**Database Lab Take Home Assignment - 5**

**Q1. Construct a table with the following details given below:**

**Inventory (PID, PNAME, QUANTITY, PRICE)**

**Product (ProdID, PNAME, QUANTITY, STATUS, LOG)**

**Purchase (ProductID, BNAME, PNAME, QUANTITY, PRICE, DATE)**

**PID: Primary key for the table Inventory.**

**ProdID: Primary key for the table Product.**

**ProductID, BNAME: Primary keys for the table Purchase.**

**Database Creation**

| **CREATE DATABASE 214161008\_05;** |
| --- |

****

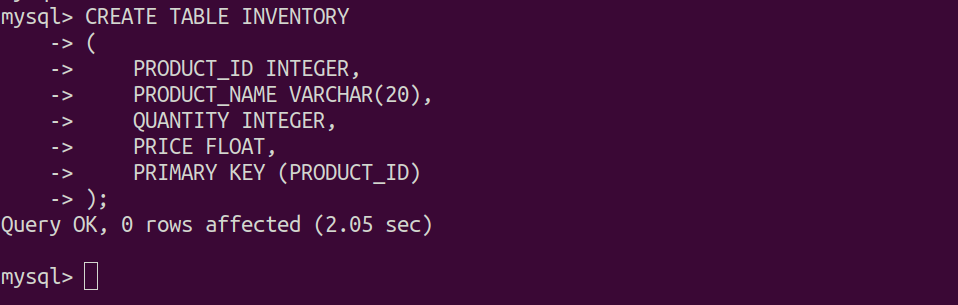
**Select Database**

| **USE 214161008\_05;** |
| --- |

****

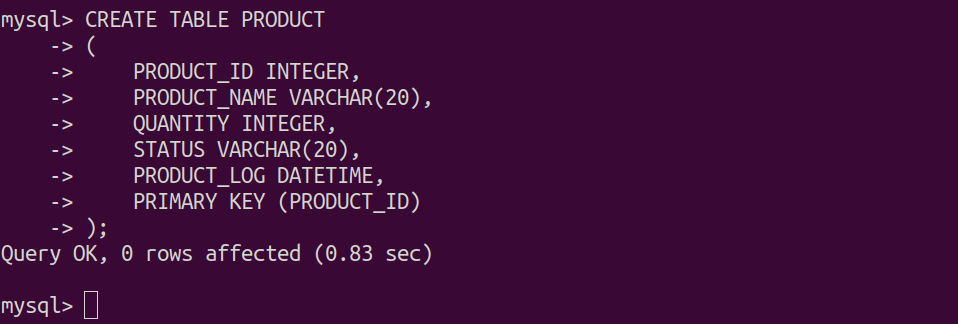
**Inventory table creation**

| **CREATE TABLE INVENTORY (  PRODUCT\_ID INTEGER,  PRODUCT\_NAME VARCHAR(20),  QUANTITY INTEGER,  PRICE FLOAT,  PRIMARY KEY (PRODUCT\_ID) );** |
| --- |

****

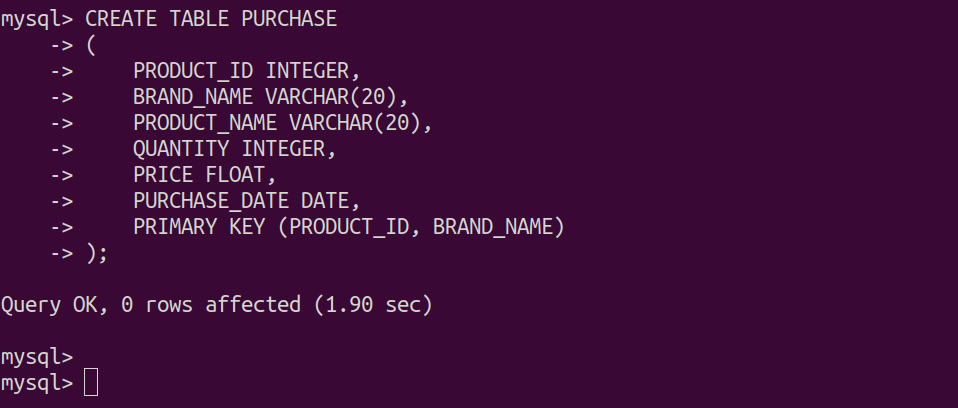
**Product Table creation**

| **CREATE TABLE PRODUCT (  PRODUCT\_ID INTEGER,  PRODUCT\_NAME VARCHAR(20),  QUANTITY INTEGER,  STATUS VARCHAR(20),  PRODUCT\_LOG DATETIME,  PRIMARY KEY (PRODUCT\_ID) );** |
| --- |

****

**Purchase Table Creation**

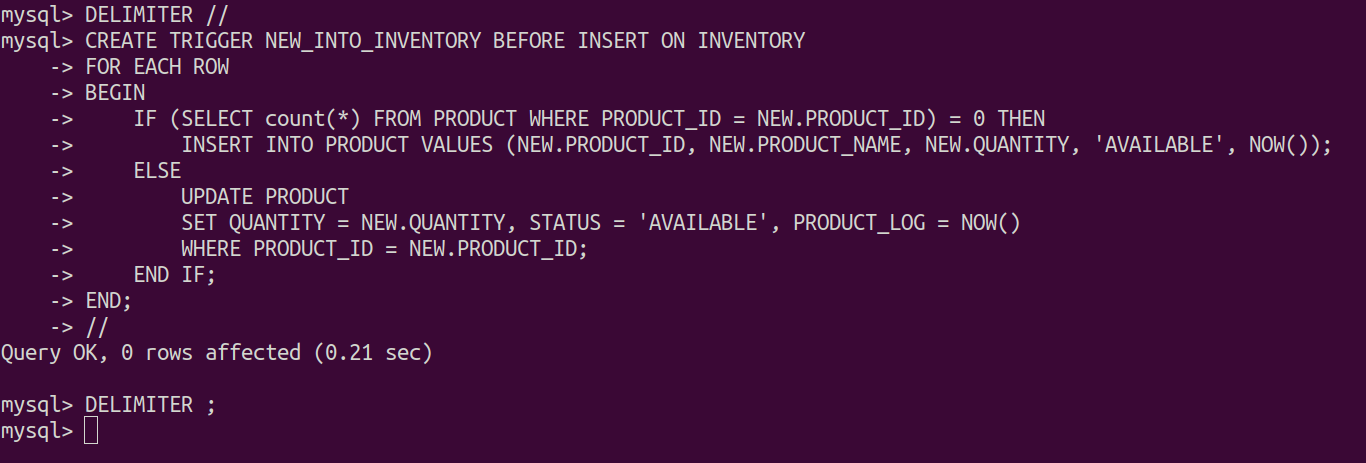
| **CREATE TABLE PURCHASE (  PRODUCT\_ID INTEGER,  BRAND\_NAME VARCHAR(20),  PRODUCT\_NAME VARCHAR(20),  QUANTITY INTEGER,  PRICE FLOAT,  PURCHASE\_DATE DATE,  PRIMARY KEY (PRODUCT\_ID, BRAND\_NAME) );** |
| --- |

****

1. **Write a before insert and before delete trigger on Inventory table to log the products in the Product.**

