PL EXERCISE 7

Create the following 3 tables and insert sample data as shown:

```
mysql> CREATE TABLE Ord_mst (
    ->     Ord_no INT,
    ->     Cust_cd VARCHAR(10),
    ->     Status CHAR(1)
    -> );
Query OK, 0 rows affected (1.86 sec)

mysql>
mysql> INSERT INTO Ord_mst (Ord_no, Cust_cd, Status)
    -> VALUES
    -> (1, 'C1', 'P');
Query OK, 1 row affected (0.01 sec)

mysql> CREATE TABLE Ord_dtl (
    ->     Ord_no INT,
    ->     Prod_cd VARCHAR(10),
    ->     Qty INT
    -> );
Query OK, 0 rows affected (0.17 sec)

mysql>
mysql> INSERT INTO Ord_dtl (Ord_no, Prod_cd, Qty)
    -> VALUES
    -> (1, 'P1', 100),
    -> (1, 'P2', 200);
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> CREATE TABLE Prod_mst (
    ->     Prod_cd VARCHAR(10),
    ->     Prod_name VARCHAR(50),
    ->     Qty_in_stock INT,
    ->     Booked_qty INT
    -> );
Query OK, 0 rows affected (0.13 sec)

mysql>
mysql> INSERT INTO Prod_mst (Prod_cd, Prod_name, Qty_in_stock, Booked_qty)
    -> VALUES
    -> ('P1', 'Floppies', 10000, 1000),
    -> ('P2', 'Printers', 5000, 600),
    -> ('P3', 'Modems', 3000, 200);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

1. Write a Before Insert trigger on Ord_dtl. Anytime a row is inserted in Ord_dtl, the Booked_qty in Prod_mst should be increased accordingly.

```
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER before_insert_ord_dtl
    -> BEFORE INSERT ON Ord_dtl
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Update the booked quantity in Prod_mst
    ->     UPDATE Prod_mst
    ->     SET Booked_qty = Booked_qty + NEW.Qty
    ->     WHERE Prod_cd = NEW.Prod_cd;
    -> END;
    -> //
Query OK, 0 rows affected (0.66 sec)

mysql>
mysql> DELIMITER ;
mysql> SELECT * FROM Prod_mst WHERE Prod_cd = 'P1';
+---------+-----------+--------------+------------+
| Prod_cd | Prod_name | Qty_in_stock | Booked_qty |
+---------+-----------+--------------+------------+
| P1      | Floppies  |        10000 |       1000 |
+---------+-----------+--------------+------------+
1 row in set (0.00 sec)

mysql> INSERT INTO Ord_dtl (Ord_no, Prod_cd, Qty)
    -> VALUES (2, 'P1', 50);
Query OK, 1 row affected (0.13 sec)

mysql> SELECT * FROM Prod_mst WHERE Prod_cd = 'P1';
+---------+-----------+--------------+------------+
| Prod_cd | Prod_name | Qty_in_stock | Booked_qty |
+---------+-----------+--------------+------------+
| P1      | Floppies  |        10000 |       1050 |
+---------+-----------+--------------+------------+
1 row in set (0.00 sec)
```

2. Write a Before Delete trigger on Ord_dtl. Anytime a row is deleted from Ord_dtl, the Booked_qty in Prod_mst should be decreased accordingly.

```
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER trg_before_delete_ord_dtl
    -> BEFORE DELETE ON Ord_dtl
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Decrease the Booked_qty in Prod_mst by the quantity being deleted
    ->     UPDATE Prod_mst
    ->     SET Booked_qty = Booked_qty - OLD.Qty
    ->     WHERE Prod_cd = OLD.Prod_cd;
    -> END;
    -> //
Query OK, 0 rows affected (0.11 sec)

mysql>
mysql> DELIMITER ;
mysql> SELECT * FROM Prod_mst WHERE Prod_cd = 'P1';
+---------+-----------+--------------+------------+
| Prod_cd | Prod_name | Qty_in_stock | Booked_qty |
+---------+-----------+--------------+------------+
| P1      | Floppies  |        10000 |       1050 |
+---------+-----------+--------------+------------+
1 row in set (0.00 sec)

