Exercise # 4

I. Write the exceptions block for all the below procedures/functions which you have written in the Exercise #3.

1) Write a procedure to fetch data from table SALES for a given parameter orderid and display the data.

2) Write a procedure which does the following operations

   - Fetch data from table SALES for a given parameter orderid and display the data.
   - Return the number of rows(using OUT parameter) in the SALES table for that sales date (get sales date from the about operation)

3) Write a function which accepts 2 numbers N1 and N2 and returns the power of N1 to N2. (Example: If I pass values 10 and 3, the output should be 1000)

4) Write a function to display the number of rows in the SALES table for a given sales date.

II. Write a user defined exception for function 3 which displays an exception saying "Invalid Number" or "Number must be less than 100", if it meets the below conditions

   - If N1 or N2 is null or zero
   - If N1 or N2 is greater than 100.

Answers

I.

1)

```
CREATE OR REPLACE PROCEDURE FETCH_SALES (S_ORDERID NUMBER)
AS
L_DATE SALES.SALES_DATE%TYPE;
L_ORDERID SALES.ORDER_ID%TYPE;
L_PRODUCTID SALES.PRODUCT_ID%TYPE;
L_CUSTOMERID SALES.CUSTOMER_ID%TYPE;
L_SALESPERSONID SALES.SALESPERSON_ID%TYPE;
L_QUANTITY SALES.QUANTITY%TYPE;
L_UNITPRICE SALES.UNIT_PRICE%TYPE;
L_SALESAMOUNT SALES.SALES_AMOUNT%TYPE;
L_TAXAMOUNT SALES.TAX_AMOUNT%TYPE;
L_TOTALAMOUNT SALES.TOTAL_AMOUNT%TYPE;
BEGIN

SELECT SALES_DATE, ORDER_ID, PRODUCT_ID, CUSTOMER_ID, SALESPERSON_ID, QUANTITY,
UNIT_PRICE, SALES_AMOUNT, TAX_AMOUNT, TOTAL_AMOUNT
INTO
L_DATE, L_ORDERID, L_PRODUCTID, L_CUSTOMERID, L_SALESPERSONID, L_QUANTITY, L_UNITPRICE,
L_SALESAMOUNT, L_TAXAMOUNT, L_TOTALAMOUNT
FROM SALES
WHERE ORDER_ID = S_ORDERID;

   DBMS_OUTPUT.PUT_LINE (L_DATE);
   DBMS_OUTPUT.PUT_LINE (L_ORDERID);
   DBMS_OUTPUT.PUT_LINE (L_PRODUCTID);
   DBMS_OUTPUT.PUT_LINE (L_CUSTOMERID);
   DBMS_OUTPUT.PUT_LINE (L_SALESPERSONID);
   DBMS_OUTPUT.PUT_LINE (L_QUANTITY);
   DBMS_OUTPUT.PUT_LINE (L_UNITPRICE);
   DBMS_OUTPUT.PUT_LINE (L_SALESAMOUNT);
   DBMS_OUTPUT.PUT_LINE (L_TAXAMOUNT);
   DBMS_OUTPUT.PUT_LINE (L_TOTALAMOUNT);
EXCEPTION
   WHEN no_data_found THEN
     dbms_output.put_line('No such Order!');
   WHEN too_many_rows THEN
     dbms_output.put_line('You got more than 1 row!');
   WHEN others THEN
     dbms_output.put_line('Error!');
END;


EXEC FETCH_SALES (1269);
```

