Homework 6

Name: Cameron Stark

Problem 1: What is the difference between Procedure and Function?

A Procedure and function do the same actions but a function has the ability to return values to the requested source

Problem 2: Suppose we have two tables:

orders(order_id int, product_id int, quantity int, shipping varchar(255), order_date Date)
products(product_id int, price double, description varchar(255))

Create a **function** "**getPriority**" with order_id as input. Our **getPriority** function will get the priority of the order based on the total cost of the order as follows

Order Total Cost	Priority
>1000	High
500 - 1000	Medium
<500	Low

The order total cost can be calculated as "order total cost = quantity * product price".

```
delimiter ||
CREATE FUNCTION getPriority(quantity double, price double) RETURNS VARCHAR(10)
BEGIN
   DECLARE priority VARCHAR(10);
   DECLARE cost double;
   SET cost = quantity * price;

   IF (cost > 1000) THEN
            SET priority = 'High';
      ELSEIF (cost <= 1000 AND cost >= 500) THEN
            SET priority = 'Medium';
      ELSEIF (cost < 500) THEN
            SET priority = 'Low';
      END IF;

   RETURN (priority);
END
||
```

Problem 3: Suppose we have two tables:

orders(order_id int, product_id int, quantity, **total_cost double**, shipping varchar(255), order_date Date)

products(product_id int, price double, description varchar(255), **on_sale int**)

In the products table, if the product is on sale, then the value for the "**on_sale**" attribute is 1, otherwise the value is 0. To facilitate the order processing, we now add a new attribute "**total_cost**" into our orders table. The value for the total_cost is calculated as

total_cost = quantity * product price * 0.8, if on_sale =1
total_cost = quantity * product price if on_sale =0

Suppose we have 8 orders in our order tables now as

order_id	producit_id	quantity	total_cost	shipping	order_date
1	2	10		test	3-2-2016
2	3	2		test	3-1-2016
3	1	13		test	2-1-2016
4	3	3		test	1-21-2016
5	2	1		test	1-12-2015
6	1	4		test	1-11-2016
7	3	5		test	1-10-2016
8	4	8		test	1-10-2016

Create a **procedure** updateTotalCost to update the "total_cost" attribute for all 8 orders in the orders table. You **must use a loop design** in your procedure.

```
delimiter ||
CREATE PROCEDURE updateTotalCost()
BEGIN
        DECLARE n INT;
   DECLARE i INT;
   DECLARE newPrice DOUBLE;

        DECLARE quan INT;
   DECLARE prod INT;
   DECLARE sale INT;
   DECLARE cost INT;
   SELECT COUNT(*) FROM Orders INTO n;
        SET i = 1;

   WHILE i < n DO
                SELECT quantity FROM orders WHERE order_id == i INTO quan;
      SELECT product_id FROM orders WHERE order_id == i INTO prod;
      SELECT on_sale FROM products WHERE product_id == prod INTO sale;
      SELECT price FROM products WHERE product_id == prod INTO cost;

      IF (sale == 1) THEN
                        SET newPrice = quan * cost * 0.8;
                ELSEIF (sale == 0) THEN
                        SET newPrice = quan * cost;
                END IF

      UPDATE orders SET total_cost = newPrice WHERE order_id == i;
      SET i = i + 1;
        END WHILE;
END;
||
```