Database Management System (UE20CS301)
DBMS - Mini Project

FARM MANAGEMENT SYSTEM

Submitted By:
Name: SYED AZFAR RAYAN
SRN: PES1UG20CS453
Vth Semester
Section: “H”

PAGE OF CONTENTS

Description of the Project

This project is an Agriculture Farm Management System. It helps the producers to buy fresh produce from the farmers. This helps to reduce the commercial gap between the consumer & producer. It has provision to make an account, both for farmer as well as the consumer. The farmer can upload his produce onto the farm catalogue. The user can browse through the catalogue and add produce to the cart.

The aim is to bring fresh produce to the customer directly from the farmers, & for the farmers to eliminate middlemen. The user interface has been made simple to use, seamless & has required functionalities. . This Farm Management System will reduce the burden on farmers and will make the system efficient by providing the more accurate details about the customer orders & demand. The frontend is capable of performing many operations & is easy to use.

Scope of the Project:

The database used is MySQL. The frontend is primarily implemented in PHP. It uses a XAMPP server for it to be locally hosted on the local network. This provides for a good user experience for the farmer & consumer.

The project consists of four tables. The farmer table contains all details of the farmer, the buyer table contains all details of the buyer. The fproduct tables hold the inventory of the products, & the mycart table hold the details of the transactions between buyer & the farmer.