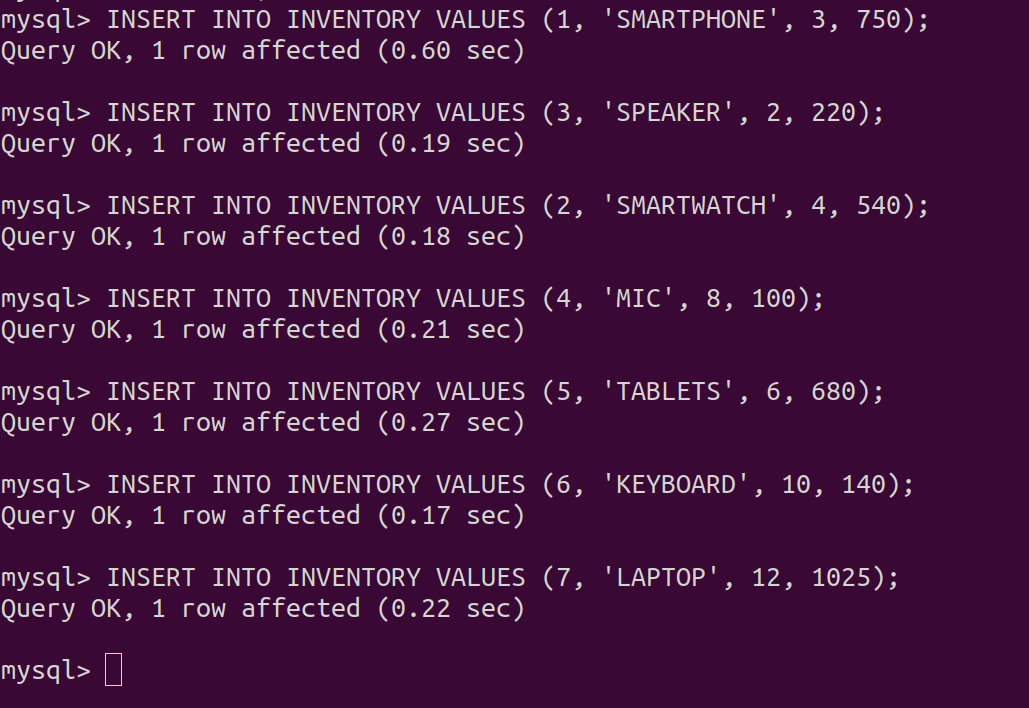
**Before insert trigger**

| **DELIMITER // CREATE TRIGGER NEW\_INTO\_INVENTORY BEFORE INSERT ON INVENTORY FOR EACH ROW  BEGIN  IF (SELECT count(\*) FROM PRODUCT WHERE PRODUCT\_ID = NEW.PRODUCT\_ID) = 0 THEN  INSERT INTO PRODUCT VALUES (NEW.PRODUCT\_ID, NEW.PRODUCT\_NAME, NEW.QUANTITY, 'AVAILABLE', NOW());  ELSE  UPDATE PRODUCT  SET QUANTITY = NEW.QUANTITY, STATUS = 'AVAILABLE', PRODUCT\_LOG = NOW()  WHERE PRODUCT\_ID = NEW.PRODUCT\_ID;  END IF; END; // DELIMITER ;** |
| --- |

****

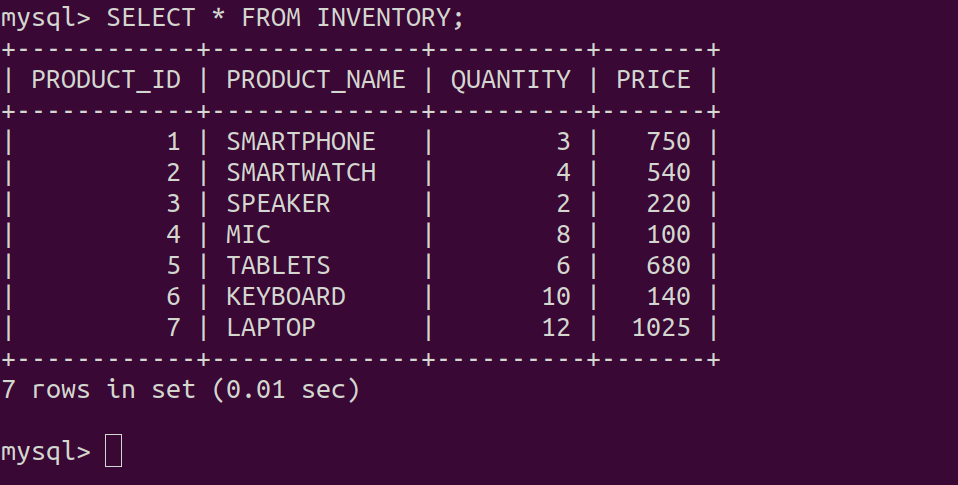
**Inserting Data into table**

| **INSERT INTO INVENTORY VALUES (1, 'SMARTPHONE', 3, 750); INSERT INTO INVENTORY VALUES (3, 'SPEAKER', 2, 220); INSERT INTO INVENTORY VALUES (2, 'SMARTWATCH', 4, 540); INSERT INTO INVENTORY VALUES (4, 'MIC', 8, 100); INSERT INTO INVENTORY VALUES (5, 'TABLETS', 6, 680); INSERT INTO INVENTORY VALUES (6, 'KEYBOARD', 10, 140); INSERT INTO INVENTORY VALUES (7, 'LAPTOP', 12, 1025);** |
| --- |

