Assignment 2

Objective:

In this assignment, you will create a set of stored procedures and user defined functions using Oracle (PL/SQL). This assignment helps students learn a basic understanding of application development using Oracle database using PL/SQL.

Submission:

Your submission will include:

1- A single .sql file containing the stored procedures

2- A single output file demonstrating that your stored procedures all function correctly

Here are the filenames for the two files you will submit:

A02-<lastname>-<firstname>.sql

A02-<lastname>-<firstname>-output

Make sure you follow proper indentation and comments and exception handling and good coding standards. This is worth marks.

Instruction:

In this assignment, we use the same database that you have been using for the labs and the assignment 1.

Note: For each query in your assignment, make sure you handle the errors and display the proper message including the error code and the error message.

Stored Procedures:

find_customer (customer_id IN NUMBER, found OUT NUMBER);

This procedure has an input parameter to receive the customer ID and an output parameter named found.

This procedure looks for the given customer ID in the database. If the customer exists, it sets the variable *found* to 1. Otherwise, the *found* variable is set to 0.
Make sure to handle the situation when no data was found.

This stored procedure will print out an appropriate message about customer ID being found or not found, using the found variable.

find_product (product_id IN NUMBER, price OUT products.list_price%TYPE);

This procedure has an input parameter to receive the product ID and an output parameter named price.

This procedure looks for the given product ID in the database. If the product exists, it stores the product's list_price in the variable *price*. Otherwise, the *price* variable is set to 0.

Make sure to handle the situation where no data is found.

This stored procedure will print out an appropriate message about the price of the product, using the variables described.

add_order (customer_id IN NUMBER, new_order_id OUT NUMBER)

This procedure has an input parameter to receive the customer ID and an output parameter named new_order_id.

To add a new order for the given customer ID, you need to generate the new order Id. To calculate the new order Id, find the maximum order ID in the orders table and increase it by 1.

This procedure inserts the following values in the orders table:
new_order_id
customer_id (input parameter)
'Shipped' (The value for the order status)
56 (The sales person ID)
sysdate (order date which is the current date)

This stored procedure will print out the information associated with the order being added including the information above and appropriate text summarizing the order information.

add_order_item (orderId IN order_items.order_id%type,
itemId IN order_items.item_id%type,
productId IN order_items.product_id%type,
quantity IN order_items.quantity%type,
price IN order_items.unit_price%type)

This procedure has five IN parameters. It stores the values of these parameters to the table order_items.

This procedure needs to handle errors such as an invalid order ID

display_order (orderId IN NUMBER)

This procedure has an input parameter to receive the order ID and no output parameters.

This procedure will display the order items associated with a particular order ID.

The information to be displayed should include:

- Order ID
- Customer ID

Then should display a row for each item in the order including:

- Item ID
- Product ID
- Quantity
- Price

Then should print a statement showing the total price of the entire order.

There should be an appropriate message for a non-existent order ID

master_proc (task IN NUMBER,
parm1 IN NUMBER)

This procedure is a master procedure for four of the five procedures above.

If task = 1, then, call find_customer(parm1)

If task = 2, then, call find_product(parm1)

If task = 3, then, call add_order(parm1)

If task = 4, then, call display-order(parm1)

In all cases, parm1 is the single input parameter required for the specific function.

You need to handle appropriate error messages here as well.

What to include in your SQL file:

In your SQL file, you should include all of your:

CREATE OR REPLACE PROCEDURE commands (5)

CALL commands (14)

What to include in your OUTPUT file:

In your output file, you should include:

1 – find_customer – with a valid customer ID

2 – find_customer – with an invalid customer ID

3 – find_product – with a valid product ID

4 – find_product – with an invalid product ID

5 – add_order – with a valid customer ID

6 – add_order – with an invalid customer ID

7 – add_order_item – should execute successfully 5 times

8 – add_order_item – should execute with an invalid order ID

9 – display_order – with a valid order ID which has at least 5 order items

10 – display_order – with an invalid order ID

For 1 – 6 and 9 – 10 – your call should be to the master_proc procedure. It will call the actual procedure itself.