ER Diagram

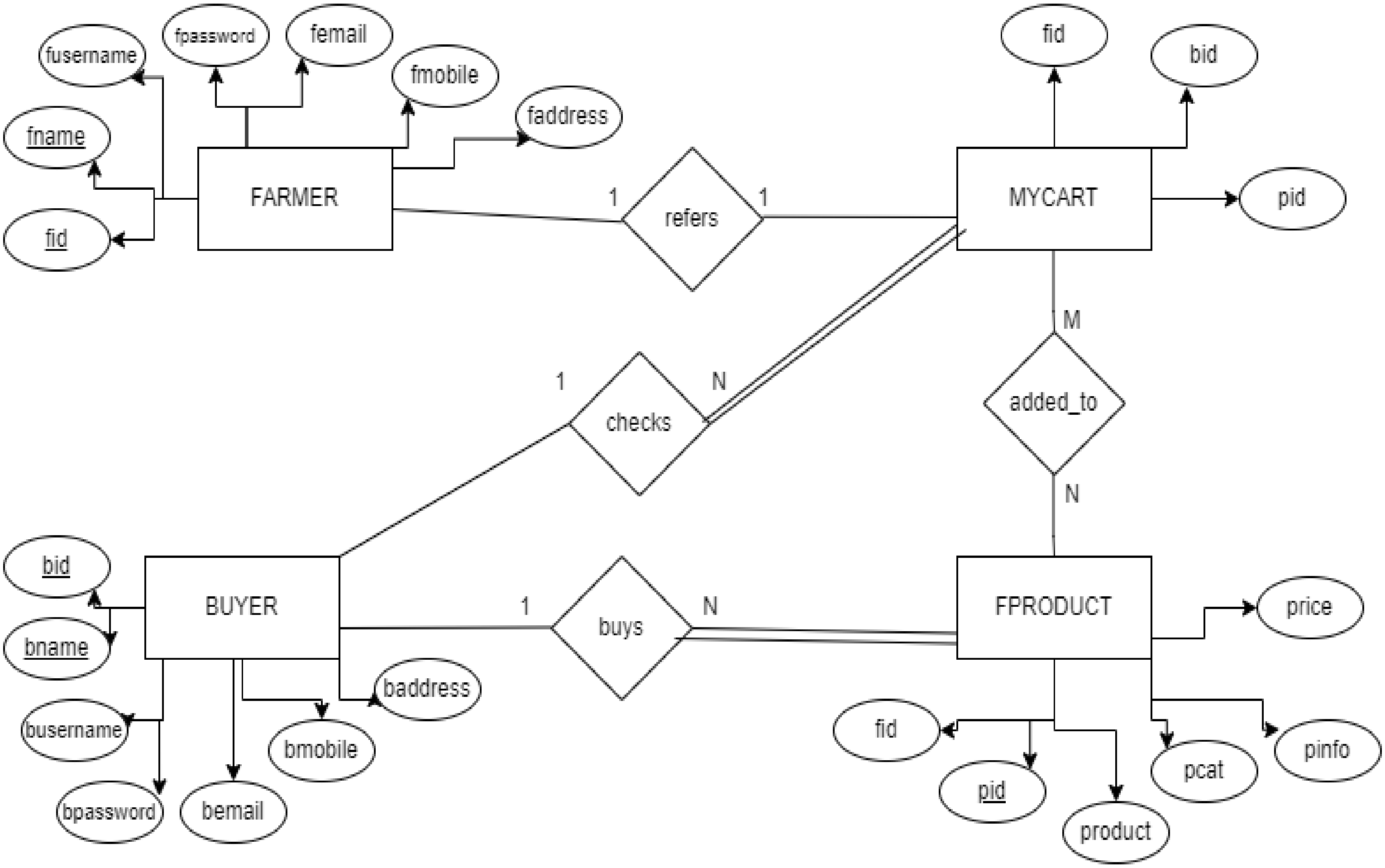


Fig. 1: ER Diagram

Relational Schema

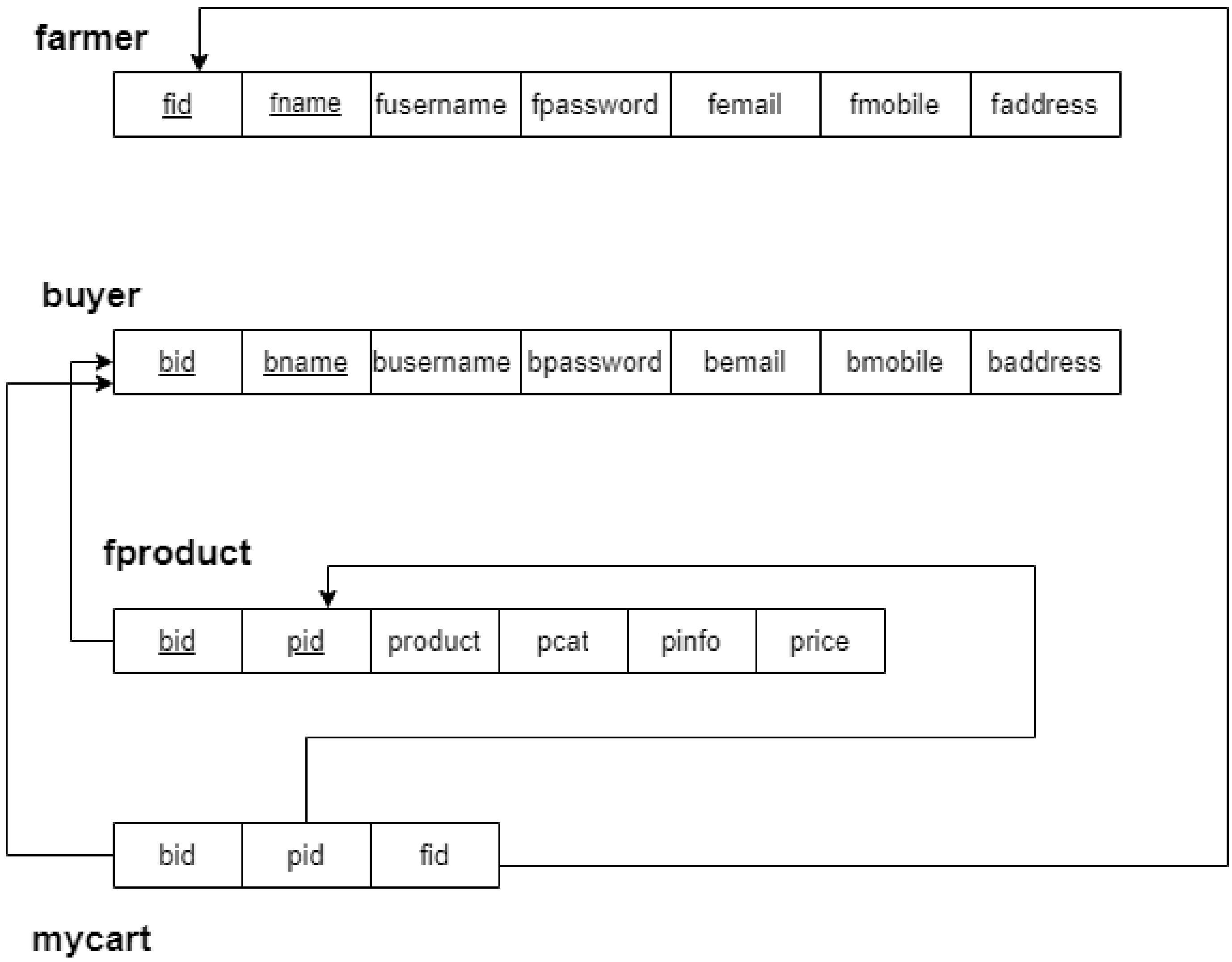
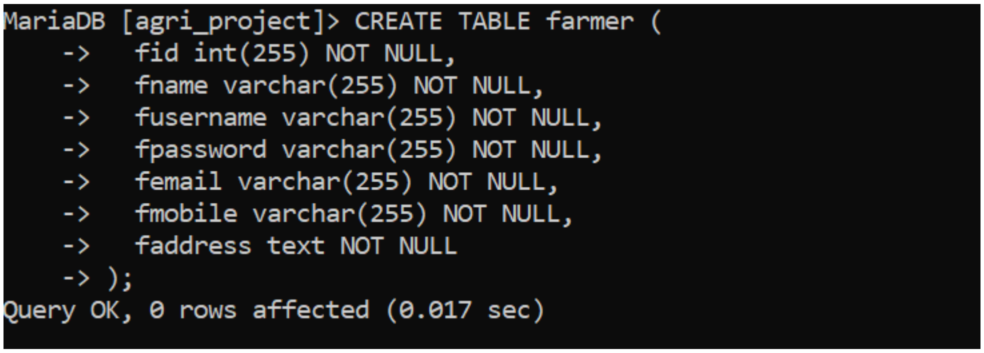