****

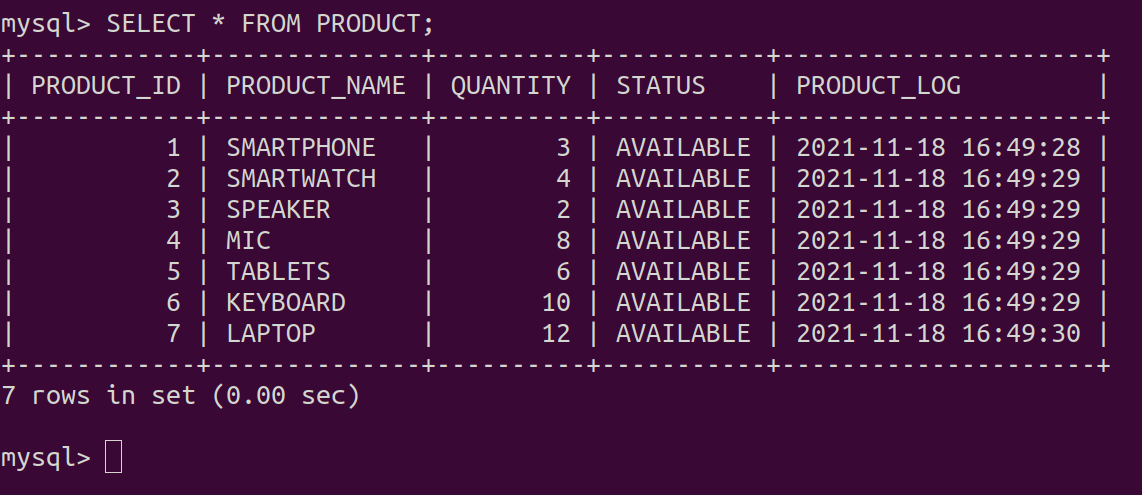
**Updated table data in inventory**

| **SELECT \* FROM INVENTORY;** |
| --- |

****

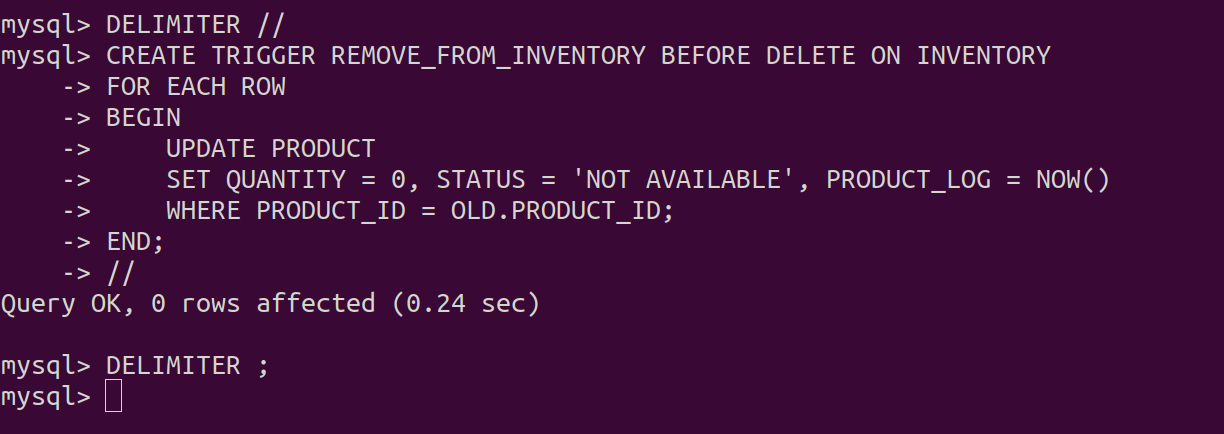
**Updated table data in Product**

| **SELECT \* FROM PRODUCT;** |
| --- |

****

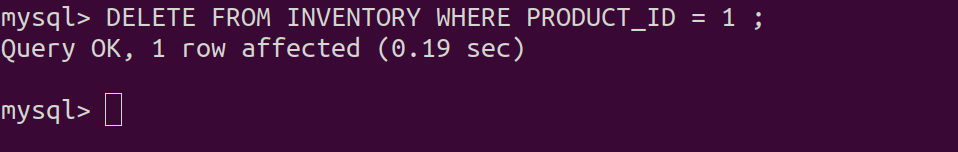
**Before delete Trigger**

| **DELIMITER // CREATE TRIGGER REMOVE\_FROM\_INVENTORY BEFORE DELETE ON INVENTORY FOR EACH ROW BEGIN  UPDATE PRODUCT  SET QUANTITY = 0, STATUS = 'NOT AVAILABLE', PRODUCT\_LOG = NOW()  WHERE PRODUCT\_ID = OLD.PRODUCT\_ID; END; // DELIMITER ;** |
| --- |

****

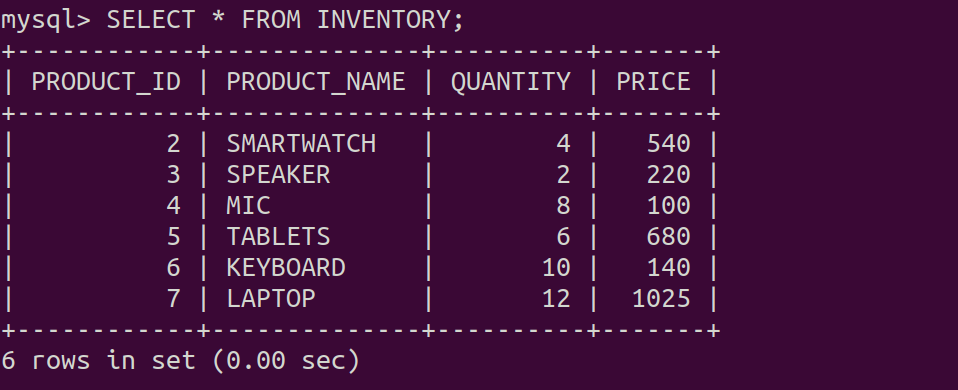
**Delete product having product\_id = 1**

