Practical 4

FUNCTIONS:

1. Write a PL/SQL block to find the factorial of a number without recursion using function.

Code:

```
SET SERVEROUTPUT ON;
CREATE OR REPLACE FUNCTION FACTORIAL_02 (N in number)
RETURN NUMBER IS
RES NUMBER:=1;
TEMP NUMBER:=N;
BEGIN
WHILE TEMP>0 LOOP
    RES:=TEMP*RES;
    TEMP:=TEMP-1;
END LOOP;
RETURN RES;
END;
/
DECLARE
N NUMBER;
BEGIN
N := &N;
DBMS_OUTPUT.PUT_LINE('FACTORIAL OF '|| N);
DBMS_OUTPUT.PUT_LINE(FACTORIAL_02(N));
END;
/
```

Output:

```
SQL> @ E:\HammadDBMS\PLSQL_FUNCTION\func_1_CALL.SQL
Enter value for n: 4
old   5: N := &N;
new   5: N := 4;
```

```
ENTER A NUMBER TO FIND FACTORIAL
24

PL/SQL procedure successfully completed.

SQL> /
Enter value for n: 5
old   5: N := &N;
new   5: N := 5;
ENTER A NUMBER TO FIND FACTORIAL
120
```

2. Write a PL/SQL block to find the factorial of a number using recursive function.

Code:

```
DECLARE
   N NUMBER;
   RES NUMBER;
CREATE OR REPLACE FUNCTION FACT_R_02(X NUMBER)
RETURN NUMBER
IS
   F NUMBER;
BEGIN
   IF X=0 THEN
      F := 1;
   ELSE
      F := X * FACT_R_02(X-1);
   END IF;
RETURN F;
END FACT_R_02;

BEGIN
   N:= &N;
   RES := FACT_R_02(N);
   dbms_output.put_line(' Factorial '|| N || ' is ' || RES);
END;
/
```

Output:

```
SQL> @ E:\HammadDBMS\PLSQL_FUNCTION\func_2.SQL
Enter value for n: 4
old  18:    N:= &N;
new  18:    N:= 4;
Factorial 4 is 24

PL/SQL procedure successfully completed.

SQL> /
Enter value for n: 5
old  18:    N:= &N;
new  18:    N:= 5;
Factorial 5 is 120

PL/SQL procedure successfully completed.
```

3. Write a PL/SQL block to demonstrate Function Overloading.

Code:

```
SET SERVEROUTPUT ON;
DECLARE
   N1 NUMBER;
   N2 NUMBER;
   N3 NUMBER;
   RES1 NUMBER;
FUNCTION ADD_02(X NUMBER, Y NUMBER)
RETURN NUMBER
IS
R NUMBER;
BEGIN
   R := X + Y;
RETURN R;
END ADD_02;

FUNCTION ADD_02(X NUMBER, Y NUMBER,Z NUMBER)
```

```
RETURN NUMBER
IS
R NUMBER;
BEGIN
   R := X + Y + Z;
RETURN R;
END ADD_02;

BEGIN
   N1 := &N;
   N2 := &N;
   N3 := &N;
   RES1 := ADD_02(N1,N2);
   dbms_output.put_line(' ADDITION OF FIRST TWO NUMBERS ' || RES1);
   RES1 := ADD_02(N1,N2,N3);
   dbms_output.put_line(' ADDITION OF THREE NUMBERS ' || RES1);
END;
/
```

Output:

```
SQL> @ E:\HammadDBMS\PLSQL_FUNCTION\FUNC_3.SQL
Enter value for n: 3
old  25:    N1 := &N;
new  25:    N1 := 3;
Enter value for n: 4
old  26:    N2 := &N;
new  26:    N2 := 4;
Enter value for n: 5
old  27:    N3 := &N;
new  27:    N3 := 5;
ADDITION OF FIRST TWO NUMBERS 7
ADDITION OF THREE NUMBERS 12

PL/SQL procedure successfully completed.
```