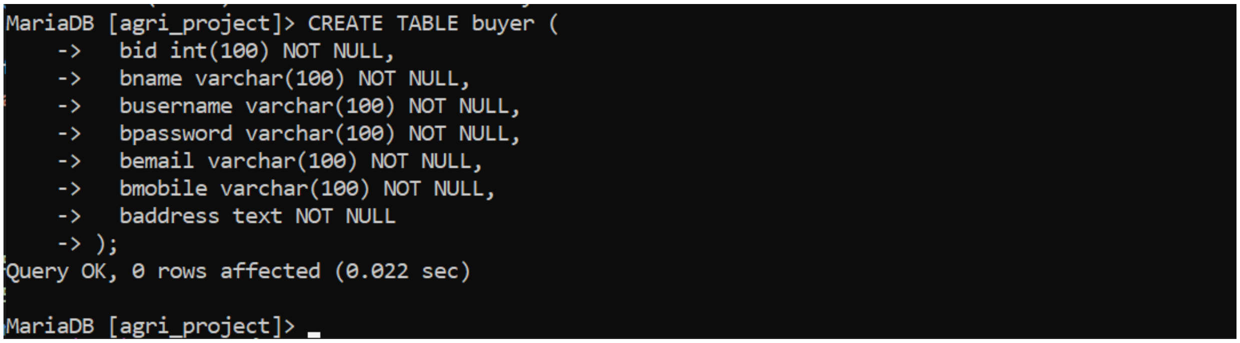
Fig. 2: Relational Schema

DDL statements - Building the database



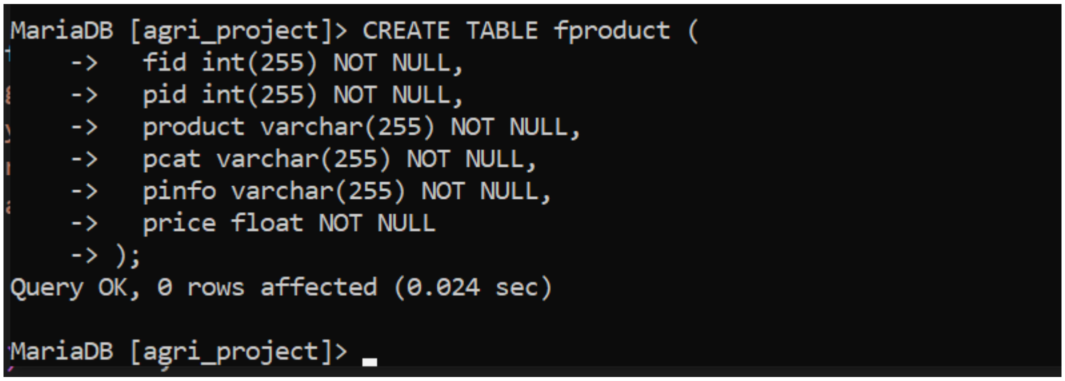
```
MariaDB [agri_project]> CREATE TABLE farmer (
    ->   fid int(255) NOT NULL,
    ->   fname varchar(255) NOT NULL,
    ->   fusername varchar(255) NOT NULL,
    ->   fpassword varchar(255) NOT NULL,
    ->   femail varchar(255) NOT NULL,
    ->   fmobile varchar(255) NOT NULL,
    ->   faddress text NOT NULL
    -> );
Query OK, 0 rows affected (0.017 sec)
```

Fig. 3: Creating Farmer Table



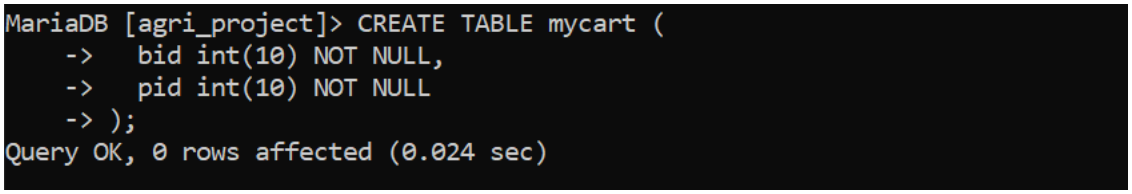
```
MariaDB [agri_project]> CREATE TABLE buyer (
    ->   bid int(100) NOT NULL,
    ->   bname varchar(100) NOT NULL,
    ->   busername varchar(100) NOT NULL,
    ->   bpassword varchar(100) NOT NULL,
    ->   bemail varchar(100) NOT NULL,
    ->   bmobile varchar(100) NOT NULL,
    ->   baddress text NOT NULL
    -> );
Query OK, 0 rows affected (0.022 sec)

MariaDB [agri_project]> _
```