| **DELETE FROM INVENTORY WHERE PRODUCT\_ID = 1 ;** |
| --- |

****

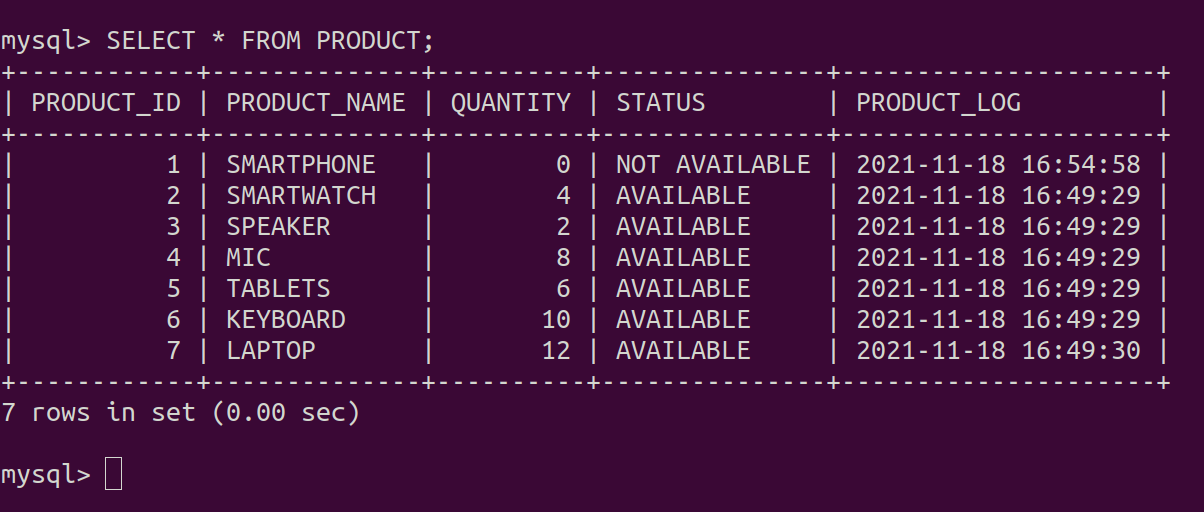
**Updated table data in inventory**

| **SELECT \* FROM INVENTORY;** |
| --- |

****

**Updated table data in Product**

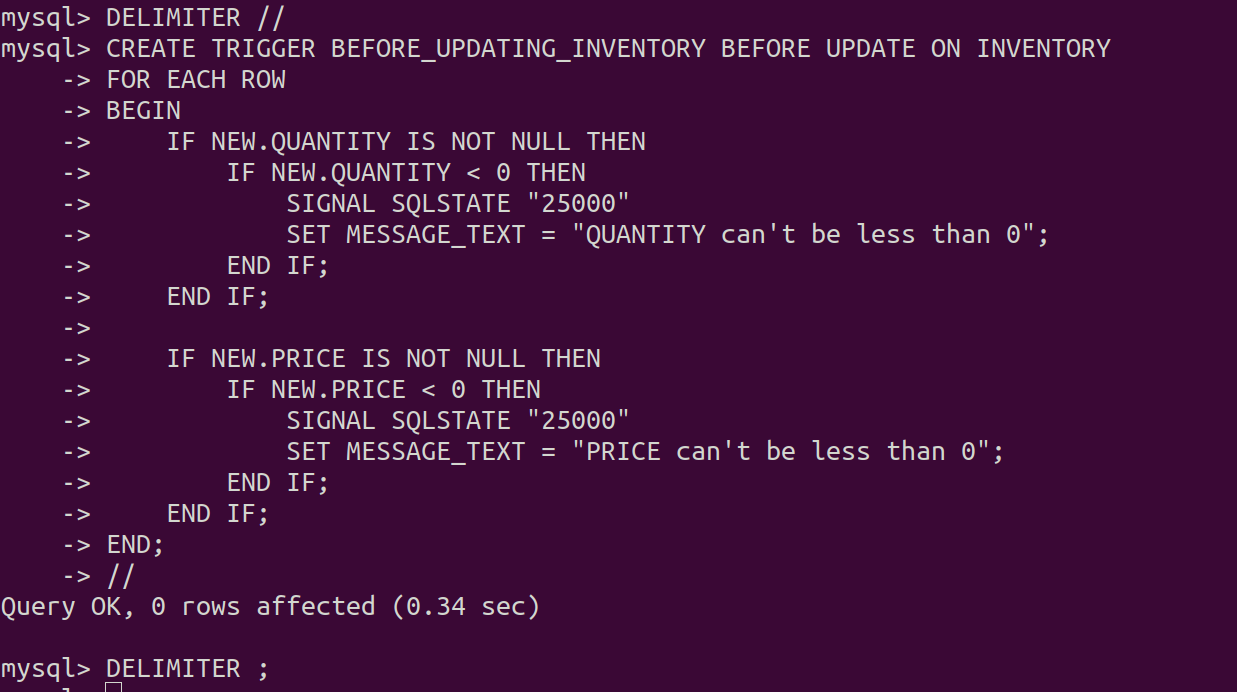
| **SELECT \* FROM PRODUCT;** |
| --- |

****

1. **Write a before update trigger on Inventory table to check that Quantity and Price should not be less than ‘0’.**

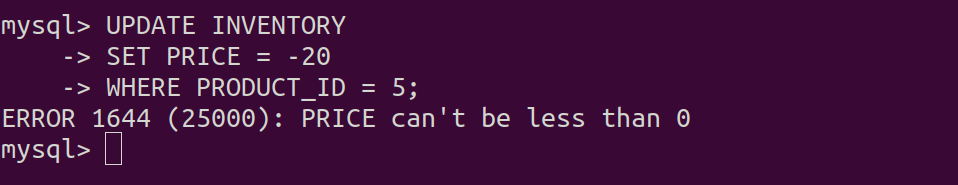
**Before Update trigger on Inventory**

| **DELIMITER // CREATE TRIGGER BEFORE\_UPDATING\_INVENTORY BEFORE UPDATE ON INVENTORY FOR EACH ROW BEGIN  IF NEW.QUANTITY IS NOT NULL THEN  IF NEW.QUANTITY < 0 THEN  SIGNAL SQLSTATE "25000"  SET MESSAGE\_TEXT = "QUANTITY can't be less than 0";  END IF;  END IF;   IF NEW.PRICE IS NOT NULL THEN  IF NEW.PRICE < 0 THEN  SIGNAL SQLSTATE "25000"  SET MESSAGE\_TEXT = "PRICE can't be less than 0";  END IF;  END IF; END; // DELIMITER ;** |
| --- |

****

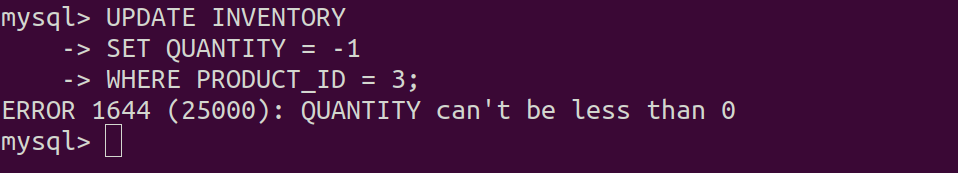
**Update query for the inventory (failed hence price can’t be to less than zero)**

| **UPDATE INVENTORY SET PRICE *= -20* WHERE *PRODUCT\_ID = 5*;** |
| --- |

****

**Update query for the inventory(failed hence quantity can’t be to less than zero)**

| **UPDATE INVENTORY SET QUANTITY *= -1* WHERE *PRODUCT\_ID = 3*;** |
| --- |

****

1. **Write a after delete trigger on Inventory table to remove the products having Quantity less than ‘0’.**

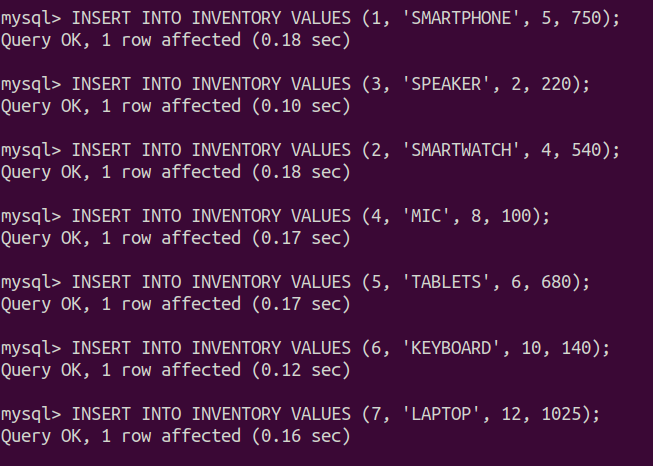
**(Delete all the data for fresh start)**

| **DELETE FROM INVENTORY;** |
| --- |

****

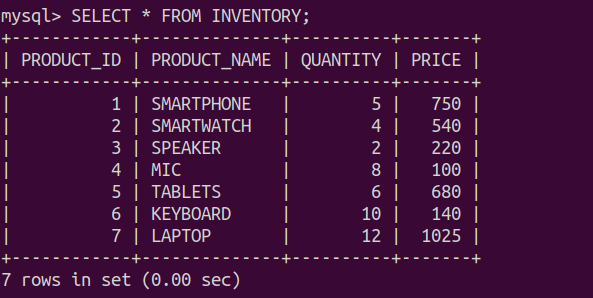
**Insert into inventory table**

| **INSERT INTO INVENTORY VALUES (1, 'SMARTPHONE', 5, 750); INSERT INTO INVENTORY VALUES (3, 'SPEAKER', 2, 220); INSERT INTO INVENTORY VALUES (2, 'SMARTWATCH', 4, 540); INSERT INTO INVENTORY VALUES (4, 'MIC', 8, 100); INSERT INTO INVENTORY VALUES (5, 'TABLETS', 6, 680); INSERT INTO INVENTORY VALUES (6, 'KEYBOARD', 10, 140); INSERT INTO INVENTORY VALUES (7, 'LAPTOP', 12, 1025);** |
| --- |