4. Write a PL/SQL block to display name of depositor having highest bank amount using function.

Code:

```
SET SERVEROUTPUT ON;
DECLARE
FUNCTION HIGH_02
return VARCHAR as
      NAME VARCHAR(20);
      CURSOR c1 IS SELECT * FROM DEPOSIT_02;
      r c1%ROWTYPE;
      MAX DEPOSIT_02.AMOUNT%TYPE:=0;

BEGIN
      MAX:=0;
      FOR r IN c1 LOOP
            MAX:=r.amount;
            NAME:=r.cname;
      END LOOP;
      RETURN NAME;
END HIGH_02;

BEGIN
   dbms_output.put_line(' NAME OF THE HIGHEST DEPOSITOR IS ' ||
HIGH_02());
END;
/
```

Output:

```
SQL> @ E:\HammadDBMS\PLSQL_FUNCTION\FUNC_4.SQL

Function created.

NAME OF THE HIGHEST DEPOSITOR IS NAREN

PL/SQL procedure successfully completed.
```

5. Write a PL/SQL block to display number of depositors using function.

Code:

```
SET SERVEROUTPUT ON;
DECLARE
CNT NUMBER;
```

```
FUNCTION COUNT_02
return VARCHAR as
      NAME VARCHAR(20);
      CURSOR c1 IS SELECT * FROM DEPOSIT_02;
      r c1%ROWTYPE;

BEGIN
CNT:=0;
OPEN c1;
FETCH c1 into r;
WHILE(c1%found)LOOP
    CNT:=CNT+1;
    FETCH c1 INTO r;
END LOOP;
RETURN CNT;
END COUNT_02;

BEGIN
   dbms_output.put_line(' NUMBER OF DEPOSITORS : ' || COUNT_02());
END;
/
```

Output:

```
SQL> @ E:\HammadDBMS\PLSQL_FUNCTION\FUNC_5.SQL

Function created.

NUMBER OF DEPOSITORS : 9

PL/SQL procedure successfully completed.
```

6. Write a PL/SQL block to display number of depositors whose name starts with A using function.

Code:

```
SET SERVEROUTPUT ON;
```

```
DECLARE
CNT NUMBER;
FUNCTION COUNT2_02
return VARCHAR as
      NAME VARCHAR(20);
      CURSOR c1 IS SELECT * FROM DEPOSIT_02 WHERE CNAME LIKE 'A%';
      r c1%ROWTYPE;
BEGIN
CNT:=0;
OPEN c1;
FETCH c1 into r;
WHILE(c1%found)LOOP
    CNT:=CNT+1;
    FETCH c1 INTO r;
END LOOP;
RETURN CNT;
END COUNT2  _02;

BEGIN
   dbms_output.put_line(' NUMBER OF DEPOSITOR WHOSE NAME STARTS WITH A : '
|| COUNT2_02());
END;
/
```

Output:

```
SQL> @ E:\HammadDBMS\PLSQL_FUNCTION\FUNC_6.SQL

Function created.

NUMBER OF DEPOSITOR WHOSE NAME STARTS WITH A : 1

PL/SQL procedure successfully completed.
```

7. Write a PL/SQL block to display branch name with fifth highest amount in deposit using function.

Code:

```
SET SERVEROUTPUT ON;
DECLARE
FUNCTION FIFTH_02
RETURN VARCHAR AS
      NAME VARCHAR(20);
      CURSOR C1 IS SELECT BNAME BM FROM DEPOSIT_02 ORDER BY AMOUNT DESC;
      R C1%ROWTYPE;
BEGIN
FOR R IN C1 LOOP
    IF(C1%rowcount = 5)THEN
        NAME := R.BM;
    END IF;
END LOOP;
RETURN NAME;
END FIFTH_02;

BEGIN
   dbms_output.put_line('NAME OF THE BRANCH WITH FIFTH HIGHEST AMOUNT : '
|| FIFTH_02());
END;
/
```