Fig. 4: Creating Buyer table



```
MariaDB [agri_project]> CREATE TABLE fproduct (
    ->   fid int(255) NOT NULL,
    ->   pid int(255) NOT NULL,
    ->   product varchar(255) NOT NULL,
    ->   pcat varchar(255) NOT NULL,
    ->   pinfo varchar(255) NOT NULL,
    ->   price float NOT NULL
    -> );
Query OK, 0 rows affected (0.024 sec)

MariaDB [agri_project]> _
```

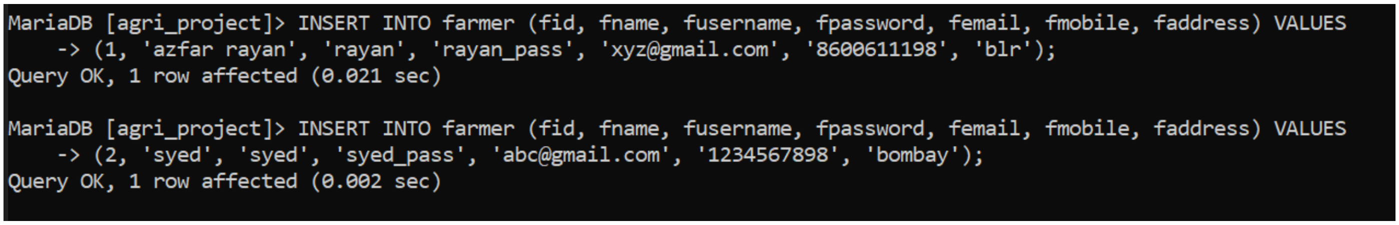
Fig. 5: Creating fproduct table



```
MariaDB [agri_project]> CREATE TABLE mycart (
    ->   bid int(10) NOT NULL,
    ->   pid int(10) NOT NULL
    -> );
Query OK, 0 rows affected (0.024 sec)
```

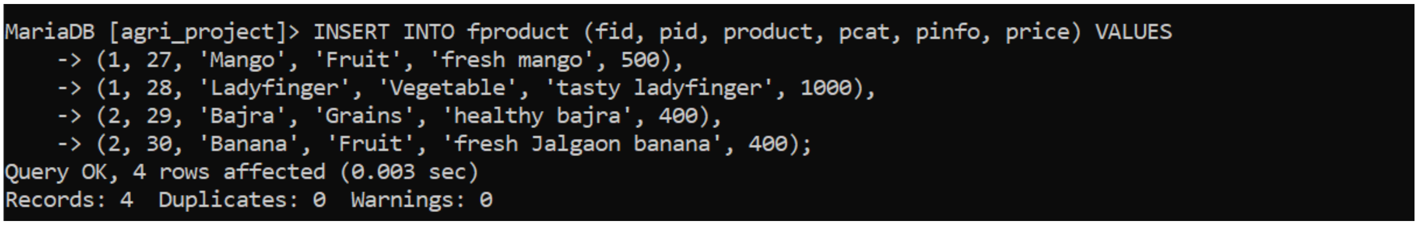
Fig. 6: Creating mycart table

Populating the Database



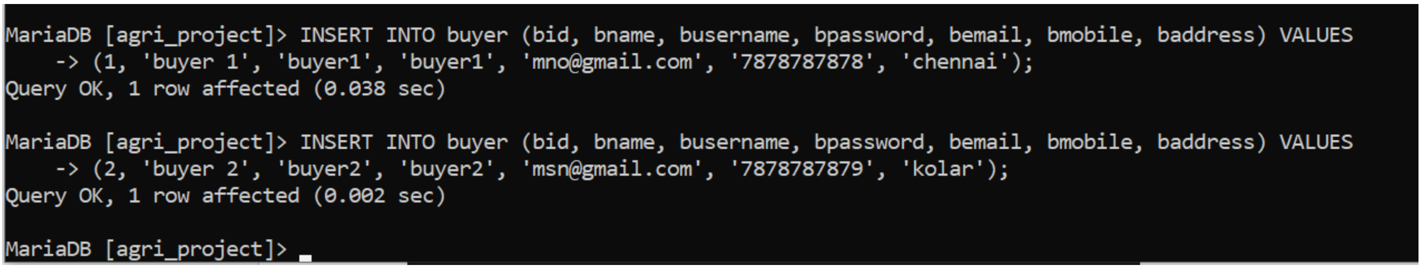
```
MariaDB [agri_project]> INSERT INTO farmer (fid, fname, fusername, fpassword, femail, fmobile, faddress) VALUES
    -> (1, 'azfar rayan', 'rayan', 'rayan_pass', 'xyz@gmail.com', '8600611198', 'blr');
Query OK, 1 row affected (0.021 sec)

MariaDB [agri_project]> INSERT INTO farmer (fid, fname, fusername, fpassword, femail, fmobile, faddress) VALUES
    -> (2, 'syed', 'syed', 'syed_pass', 'abc@gmail.com', '1234567898', 'bombay');
Query OK, 1 row affected (0.002 sec)
```