****

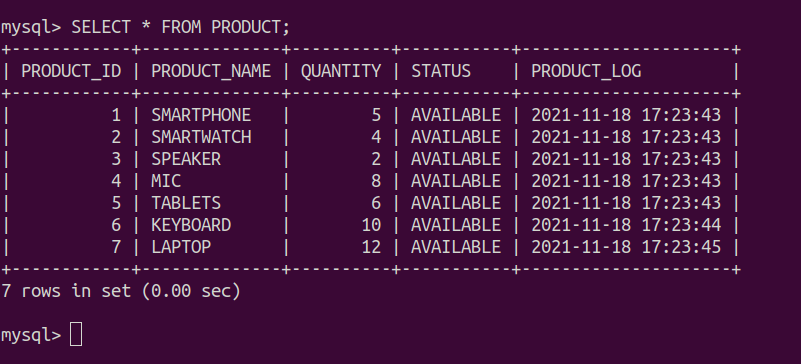
**Updated inventory table**

| **SELECT \* FROM INVENTORY;** |
| --- |

****

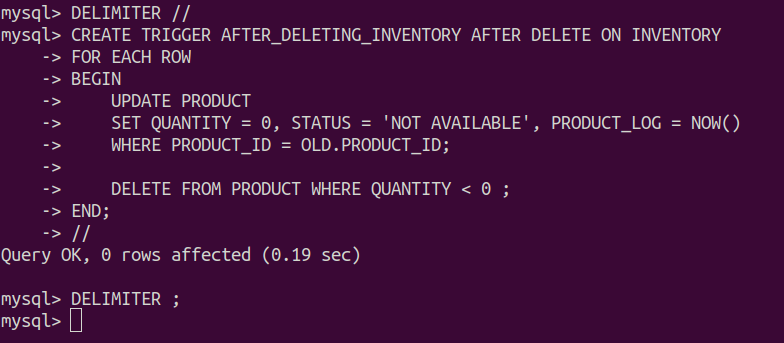
**Update product table**

| **SELECT \* FROM PRODUCT;** |
| --- |

****

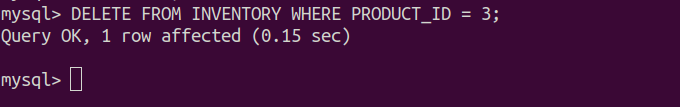
**After delete trigger on inventory**

| **DELIMITER // CREATE TRIGGER AFTER\_DELETING\_INVENTORY AFTER DELETE ON INVENTORY FOR EACH ROW BEGIN  UPDATE PRODUCT  SET QUANTITY = 0, STATUS = 'NOT AVAILABLE', PRODUCT\_LOG = NOW()  WHERE PRODUCT\_ID = OLD.PRODUCT\_ID;   DELETE FROM PRODUCT WHERE QUANTITY < 0 ; END; // DELIMITER ;** |
| --- |

****

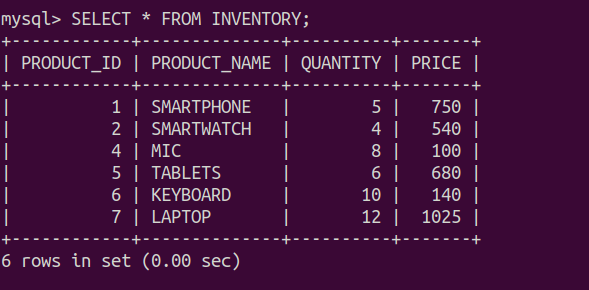
**Delete command on inventory**

| **DELETE FROM INVENTORY WHERE PRODUCT\_ID = 3;** |
| --- |

****

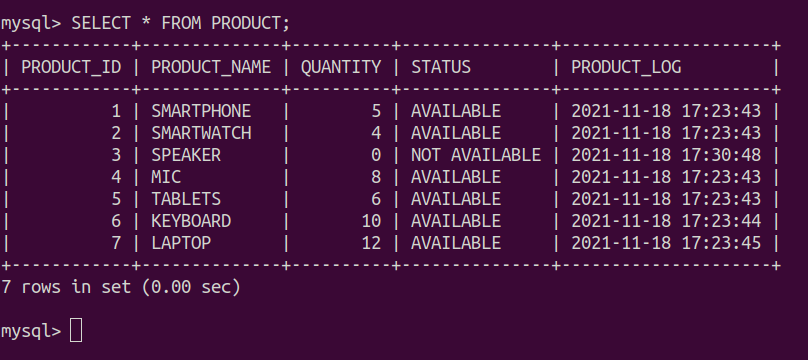
**(updated inventory product\_id=3 gone)**

| **SELECT \* FROM INVENTORY;** |
| --- |

****

**(updated product product\_id=3 updated quantity as 0 and not available)**

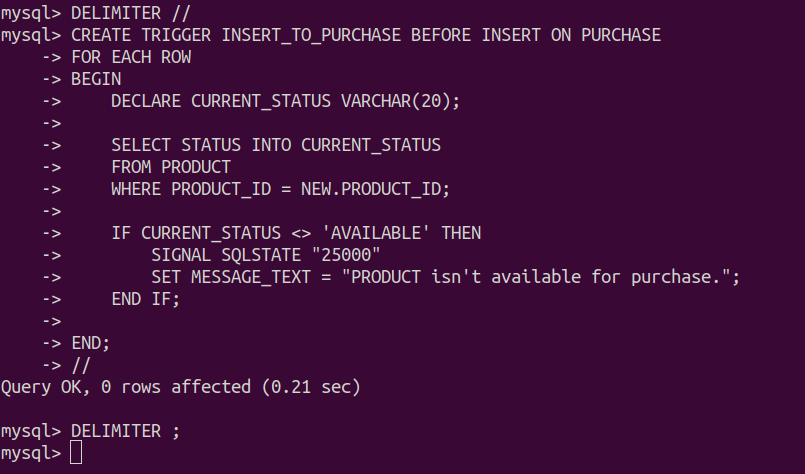
| **SELECT \* FROM PRODUCT;** |
| --- |

****

1. **Write a before insert trigger on Purchase table to check whether the product selected by the user is available or not for purchase. If available,enter an entry else show error.**

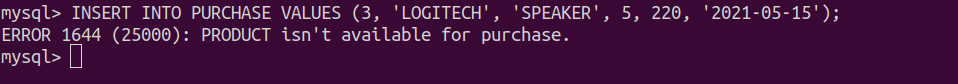
**Before insert trigger on purchase**

| **DELIMITER *//* CREATE TRIGGER INSERT\_TO\_PURCHASE BEFORE INSERT ON PURCHASE FOR EACH ROW BEGIN  DECLARE CURRENT\_STATUS VARCHAR(20);   SELECT STATUS INTO CURRENT\_STATUS  FROM PRODUCT  WHERE PRODUCT\_ID = NEW.PRODUCT\_ID;   IF CURRENT\_STATUS <> 'AVAILABLE' THEN  SIGNAL SQLSTATE "25000"  SET MESSAGE\_TEXT = "PRODUCT isn't available for purchase.";  END IF;  END; *//* DELIMITER ;** |
| --- |