Output:

```
SQL> @E:\HammadDBMS\PLSQL_FUNCTION\FUNC_7.SQL
NAME OF THE BRANCH WITH FIFTH HIGHEST AMOUNT : M.G.ROAD

PL/SQL procedure successfully completed.

SQL> SPOOL OFF;
```

PROCEDURES:

1. Write a PL/SQL block to display depositor name and date whose opening account is after 12/3/1998 using procedure.

Code:

```
create or replace procedure opening_after_date
is
cursor c is SELECT cname, adate FROM deposit_02 WHERE adate > '12-MAR-98';
is_found_rec boolean := false;
not_found exception;
BEGIN
    for i in c loop
        is_found_rec := true;
        dbms_output.put_line(i.cname || ' ' || i.adate);
    end loop;

    if not is_found_rec THEN
        raise not_found;
    end if;

exception
    when not_found then
        dbms_output.put_line('No depositors joined after 12/3/1998.');
end opening_after_date;
/

DECLARE
BEGIN
    opening_after_date();
end;
/
```

Output:

```
SQL> E:\HammadDBMS\PLSQL_PROCEDURES\PRO_1.SQL
```

```
Procedure created.

No depositors joined after 12/3/1998.

PL/SQL procedure successfully completed.
```

2. Write a PL/SQL block to display depositor details of a specific name using procedure.

Code:

```
SERVEROUTPUT ON;
DECLARE
CUST VARCHAR2(20);
PROCEDURE PRO_2(CUSTNAME IN VARCHAR2) IS
CURSOR C1 IS SELECT * FROM DEPOSIT_02 WHERE CNAME = CUSTNAME;
BEGIN
FOR R IN C1 LOOP
    DBMS_OUTPUT.PUT_LINE('ACCOUNT NUMBER : '|| R.AC_NO);
    DBMS_OUTPUT.PUT_LINE('NAME : '|| R.CNAME);
    DBMS_OUTPUT.PUT_LINE('BRANCH : '|| R.BNAME);
    DBMS_OUTPUT.PUT_LINE('AMOUNT : '|| R.AMOUNT);
    DBMS_OUTPUT.PUT_LINE('CREATE DATE : '|| R.ADATE);
END LOOP;
END PRO_2;

BEGIN
    CUST := '&CUST';
    PRO_2(CUST);
END;
/
```

Output:

```
SQL>  @E:\HammadDBMS\PLSQL_PROCEDURES\PRO_2.SQL
Enter value for cust: ANIL
```

```
old  16:     CUST := '&CUST';
new  16:     CUST := 'ANIL';
ACCOUNT NUMBER : 100
NAME : ANIL
BRANCH : VRCE
AMOUNT : 1120
CREATE DATE : 01-MAR-95

PL/SQL procedure successfully completed.

SQL> SPOOL OFF;
```

3. Write a PL/SQL block to update (replace) all the names of customers with first character capital and others in lower case using procedure.

Code:

```
create or replace procedure capitalize_first_letter
is
BEGIN
    UPDATE customer_02 SET cname = INITCAP(cname);
    UPDATE deposit_02 SET cname = INITCAP(cname);
    UPDATE borrow_02 SET cname = INITCAP(cname);
end capitalize_first_letter;
/

DECLARE
BEGIN
    capitalize_first_letter();
    dbms_output.put_line('First letters of all names are not capital, and
the rest are small.');
end;
/
```

Output:

```
SQL> ALTER TABLE deposit_02 DISABLE CONSTRAINT FK_Customer;
```

```
Table altered.

SQL> ALTER TABLE borrow_02 DISABLE CONSTRAINT FK_Customer1;

Table altered.

SQL> select * from customer_02;

CNAME                CITY
-------------------- --------------------
ANIL                 KOLKATA
SUNIL                DELHI
MEHUL                BARODA
MANDAR               PATNA
MADHURI              NAGPUR
PRAMOD               NAGPUR
SANDIP               SURAT
SHIVANI              MUMBAI
KRANTI               MUMBAI
NAREN                MUMBAI

10 rows selected.

SQL>  @E:\HammadDBMS\PLSQL_PROCEDURES\PRO_3.SQL

Procedure created.

First letters of all names are not capital, and the rest are small.

PL/SQL procedure successfully completed.

SQL> select * from customer_02;

CNAME                CITY
-------------------- --------------------
Anil                 KOLKATA
Sunil                DELHI
Mehul                BARODA
Mandar               PATNA
Madhuri              NAGPUR
Pramod               NAGPUR
Sandip               SURAT
```

```
Shivani             MUMBAI
Kranti              MUMBAI
Naren               MUMBAI

10 rows selected.
```