Fig. 7: Adding Farmer Details



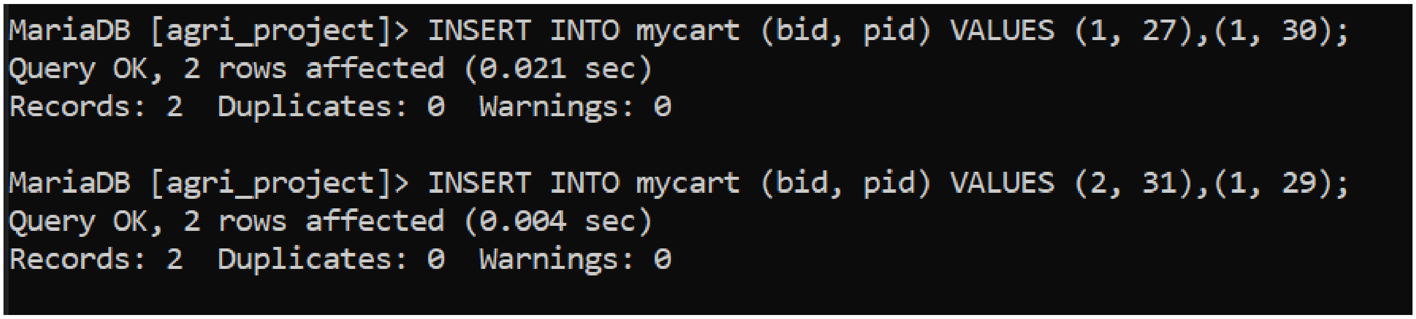
```
MariaDB [agri_project]> INSERT INTO fproduct (fid, pid, product, pcat, pinfo, price) VALUES
    -> (1, 27, 'Mango', 'Fruit', 'fresh mango', 500),
    -> (1, 28, 'Ladyfinger', 'Vegetable', 'tasty ladyfinger', 1000),
    -> (2, 29, 'Bajra', 'Grains', 'healthy bajra', 400),
    -> (2, 30, 'Banana', 'Fruit', 'fresh Jalgaon banana', 400);
Query OK, 4 rows affected (0.003 sec)
Records: 4  Duplicates: 0  Warnings: 0
```

Fig. 8: Adding product Details



```
MariaDB [agri_project]> INSERT INTO buyer (bid, bname, busername, bpassword, bemail, bmobile, baddress) VALUES
    -> (1, 'buyer 1', 'buyer1', 'buyer1', 'mno@gmail.com', '7878787878', 'chennai');
Query OK, 1 row affected (0.038 sec)

MariaDB [agri_project]> INSERT INTO buyer (bid, bname, busername, bpassword, bemail, bmobile, baddress) VALUES
    -> (2, 'buyer 2', 'buyer2', 'buyer2', 'msn@gmail.com', '7878787879', 'kolar');
Query OK, 1 row affected (0.002 sec)

MariaDB [agri_project]> _
```

