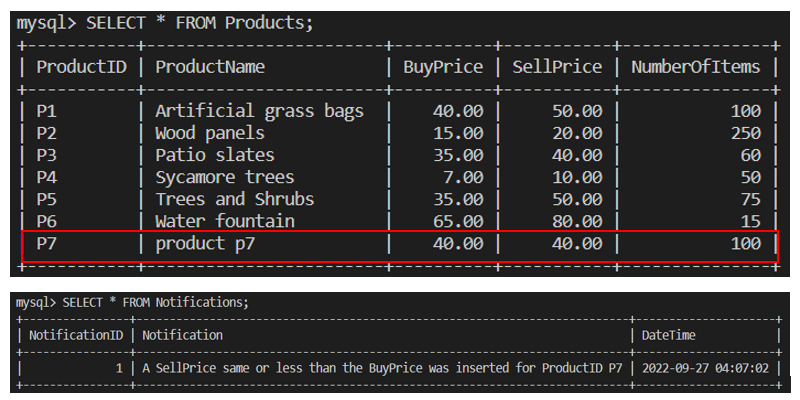
## Task 1 solution:

Create an INSERT trigger called ProductSellPriceInsertCheck. This trigger must check if the SellPrice of the product is less than the BuyPrice after a new product is inserted in the Products table. If this occurs, then a notification must be added to the notifications table to inform the sales department. The sales department can then ensure that the incorrect values were not inserted by mistake.

The notification message should be in the following format: A SellPrice less than the BuyPrice was inserted for ProductID + ProductID.

The expected output result should be the same as that generated by the values for the ProductID P7 in the following screenshot.



DELIMITER //

CREATE TRIGGER ProductSellPriceInsertCheck

AFTER INSERT

ON Products FOR EACH ROW

BEGIN

IF NEW.SellPrice <= NEW.BuyPrice THEN

INSERT INTO Notifications(Notification,DateTime)

VALUES(CONCAT('A SellPrice same or less than the BuyPrice was inserted for ProductID ', NEW.ProductID), NOW());

END IF;

END; //

DELIMITER;

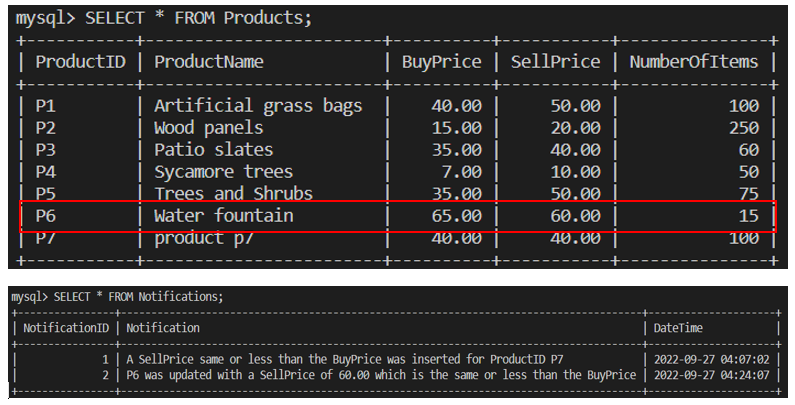
## **Task 2 solution:**

Create an UPDATE trigger called ProductSellPriceUpdateCheck. This trigger must check that products are not updated with a SellPrice that is less than or equal to the BuyPrice.

If this occurs, add a notification to the notifications table for the sales department so they can ensure that product prices were not updated with the incorrect values. This trigger sends a notification to the Notifications table that warns the sales department of the issue.

The notification message should be in the following format: ProductID + 'was updated with a SellPrice of ' + SellPrice + ' which is the same or less than the BuyPrice'

The expected output result should be the same as that generated by the values for the ProductID P6 in the following screenshot.



DELIMITER //

CREATE TRIGGER ProductSellPriceUpdateCheck

AFTER UPDATE

ON Products FOR EACH ROW

BEGIN

IF NEW.SellPrice <= NEW.BuyPrice THEN

INSERT INTO Notifications(Notification,DateTime)

VALUES(CONCAT(NEW.ProductID,' was updated with a SellPrice of ', NEW.SellPrice,' which is the same or less than the BuyPrice'), NOW());

END IF;

END; //

DELIMITER;

## Task 3 solution:

Create a DELETE trigger called NotifyProductDelete. This trigger must insert a notification in the notifications table for the sales department after a product has been deleted from the Products table.

The notification message should be in the following format: 'The product with a ProductID ' + ProductID + ' was deleted'.

The expected output result should be the same as that in the following screenshots.



DELIMITER //

CREATE TRIGGER NotifyProductDelete

AFTER DELETE

ON Products FOR EACH ROW

INSERT INTO Notifications(Notification, DateTime)

VALUES(CONCAT('The product with a ProductID ', OLD.ProductID,' was deleted'), NOW()); //

DELIMITER ;