****

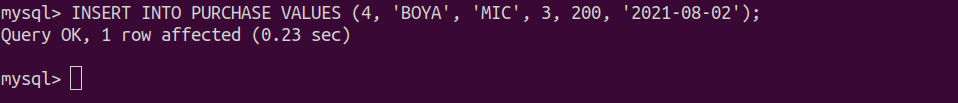
**Insert a new data to purchase(since the product\_id = 3 isn’t ‘available’ so throw the error message)**

| **INSERT INTO PURCHASE VALUES (3, 'LOGITECH', 'SPEAKER', 5, 220, '2021-05-15');** |
| --- |

****

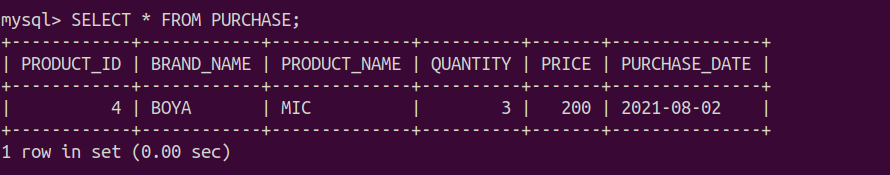
**Insert a new data to purchase(since the product\_id = 4 is ‘available’ so insertion successful)**

| **INSERT INTO PURCHASE VALUES (4, 'BOYA', 'MIC', 3, 200, '2021-08-02');** |
| --- |

****

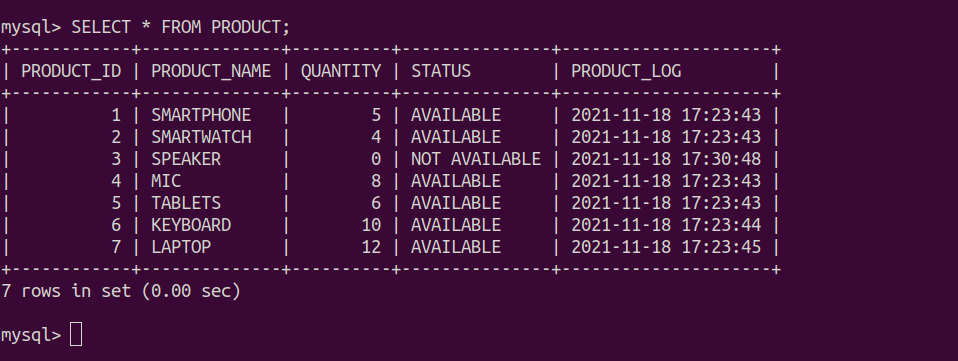
**Update purchase table with product\_id = 4**

| **SELECT \* FROM PURCHASE;** |
| --- |

****

**Quantity is to updated in the upcoming question’s query**

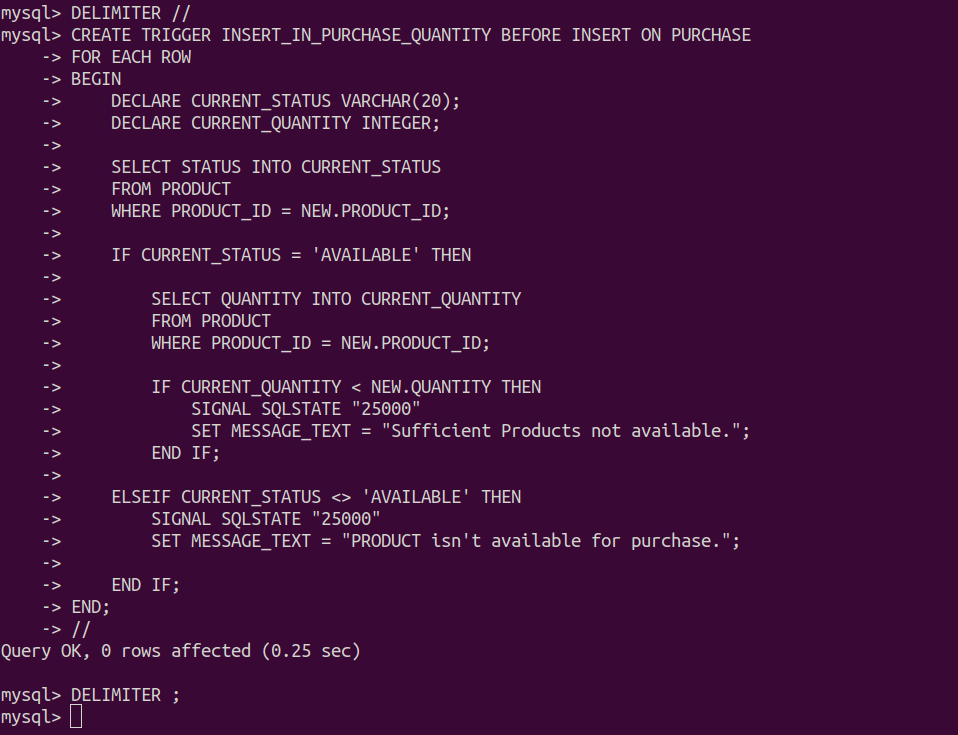
| **SELECT \* FROM PRODUCT;** |
| --- |

****

1. **Extending IV, also check for the quantity of the product. If available, enter an entry else show error.**

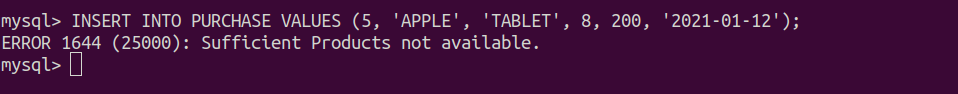
**Before insert trigger on purchase**

| **DELIMITER // CREATE TRIGGER INSERT\_IN\_PURCHASE\_QUANTITY BEFORE INSERT ON PURCHASE FOR EACH ROW BEGIN  DECLARE CURRENT\_STATUS VARCHAR(20);  DECLARE CURRENT\_QUANTITY INTEGER;   SELECT STATUS INTO CURRENT\_STATUS  FROM PRODUCT  WHERE PRODUCT\_ID = NEW.PRODUCT\_ID;   IF CURRENT\_STATUS = 'AVAILABLE' THEN   SELECT QUANTITY INTO CURRENT\_QUANTITY  FROM PRODUCT  WHERE PRODUCT\_ID = NEW.PRODUCT\_ID;   IF CURRENT\_QUANTITY < NEW.QUANTITY THEN  SIGNAL SQLSTATE "25000"  SET MESSAGE\_TEXT = "Sufficient Products not available.";  END IF;   ELSEIF CURRENT\_STATUS <> 'AVAILABLE' THEN  SIGNAL SQLSTATE "25000"  SET MESSAGE\_TEXT = "PRODUCT isn't available for purchase.";   END IF; END; // DELIMITER ;** |
| --- |

****

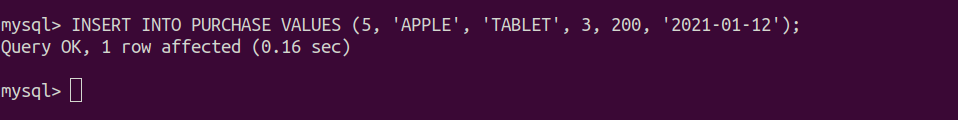
**Insert a new data to purchase(since the product\_id = 5 is available but isn’t having the required quantity so throw the error message)**

| **INSERT INTO PURCHASE VALUES (5, 'APPLE', 'TABLET', 8, 200, '2021-01-12');** |
| --- |

****

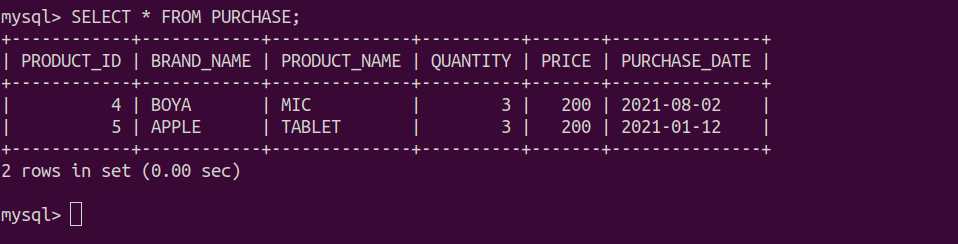
**Insert a new data to purchase(since the product\_id = 5 is ‘available’ and purchasing quantity is less than available so successfully inserted)**

| **INSERT INTO PURCHASE VALUES (5, 'APPLE', 'TABLET', 3, 200, '2021-01-12');** |
| --- |

****

**Updated data on purchase table**

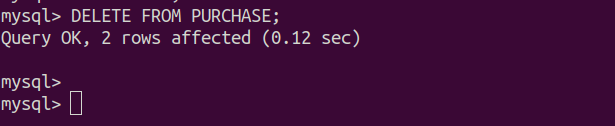
| **SELECT \* FROM PURCHASE;** |
| --- |