mysql> DELETE FROM Ord_dtl WHERE Ord_no = 2 AND Prod_cd = 'P1';
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM Prod_mst WHERE Prod_cd = 'P1';
+---------+-----------+--------------+------------+
| Prod_cd | Prod_name | Qty_in_stock | Booked_qty |
+---------+-----------+--------------+------------+
| P1      | Floppies  |        10000 |       1000 |
+---------+-----------+--------------+------------+
1 row in set (0.00 sec)
```

3. Write a Before Update of Prod_cd, Qty trigger on Ord_dtl. Anytime the Prod_cd or Qty is updated, the Booked_qty in Prod_mst should be increased/decreased accordingly.

```
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER trg_before_update_ord_dtl
    -> BEFORE UPDATE ON Ord_dtl
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Case 1: If the product code is changed
    ->     IF OLD.Prod_cd != NEW.Prod_cd THEN
    ->         -- Decrease Booked_qty from the old product
    ->         UPDATE Prod_mst
    ->         SET Booked_qty = Booked_qty - OLD.Qty
    ->         WHERE Prod_cd = OLD.Prod_cd;
    ->
    ->         -- Increase Booked_qty in the new product
    ->         UPDATE Prod_mst
    ->         SET Booked_qty = Booked_qty + NEW.Qty
    ->         WHERE Prod_cd = NEW.Prod_cd;
    ->
    ->     -- Case 2: If only the quantity is changed
    ->     ELSEIF OLD.Qty != NEW.Qty THEN
    ->         -- Adjust Booked_qty in the same product
    ->         UPDATE Prod_mst
    ->         SET Booked_qty = Booked_qty + (NEW.Qty - OLD.Qty)
    ->         WHERE Prod_cd = NEW.Prod_cd;
    ->     END IF;
    -> END;
    -> //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> SELECT * FROM Prod_mst;
+---------+-----------+--------------+------------+
| Prod_cd | Prod_name | Qty_in_stock | Booked_qty |
+---------+-----------+--------------+------------+
| P1      | Floppies  |        10000 |       1000 |
| P2      | Printers  |         5000 |        600 |
| P3      | Modems    |         3000 |        200 |
+---------+-----------+--------------+------------+
3 rows in set (0.00 sec)

mysql> UPDATE Ord_dtl
    -> SET Qty = 200
    -> WHERE Ord_no = 1 AND Prod_cd = 'P1';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE Ord_dtl
    -> SET Prod_cd = 'P2', Qty = 150
    -> WHERE Ord_no = 1 AND Prod_cd = 'P1';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> SELECT * FROM Prod_mst;
+---------+-----------+--------------+------------+
| Prod_cd | Prod_name | Qty_in_stock | Booked_qty |
+---------+-----------+--------------+------------+
| P1      | Floppies  |        10000 |        900 |
| P2      | Printers  |         5000 |        750 |
| P3      | Modems    |         3000 |        200 |
+---------+-----------+--------------+------------+
3 rows in set (0.00 sec)
```

4. Write a Before Update of Status trigger on Ord_mst. If the Status is updated from P (Pending) to D (Delivered), the Booked_qty and Qty_in_stock from Prod_mst should be decreased accordingly. If the Status is updated from P (Pending) to C (Cancelled), the details of the order should be deleted from Ord_dtl and corresponding Booked_qty from Prod_mst should be decreased accordingly. (The Before delete trigger on Ord_dtl would automatically decrease the Booked_qty from Prod_mst)