Fig. 9: Adding Buyer Details



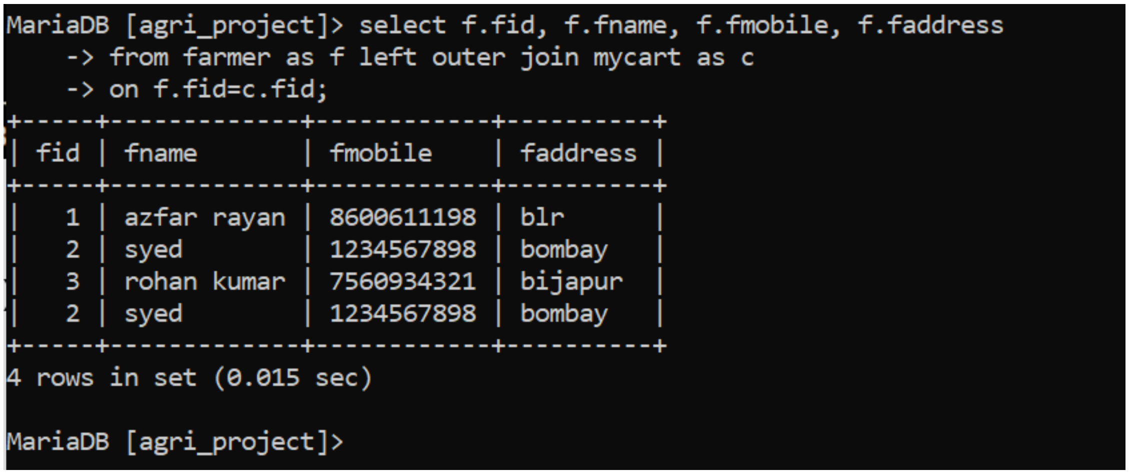
```
MariaDB [agri_project]> INSERT INTO mycart (bid, pid) VALUES (1, 27),(1, 30);
Query OK, 2 rows affected (0.021 sec)
Records: 2  Duplicates: 0  Warnings: 0

MariaDB [agri_project]> INSERT INTO mycart (bid, pid) VALUES (2, 31),(1, 29);
Query OK, 2 rows affected (0.004 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

Fig. 10: Adding cart Details

Join Queries

1) Displaying the farmer details by the product in the cart of all buyers.

select f.fid, f.fname, f.fmobile, f.faddress
-> from farmer as f left outer join mycart as c
-> on f.fid=c.fid;

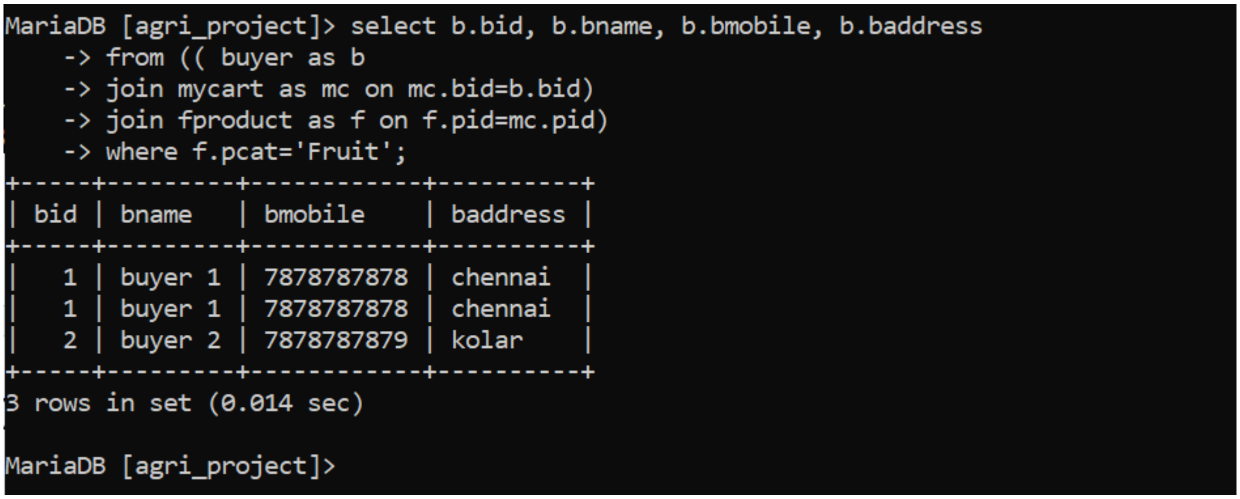


```
MariaDB [agri_project]> select f.fid, f.fname, f.fmobile, f.faddress
    -> from farmer as f left outer join mycart as c
    -> on f.fid=c.fid;
+-----+-------------+------------+----------+
| fid | fname       | fmobile    | faddress |
+-----+-------------+------------+----------+
|   1 | azfar rayan | 8600611198 | blr      |
|   2 | syed        | 1234567898 | bombay   |
|   3 | rohan kumar | 7560934321 | bijapur  |
|   2 | syed        | 1234567898 | bombay   |
+-----+-------------+------------+----------+
4 rows in set (0.015 sec)

MariaDB [agri_project]>
```

Fig. 11: Left outer join on farmer & mycart

2) Display buyer details who bought Fruits from the farmers.

select b.bid, b.bname, b.bmobile, b.baddress
-> from ((buyer as b
-> join mycart as mc on mc.bid=b.bid)
-> join fproduct as f on f.pid=mc.pid)
-> where f.pcat='Fruit';

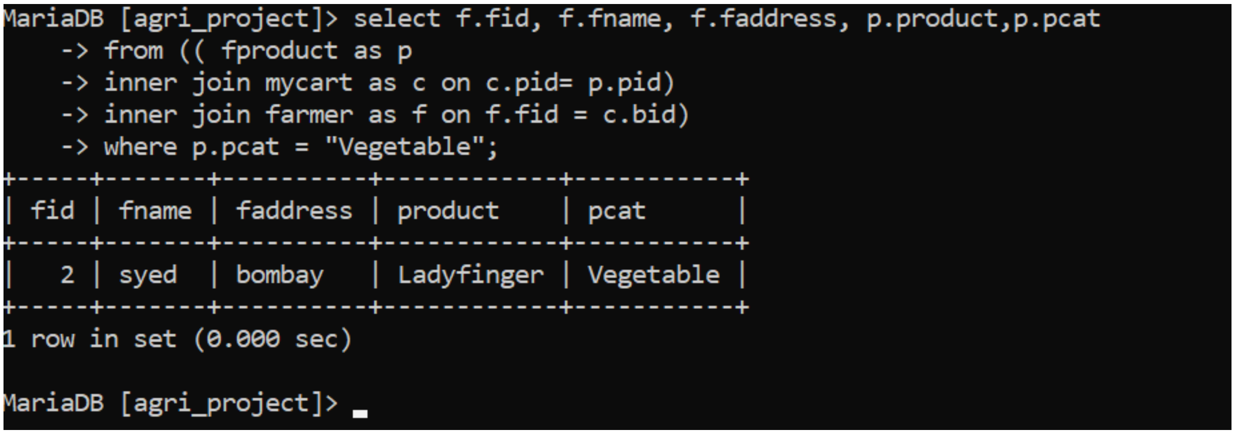