****

1. **Write a after insert trigger on Purchase table to update the values at Inventory table.**

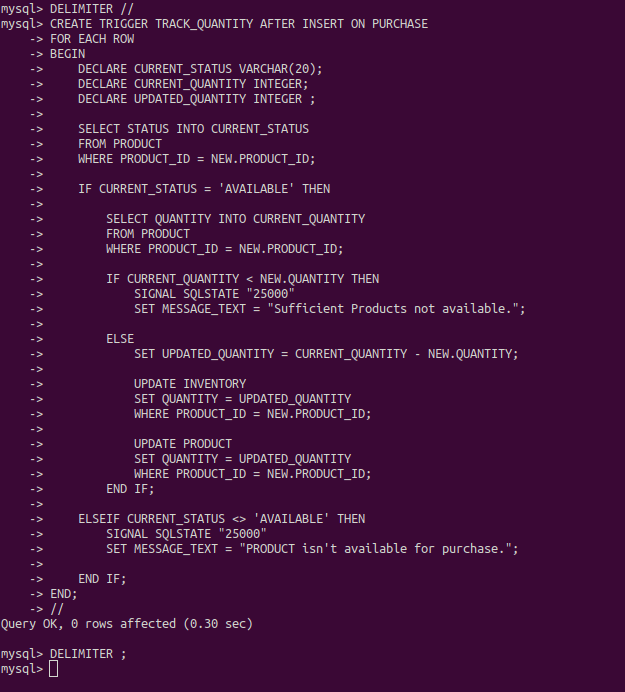
**Clean the entire purchase table since we have to update the quantities as well**

| **DELETE FROM PURCHASE;** |
| --- |

****

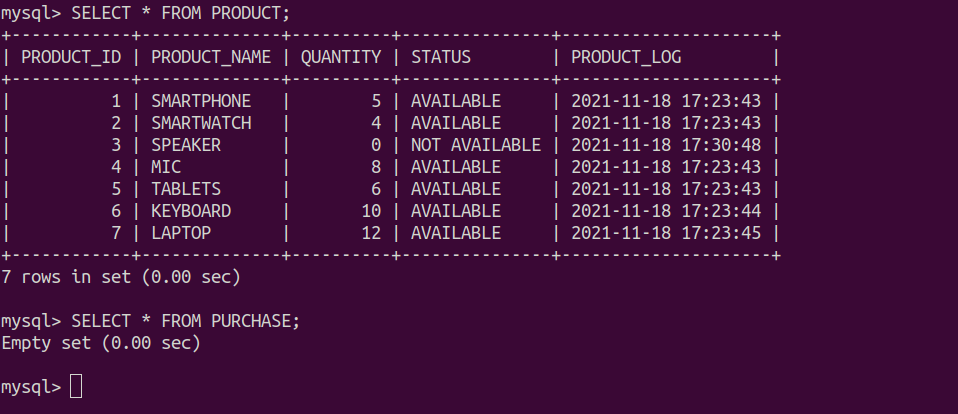
**After insert trigger on purchase table**

| **DELIMITER *//* CREATE TRIGGER TRACK\_QUANTITY AFTER INSERT ON PURCHASE FOR EACH ROW BEGIN  DECLARE CURRENT\_STATUS VARCHAR(20);  DECLARE CURRENT\_QUANTITY INTEGER;  DECLARE UPDATED\_QUANTITY INTEGER ;   SELECT STATUS INTO CURRENT\_STATUS  FROM PRODUCT  WHERE PRODUCT\_ID = NEW.PRODUCT\_ID;   IF CURRENT\_STATUS = 'AVAILABLE' THEN   SELECT QUANTITY INTO CURRENT\_QUANTITY  FROM PRODUCT  WHERE PRODUCT\_ID = NEW.PRODUCT\_ID;   IF CURRENT\_QUANTITY < NEW.QUANTITY THEN  SIGNAL SQLSTATE "25000"  SET MESSAGE\_TEXT *=* *"Sufficient Products not available."*;    ELSE  SET UPDATED\_QUANTITY *= CURRENT\_QUANTITY - NEW.QUANTITY*;    UPDATE INVENTORY  SET QUANTITY *= UPDATED\_QUANTITY*  WHERE *PRODUCT\_ID = NEW.PRODUCT\_ID*;   UPDATE PRODUCT  SET QUANTITY *= UPDATED\_QUANTITY*  WHERE *PRODUCT\_ID = NEW.PRODUCT\_ID*;  END IF;   ELSEIF CURRENT\_STATUS <> 'AVAILABLE' THEN  SIGNAL SQLSTATE "25000"  SET MESSAGE\_TEXT *=* *"PRODUCT isn't available for purchase."*;   END IF; END; *//* DELIMITER ;** |
| --- |

****

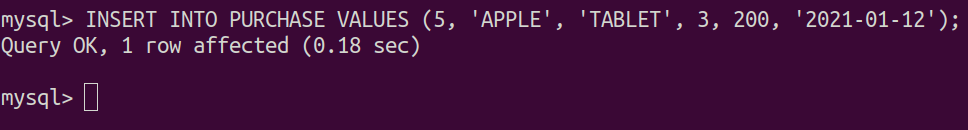
**Before requesting for any purchase the current data at product and purchase table**

| **SELECT \* FROM PRODUCT; SELECT \* FROM PURCHASE;** |
| --- |

****

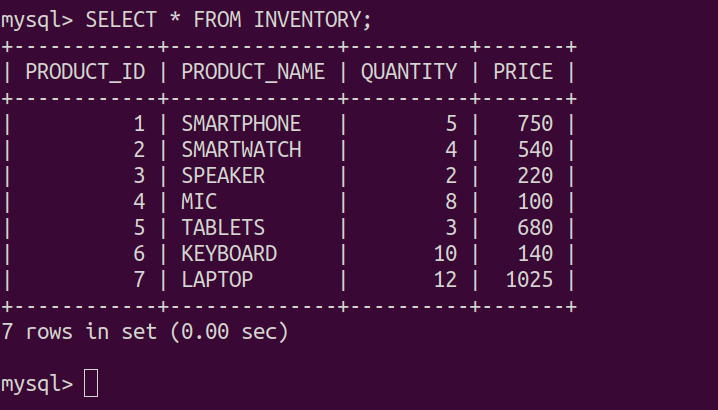
**Since the product\_id = 5 is available and required quantity is less than available so query successful**

| **INSERT INTO PURCHASE VALUES (5, 'APPLE', 'TABLET', 3, 200, '2021-01-12');** |
| --- |

****

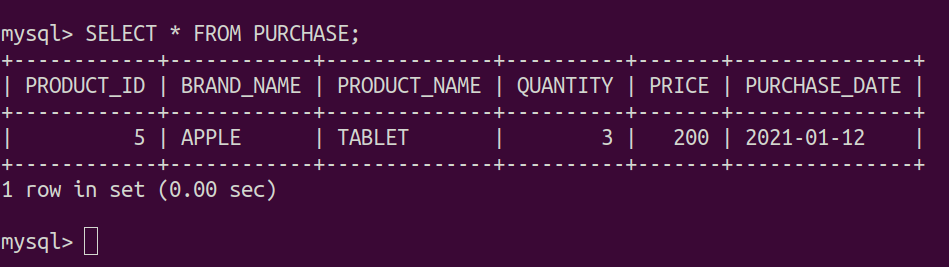
**The updated quantity for prduct\_id = 5 is reduced by 3 from 6, since 3 units are purchased**

| **SELECT \* FROM INVENTORY;** |
| --- |

****

**Updated purchase table**

| **SELECT \* FROM PURCHASE;** |
| --- |

****

Q2. Construct a table with the following details given below:

Employee (EID, DeptID, ENAME, PNUMBER, SALARY, DESIGNATION)

Department (DID, DeptNAME, NUM\_OF\_EMPLOYEES, HEAD\_NAME,

MANAGERS, WORKERS)

Post (DESIGNATION, NUM\_OF\_EMPLOYEES, TOTAL\_AMOUNT)

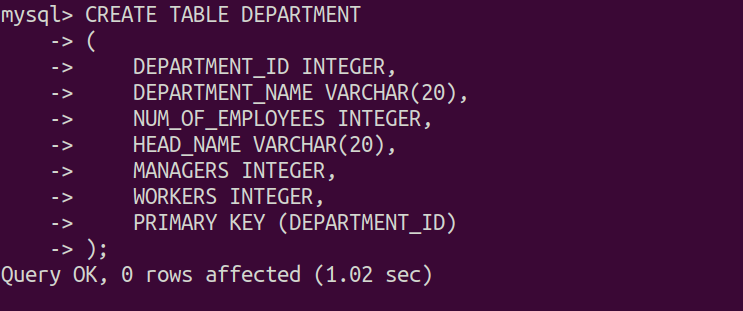
EID, DeptID: Primary keys for the table Employee.

DID: Primary key for the table Department.

DESIGNATION: Primary keys for the table Post.

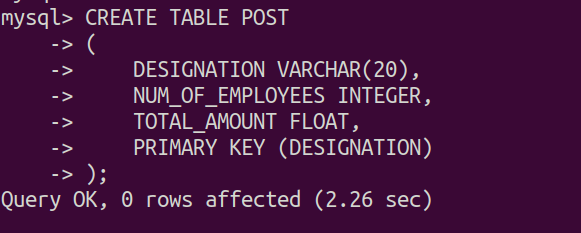
**Department Table Creation**

| **CREATE** **TABLE** DEPARTMENT (  DEPARTMENT\_ID INTEGER,  DEPARTMENT\_NAME VARCHAR(20),  NUM\_OF\_EMPLOYEES INTEGER,  HEAD\_NAME VARCHAR(20),  MANAGERS INTEGER,  WORKERS INTEGER,  PRIMARY **KEY** (DEPARTMENT\_ID) ); |
| --- |



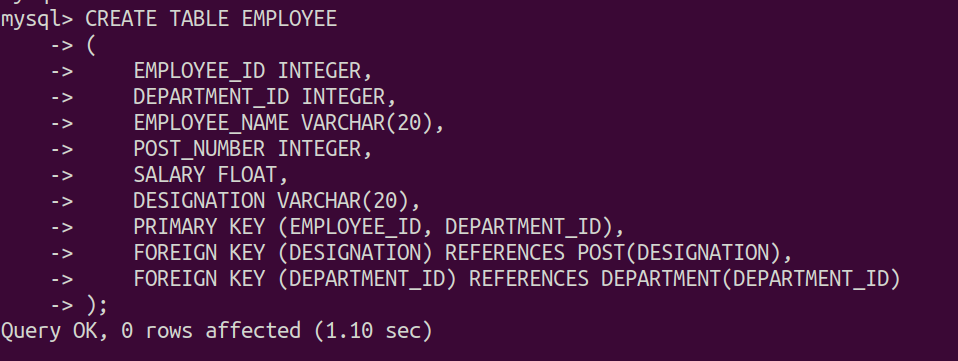
**POST Table Creation**

| **CREATE** **TABLE** POST (  DESIGNATION VARCHAR(20),  NUM\_OF\_EMPLOYEES INTEGER,  TOTAL\_AMOUNT FLOAT,  PRIMARY **KEY** (DESIGNATION) ); |
| --- |



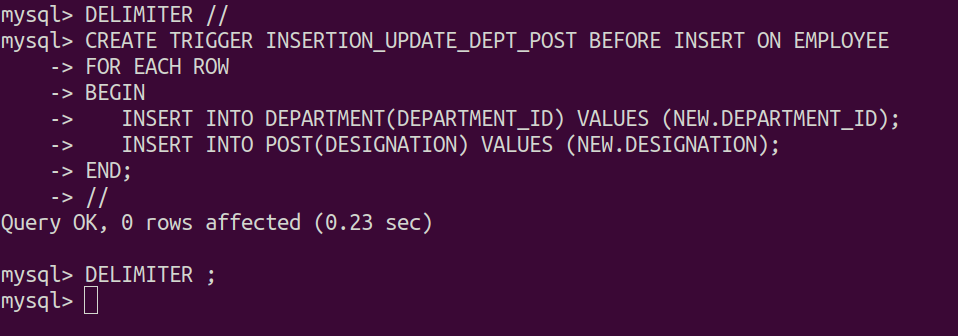
**EMPLOYEE Table Creation**

| CREATE TABLE EMPLOYEE (  EMPLOYEE\_ID INTEGER,  DEPARTMENT\_ID INTEGER,  EMPLOYEE\_NAME VARCHAR(20),  POST\_NUMBER INTEGER,  SALARY FLOAT,  DESIGNATION VARCHAR(20),  PRIMARY KEY (EMPLOYEE\_ID, DEPARTMENT\_ID),  FOREIGN KEY (DESIGNATION) REFERENCES POST(DESIGNATION),  FOREIGN KEY (DEPARTMENT\_ID) REFERENCES DEPARTMENT(DEPARTMENT\_ID) )*;* |
| --- |



1. Write a before insert and before delete trigger on the Employee table to insert/update records in Department and Post table.

| DELIMITER // **CREATE** **TRIGGER** INSERTION\_UPDATE\_DEPT\_POST **BEFORE** **INSERT** **ON** EMPLOYEE **FOR** **EACH** **ROW** **BEGIN**  **INSERT** **INTO** DEPARTMENT(DEPARTMENT\_ID) **VALUES** (NEW.DEPARTMENT\_ID);  **INSERT** **INTO** POST(DESIGNATION) **VALUES** (NEW.DESIGNATION); **END**; // DELIMITER ; |
| --- |



1. Write a before insert trigger on the Employee table to input custom value for column having NULL value.
2. Extending II, also write to allow DeptID to be an existing one.
3. Write a before delete trigger on the Employee table to update values in the Department and Post table.
4. Write a before update trigger on the Department table to restrict changes on the NUM\_OF\_EMPLOYEES column.
5. Write a before update trigger on the Department table to change value on NUM\_OF\_EMPLOYEES column only when there is a change in MANAGERS or WORKERS columns and update on Post table.
6. Write a before insert and before update trigger on the Department table to use HEAD\_NAME with the Employee having Designation as HEAD.
7. Write a before update trigger on Post table to restrict changes onNUM\_OF\_EMPLOYEES column