Exercise # 5

Move all the below procedures/functions which you have written in the Exercise #4 to a package (create package specification and then create package body).

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.

Answers

Lets create the package

```
CREATE OR REPLACE PACKAGE SALES_PACKAGE AS

PROCEDURE FETCH_SALES (S_ORDERID NUMBER);

PROCEDURE FETCH_SALES1 (S_ORDERID IN NUMBER, L_TOTALROWS OUT NUMBER);

FUNCTION MY_POWER (N1 IN NUMBER, N2 IN NUMBER)
RETURN NUMBER;

FUNCTION GET_COUNT (S_DATE DATE)
RETURN NUMBER;

END;
```

Now lets create the package body

```
CREATE OR REPLACE PACKAGE BODY SALES_PACKAGE AS

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;

PROCEDURE  FETCH_SALES1 (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;

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;

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;

END;
```