```
MariaDB [agri_project]> select b.bid, b.bname, b.bmobile, b.baddress
    -> from (( buyer as b
    -> join mycart as mc on mc.bid=b.bid)
    -> join fproduct as f on f.pid=mc.pid)
    -> where f.pcat='Fruit';
+-----+---------+------------+----------+
| bid | bname   | bmobile    | baddress |
+-----+---------+------------+----------+
|   1 | buyer 1 | 7878787878 | chennai  |
|   1 | buyer 1 | 7878787878 | chennai  |
|   2 | buyer 2 | 7878787879 | kolar    |
+-----+---------+------------+----------+
3 rows in set (0.014 sec)

MariaDB [agri_project]>
```

Fig. 12: Full join on buyer, mycart & fproduct

3) Displaying the farmer id, name & address with product name & category for the buyers who bought Vegetables.

select f.fid, f.fname, f.faddress, p.product,p.pcat
-> from ((fproduct as p
-> inner join mycart as c on c.pid= p.pid)
-> inner join farmer as f on f.fid = c.bid)
-> where p.pcat = "Vegetable";

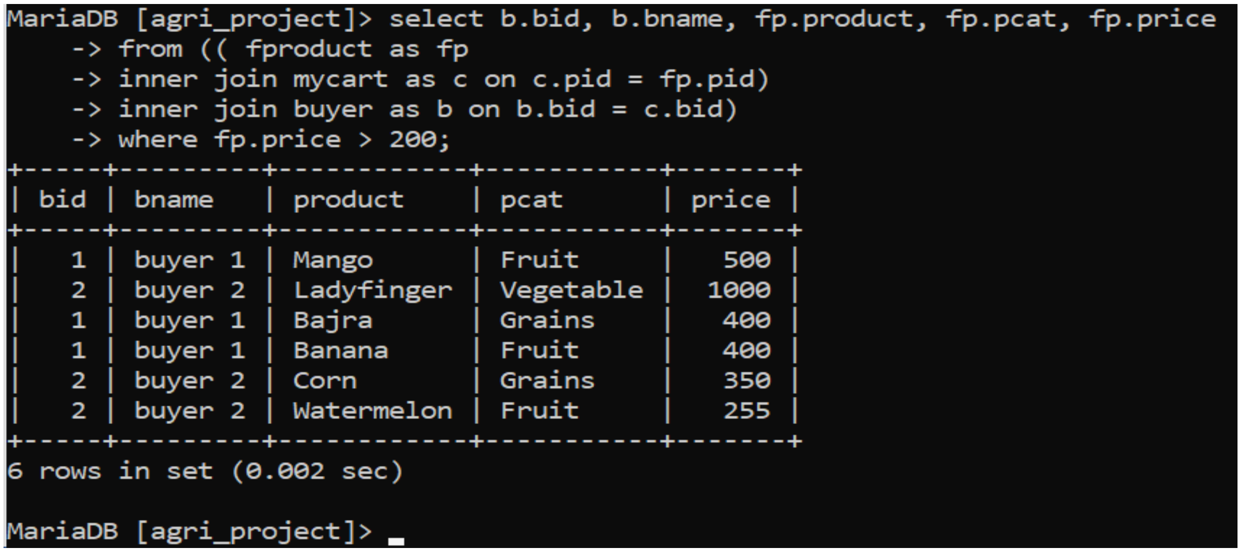


```
MariaDB [agri_project]> select f.fid, f.fname, f.faddress, p.product,p.pcat
    -> from (( fproduct as p
    -> inner join mycart as c on c.pid= p.pid)
    -> inner join farmer as f on f.fid = c.bid)
    -> where p.pcat = "Vegetable";
+-----+-------+----------+------------+-----------+
| fid | fname | faddress | product    | pcat      |
+-----+-------+----------+------------+-----------+
|   2 | syed  | bombay   | Ladyfinger | Vegetable |
+-----+-------+----------+------------+-----------+
1 row in set (0.000 sec)

MariaDB [agri_project]> _
```

Fig. 13: Inner join on fproduct, mycart, farmer

4) Display Product and Buyer details for those purchases where price of the farm produce was **greater than Rs.200**.

select b.bid, b.bname, fp.product, fp.pcat, fp.price
-> from ((fproduct as fp
-> inner join mycart as c on c.pid = fp.pid)
-> inner join buyer as b on b.bid = c.bid)
-> where fp.price > 200;