4. Write a PL/SQL block to display amount in the format 99,999,99 using procedure.

Code:

```
create or replace procedure change_number_format
is
cursor c is SELECT amount FROM deposit_02;
BEGIN
    for i in c loop
        dbms_output.put_line(to_char(i.amount, '99,999,99'));
    end loop;
end change_number_format;
/

DECLARE
BEGIN
    change_number_format();
end;
/
```

Output:

```
SQL>  @E:\HammadDBMS\PLSQL_PROCEDURES\PRO_4.SQL

Procedure created.
```

```
10,00
50,00
35,00
12,00
30,00
20,00
10,00
50,00
70,00

PL/SQL procedure successfully completed.
```

5. Write a PL/SQL block to display average amount of depositors using procedure.

Code:

```
create or replace procedure display_average_amount(average out number)
is
BEGIN
SELECT AVG(amount) INTO average FROM deposit_02;
end display_average_amount;
/

DECLARE
average number(8,2);
BEGIN
display_average_amount(average);
dbms_output.put_line('The average amount is ' || average);
end;
/
```

Output:

```
SQL>  @E:\HammadDBMS\PLSQL_PROCEDURES\PRO_5.SQL

Procedure created.
```

```
The average amount is 3188.89

PL/SQL procedure successfully completed.
```

PACKAGES:

1. Create a package which consists of a function of addition of two numbers and a procedure for multiplication of two numbers.

Code:

```
create or replace package arithmetic_02 as
function addition(a in number, b in number) return number;
procedure multiplication(a in number, b in number, c out number);
end arithmetic_02;
/

create or replace package body arithmetic_02 as
function addition_02(a in number, b in number)
return number is
begin
    return a + b;
end addition_02;

procedure multiplication_02(a in number, b in number, c out number)
is
begin
    c := a * b;
end multiplication_02;
end arithmetic_02;
/

declare
a number;
b number;
c number;
begin
    a := &a;
    b := &b;
    dbms_output.put_line('Sum of the numbers: ' ||
arithmetic_02.addition_02(a, b));
```

```
    arithmetic_02.multiplication_02(a, b, c);
    dbms_output.put_line('Product of the numbers: ' || c);
end;
/
```

Output:

```
SQL>  @E:\HammadDBMS\PLSQL_PACKAGES\PACK_1.SQL

Package created.


Package body created.

Enter value for a: 3
old   6:     a := &a;
new   6:     a := 3;
Enter value for b: 4
old   7:     b := &b;
new   7:     b := 4;
Sum of the numbers: 7
Product of the numbers: 12

PL/SQL procedure successfully completed.
```

2. Create a package which consists of a function of factorial of a number and a procedure of factorial of a two number.

Code:

```
create or replace package factorial_02 AS
function factorial_one(a in number) return number;
procedure factorial_two(a in number, b in number, fact1 out number, fact2
out number);
end factorial_02;
/

create or replace package body factorial_02 as
function factorial_one(a in number)
return number is
fact number := 1;
```

```
begin
    for i in 1 .. a loop
        fact := fact * i;
    end loop;
    return fact;
end factorial_one;

procedure factorial_two(a in number, b in number, fact1 out number, fact2
out number)
is
begin
    fact1 := 1;
    fact2 := 1;
    for i in 1 .. a loop
        fact1 := fact1 * i;
    end loop;
    for i in 1 .. b loop
        fact2 := fact2 * i;
    end loop;
end factorial_two;
end factorial_02;
/

DECLARE
    a number;
    b number;
    c number;
    fact1 number;
    fact2 number;
BEGIN
    a := &a;
    b := &b;
    c := &c;
    dbms_output.put_line('Factorial of ' || a || ': ' ||
factorial_02.factorial_one(a));
    factorial_02.factorial_two(b, c, fact1, fact2);
    dbms_output.put_line('Factorial of ' || b || ': ' || fact1);
```

```
    dbms_output.put_line('Factorial of ' || c || ': ' || fact2);
end;
/
```

