Q13.CREATE OR REPLACE PROCEDURE pri_bool( boo_name VARCHAR2, boo_val BOOLEAN) IS
BEGIN
  IF boo_val IS NULL THEN
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = NULL');
  ELSIF boo_val = TRUE THEN
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = TRUE');
  ELSE
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = FALSE');
  END IF;
END;
```

```
DECLARE
  PROCEDURE pri_m_or_n (m  BOOLEAN,  n  BOOLEAN ) IS
  BEGIN
   pri_bool ('m', m);
   pri_bool ('n', n);
   pri_bool ('m OR n', m OR n);
END pri_m_or_n;

BEGIN
DBMS_OUTPUT.PUT_LINE('------------- FOR m OR n both FALSE ---------------------');
pri_m_or_n (FALSE, FALSE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m TRUE OR n FALSE ---------------------');
pri_m_or_n (TRUE, FALSE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m FALSE OR n TRUE ---------------------');
pri_m_or_n (FALSE, TRUE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m TRUE OR n TRUE ---------------------');
pri_m_or_n (TRUE, TRUE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m TRUE OR n NULL ---------------------');
pri_m_or_n (TRUE, NULL);
DBMS_OUTPUT.PUT_LINE('------------- FOR m FALSE OR n NULL---------------------');
pri_m_or_n (FALSE, NULL);
DBMS_OUTPUT.PUT_LINE('------------- FOR m NULL OR n TRUE ---------------------');
pri_m_or_n (NULL, TRUE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m NULL OR n FALSE ---------------------');
pri_m_or_n (NULL, FALSE);
END;
```

```
Q14.CREATE OR REPLACE PROCEDURE pri_bool(boo_name    VARCHAR2, boo_val
BOOLEAN) IS
BEGIN
  IF boo_val IS NULL THEN
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = NULL');
  ELSIF boo_val = TRUE THEN
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = TRUE');
  ELSE
    DBMS_OUTPUT.PUT_LINE( boo_name || ' = FALSE');
  END IF;
END;

DECLARE
  PROCEDURE pri_not_m ( m  BOOLEAN) IS
  BEGIN
    pri_bool ('m', m);
    pri_bool ('NOT m', NOT m);
  END pri_not_m;

BEGIN
DBMS_OUTPUT.PUT_LINE('------------- FOR m TRUE ---------------------');
  pri_not_m (TRUE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m FALSE ---------------------');
  pri_not_m (FALSE);
DBMS_OUTPUT.PUT_LINE('------------- FOR m NULL ---------------------');
  pri_not_m (NULL);
END;
```

```
Q15.DECLARE
  m NUMBER := 7;
  n NUMBER := NULL;
  o NUMBER := NULL;
  p NUMBER := NULL;
  q    INTEGER := 4;
  r    INTEGER := 9;
large INTEGER;

 BEGIN
  IF m != n THEN  -- yields NULL, not TRUE
    DBMS_OUTPUT.PUT_LINE('m != n');
  ELSIF m = n THEN -- also yields NULL
    DBMS_OUTPUT.PUT_LINE('m = n');
  ELSE
    DBMS_OUTPUT.PUT_LINE ('Can not say whether m and n are equal or not.');
  END IF;
   IF o = p THEN  -- yields NULL, not TRUE
    DBMS_OUTPUT.PUT_LINE('o = p');
  ELSIF o != p THEN
    DBMS_OUTPUT.PUT_LINE('o != p');
  ELSE
    DBMS_OUTPUT.PUT_LINE('Can not say whether two NULLs are equal');
  END IF;
```

```
IF (q > r)
    THEN large  := q;
    ELSE large  := r;  FALSE or NULL
DBMS_OUTPUT.PUT_LINE('The value of large : '||large);
  END IF;
  IF NOT (q > r)
    THEN large  := r;
    ELSE large  := q;
DBMS_OUTPUT.PUT_LINE('The value of large : '||large);
  END IF;
END;
```

```
Q16.DECLARE
  PROCEDURE pat_match (   test_string   VARCHAR2,  pattern       VARCHAR2  ) IS
  BEGIN
    IF test_string LIKE pattern THEN
      DBMS_OUTPUT.PUT_LINE ('TRUE');
    ELSE
      DBMS_OUTPUT.PUT_LINE ('FALSE');
    END IF;
  END;
BEGIN
  pat_match('Blweate', 'B%a_e');
  pat_match('Blweate', 'B%A_E');
END;
```

SUBMITTED BY: VINAY KARTHIK MARADI BALACHANDRA (AZF2YA)