```
mysql> DELIMITER //
mysql>
mysql> CREATE TRIGGER trg_before_update_ord_mst
    -> BEFORE UPDATE ON Ord_mst
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Case 1: Status changed from Pending (P) to Delivered (D)
    ->     IF OLD.Status = 'P' AND NEW.Status = 'D' THEN
    ->         -- Decrease Booked_qty and Qty_in_stock in Prod_mst
    ->         DECLARE qty INT;
    ->         DECLARE prod_cd VARCHAR(10);
    ->
    ->         -- Loop through all the products in the Ord_dtl for the given order
    ->         DECLARE done INT DEFAULT 0;
    ->         DECLARE ord_cursor CURSOR FOR
    ->             SELECT Prod_cd, Qty
    ->             FROM Ord_dtl
    ->             WHERE Ord_no = OLD.Ord_no;
    ->
    ->         -- Handler to exit the cursor loop
    ->         DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->         OPEN ord_cursor;
    ->
    ->         read_loop: LOOP
    ->             FETCH ord_cursor INTO prod_cd, qty;
    ->             IF done THEN
    ->                 LEAVE read_loop;
    ->             END IF;
    ->
    ->             -- Update the Prod_mst table for each product in the order
    ->             UPDATE Prod_mst
    ->             SET Booked_qty = Booked_qty - qty,
    ->                 Qty_in_stock = Qty_in_stock - qty
    ->             WHERE Prod_cd = prod_cd;
    ->         END LOOP;
    ->
    ->         CLOSE ord_cursor;
    ->
    ->     -- Case 2: Status changed from Pending (P) to Cancelled (C)
```

```
   ->     ELSEIF OLD.Status = 'P' AND NEW.Status = 'C' THEN
   ->         -- Delete the corresponding details from Ord_dtl
   ->         DELETE FROM Ord_dtl WHERE Ord_no = OLD.Ord_no;
   ->
   ->         -- Decrease Booked_qty in Prod_mst for each product in the order
   ->         DECLARE done2 INT DEFAULT 0;
   ->         DECLARE ord_cursor2 CURSOR FOR
   ->             SELECT Prod_cd, Qty
   ->             FROM Ord_dtl
   ->             WHERE Ord_no = OLD.Ord_no;
   ->
   ->         -- Handler to exit the cursor loop
   ->         DECLARE CONTINUE HANDLER FOR NOT FOUND SET done2 = 1;
   ->
   ->         OPEN ord_cursor2;
   ->
   ->         read_loop2: LOOP
   ->             FETCH ord_cursor2 INTO prod_cd, qty;
   ->             IF done2 THEN
   ->                 LEAVE read_loop2;
   ->             END IF;
   ->
   ->             -- Update Prod_mst to decrease Booked_qty
   ->             UPDATE Prod_mst
   ->             SET Booked_qty = Booked_qty - qty
   ->             WHERE Prod_cd = prod_cd;
   ->         END LOOP;
   ->
   ->         CLOSE ord_cursor2;
   ->     END IF;
   -> END;
   -> //
   Query OK, 0 rows affected (0.01 sec)
mysql>
mysql> DELIMITER ;
mysql> SELECT * FROM Ord_mst WHERE Ord_no = 1;
+--------+---------+--------+
| Ord_no | Cust_cd | Status |
+--------+---------+--------+
|      1 | C1      | P      |
+--------+---------+--------+
1 row in set (0.00 sec)

mysql> SELECT * FROM Prod_mst;
+---------+-----------+--------------+------------+
| Prod_cd | Prod_name | Qty_in_stock | Booked_qty |
+---------+-----------+--------------+------------+
| P1      | Floppies  |        10000 |        900 |
| P2      | Printers  |         5000 |        750 |
| P3      | Modems    |         3000 |        200 |
+---------+-----------+--------------+------------+
3 rows in set (0.00 sec)

mysql> UPDATE Ord_mst SET Status = 'D' WHERE Ord_no = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> SELECT * FROM Ord_mst WHERE Ord_no = 1;
+--------+---------+--------+
| Ord_no | Cust_cd | Status |
+--------+---------+--------+
|      1 | C1      | D      |
+--------+---------+--------+
1 row in set (0.00 sec)

mysql> SELECT * FROM Prod_mst;
+---------+-----------+--------------+------------+
| Prod_cd | Prod_name | Qty_in_stock | Booked_qty |
+---------+-----------+--------------+------------+
| P1      | Floppies  |        10000 |        900 |
| P2      | Printers  |         5000 |        750 |
| P3      | Modems    |         3000 |        200 |
+---------+-----------+--------------+------------+
3 rows in set (0.00 sec)
```