2)

```
CREATE OR REPLACE PROCEDURE FETCH_SALES (S_ORDERID IN NUMBER, L_TOTALROWS OUT
NUMBER)
AS
L_DATE SALES.SALES_DATE%TYPE;
L_ORDERID SALES.ORDER_ID%TYPE;
L_PRODUCTID SALES.PRODUCT_ID%TYPE;
L_CUSTOMERID SALES.CUSTOMER_ID%TYPE;
L_SALESPERSONID SALES.SALESPERSON_ID%TYPE;
L_QUANTITY SALES.QUANTITY%TYPE;
L_UNITPRICE SALES.UNIT_PRICE%TYPE;
L_SALESAMOUNT SALES.SALES_AMOUNT%TYPE;
L_TAXAMOUNT SALES.TAX_AMOUNT%TYPE;
L_TOTALAMOUNT SALES.TOTAL_AMOUNT%TYPE;
BEGIN

SELECT SALES_DATE, ORDER_ID, PRODUCT_ID, CUSTOMER_ID, SALESPERSON_ID, QUANTITY,
UNIT_PRICE, SALES_AMOUNT, TAX_AMOUNT, TOTAL_AMOUNT
INTO
L_DATE, L_ORDERID, L_PRODUCTID, L_CUSTOMERID, L_SALESPERSONID, L_QUANTITY, L_UNITPRICE,
L_SALESAMOUNT, L_TAXAMOUNT, L_TOTALAMOUNT
FROM SALES
WHERE ORDER_ID = S_ORDERID;

   DBMS_OUTPUT.PUT_LINE (L_DATE);
   DBMS_OUTPUT.PUT_LINE (L_ORDERID);
   DBMS_OUTPUT.PUT_LINE (L_PRODUCTID);
   DBMS_OUTPUT.PUT_LINE (L_CUSTOMERID);
   DBMS_OUTPUT.PUT_LINE (L_SALESPERSONID);
   DBMS_OUTPUT.PUT_LINE (L_QUANTITY);
   DBMS_OUTPUT.PUT_LINE (L_UNITPRICE);
   DBMS_OUTPUT.PUT_LINE (L_SALESAMOUNT);
   DBMS_OUTPUT.PUT_LINE (L_TAXAMOUNT);
   DBMS_OUTPUT.PUT_LINE (L_TOTALAMOUNT);

SELECT COUNT(1) INTO L_TOTALROWS FROM SALES
WHERE SALES_DATE = L_DATE;

EXCEPTION
   WHEN no_data_found THEN
     dbms_output.put_line('No such Order!');
   WHEN too_many_rows THEN
     dbms_output.put_line('You got more than 1 row!');
   WHEN others THEN
     dbms_output.put_line('Error!');
END;



DECLARE
 TOTAL_ROWS NUMBER;
BEGIN
```

```
 FETCH_SALES (1269, TOTAL_ROWS);
 DBMS_OUTPUT.PUT_LINE ('Total Number of rows: ' || TOTAL_ROWS);
END;
```

3)

```
CREATE OR REPLACE FUNCTION MY_POWER (N1 IN NUMBER, N2 IN NUMBER)
RETURN NUMBER
AS
POWER_VALUE NUMBER:= 1;
BEGIN

FOR LCNTR IN 1..N2
LOOP
  POWER_VALUE := POWER_VALUE * N1;
END LOOP;

RETURN POWER_VALUE;

EXCEPTION
  WHEN others THEN
    dbms_output.put_line('Error!');
END;

SELECT MY_POWER(10,3) FROM DUAL;
```

4)

```
CREATE OR REPLACE FUNCTION GET_COUNT (S_DATE DATE)
RETURN NUMBER
AS
T_ROWS NUMBER;
BEGIN

SELECT COUNT(1) INTO T_ROWS FROM SALES
WHERE SALES_DATE = S_DATE;

RETURN T_ROWS;

EXCEPTION
  WHEN no_data_found THEN
    dbms_output.put_line('No orders for the given date!');
  WHEN others THEN
    dbms_output.put_line('Error!');
END;



SELECT GET_COUNT (TO_DATE ('01-JAN-2015','DD-MON-YYYY')) FROM DUAL
```

II.

```
CREATE OR REPLACE FUNCTION MY_POWER (N1 IN NUMBER, N2 IN NUMBER)
RETURN NUMBER
AS
POWER_VALUE NUMBER:= 1;
EXCEP_ZERO EXCEPTION;
EXCEP_GREAT_100 EXCEPTION;
BEGIN

IF (N1 IS NULL OR N1 = 0 OR N2 IS NULL OR N2 = 0) THEN
    RAISE EXCEP_ZERO;
END IF;

IF N1 > 100 OR  N2 > 100 THEN
    RAISE EXCEP_GREAT_100;
END IF;

FOR LCNTR IN 1..N2
LOOP
  POWER_VALUE := POWER_VALUE * N1;
END LOOP;

RETURN POWER_VALUE;

EXCEPTION
  WHEN EXCEP_ZERO THEN
    dbms_output.put_line('N1  or N2 is null or zero!');
    RETURN 0;
  WHEN EXCEP_GREAT_100 THEN
    dbms_output.put_line('N1  OR N2 is greater than 100!');
    RETURN 0;
  WHEN others THEN
    dbms_output.put_line('Error!');
END;
```