Output:

```
SQL>  @E:\HammadDBMS\PLSQL_PACKAGES\PACK_2.SQL

Package created.


Package body created.

Enter value for a: 5
old   8:     a := &a;
new   8:     a := 5;
Enter value for b: 4
old   9:     b := &b;
new   9:     b := 4;
Enter value for c: 6
old  10:     c := &c;
new  10:     c := 6;
Factorial of 5: 120
Factorial of 4: 24
Factorial of 6: 720

PL/SQL procedure successfully completed.
```

3. Create a package which consists of function to display name of depositor having highest bank amount and a procedure to display depositor name and date whose opening account is after 12/3/1998 using procedure.

Code:

```
create or replace package highest_02 as
function highest_amt return varchar2;
procedure opening_after_date;

end highest_02;
/
```

```
create or replace package body highest_02 as
function highest_amt
return varchar2 is
n varchar2(20);
BEGIN
    SELECT cname into n FROM deposit_02 WHERE amount in(SELECT MAX(amount)
FROM deposit_02);
    return n;
end highest_amt;

procedure opening_after_date
is
cursor c is SELECT cname, adate FROM deposit_02 WHERE adate > '12-MAR-98';
is_found_rec boolean := false;
not_found exception;
BEGIN
    for i in c loop
        is_found_rec := true;
        dbms_output.put_line(i.cname || ' ' || i.adate);
    end loop;

    if not is_found_rec THEN
        raise not_found;
    end if;

exception
    when not_found then
        dbms_output.put_line('No depositors joined after 12/3/1998.');
end opening_after_date;
end highest_02;
/

DECLARE
name varchar2(20);
BEGIN
    name:= highest_02.highest_amt();
```

```
    dbms_output.put_line(name || ' has the highest amount deposited in the
bank.');
    highest_02.opening_after_date();
end;
/
```

Output:

```
SQL>  @E:\HammadDBMS\PLSQL_PACKAGES\PACK_3.SQL

Package created.


Package body created.

NAREN has the highest amount deposited in the bank.
No depositors joined after 12/3/1998.

PL/SQL procedure successfully completed.
```

4. Create a package which consists of a procedure to display amount in the format 99,999,99 and a function to display average amount of depositors.

Code:

```
create or replace package average_02 AS
procedure change_number_format;
function display_average_amount return number;
end average_02;
/

create or replace package body average_02 AS

procedure change_number_format
is
cursor c is SELECT amount FROM deposit_02;
BEGIN
    for i in c loop
        dbms_output.put_line(to_char(i.amount, '99,999,99'));
```

```
    end loop;
end change_number_format;

function display_average_amount
return number
is
average number(8, 2);
BEGIN
    SELECT AVG(amount) INTO average FROM deposit_02;
    return average;
end display_average_amount;
end average_02;
/

DECLARE
BEGIN
    dbms_output.put_line('Average is ' ||
average_02.display_average_amount());
    dbms_output.put_line('Amounts with changed format:');
    average_02.change_number_format();
end;
/
```

Output:

```
SQL>  @E:\HammadDBMS\PLSQL_PACKAGES\PACK_5.SQL

Package created.


Package body created.

Average is 3188.89
Amounts with changed format:
10,00
50,00
35,00
12,00
30,00
```

```
20,00
10,00
50,00
70,00

PL/SQL procedure successfully completed.
```