```
MariaDB [agri_project]> select b.bid, b.bname, fp.product, fp.pcat, fp.price
    -> from (( fproduct as fp
    -> inner join mycart as c on c.pid = fp.pid)
    -> inner join buyer as b on b.bid = c.bid)
    -> where fp.price > 200;
+-----+---------+------------+-----------+-------+
| bid | bname   | product    | pcat      | price |
+-----+---------+------------+-----------+-------+
|   1 | buyer 1 | Mango      | Fruit     |   500 |
|   2 | buyer 2 | Ladyfinger | Vegetable |  1000 |
|   1 | buyer 1 | Bajra      | Grains    |   400 |
|   1 | buyer 1 | Banana     | Fruit     |   400 |
|   2 | buyer 2 | Corn       | Grains    |   350 |
|   2 | buyer 2 | Watermelon | Fruit     |   255 |
+-----+---------+------------+-----------+-------+
6 rows in set (0.002 sec)

MariaDB [agri_project]> _
```

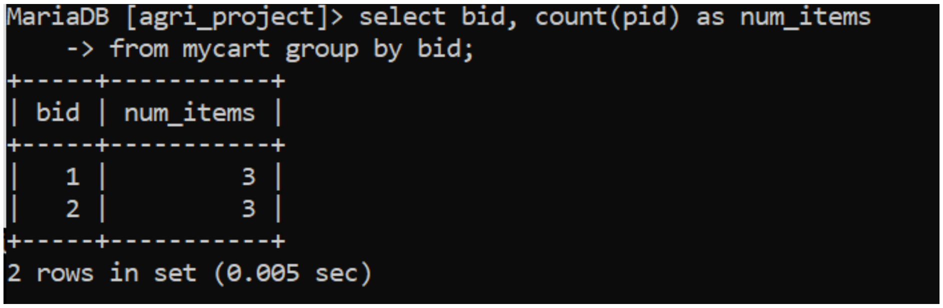
Fig. 14: Inner join on buyer, mycart & fproduct

Aggregate Functions

1) Count the number of items bought by the buyer.

select bid, count(pid) as num_items

-> from mycart group by bid;



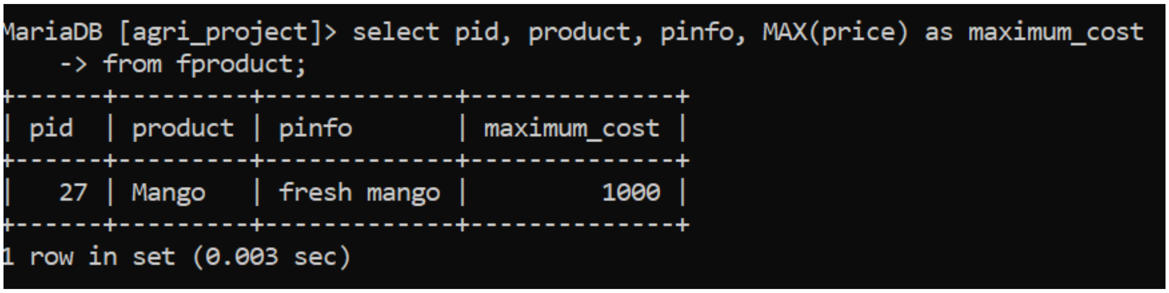
```
MariaDB [agri_project]> select bid, count(pid) as num_items
    -> from mycart group by bid;
+-----+-----------+
| bid | num_items |
+-----+-----------+
|   1 |         3 |
|   2 |         3 |
+-----+-----------+
2 rows in set (0.005 sec)
```

Fig. 15: Using count function

2) Finding the most costly farm produce in the dataset.

select pid, product, pinfo, MAX(price) as maximum_cost

-> from fproduct;

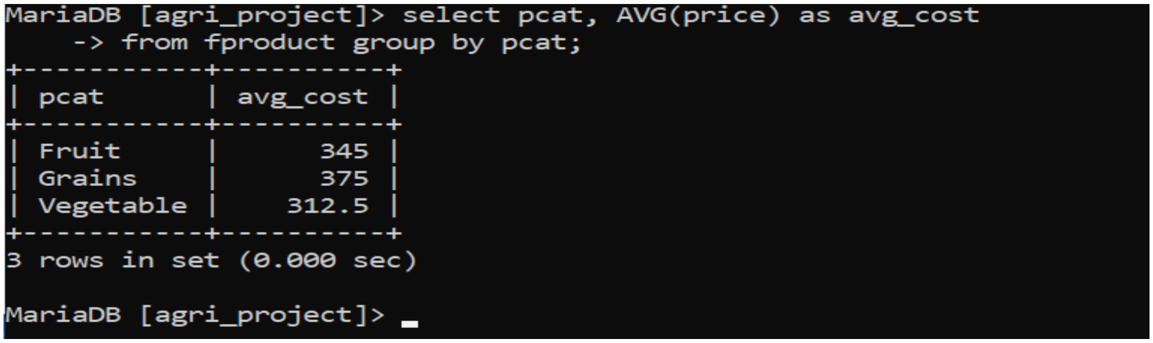


```
MariaDB [agri_project]> select pid, product, pinfo, MAX(price) as maximum_cost
    -> from fproduct;
+------+---------+-------------+--------------+
| pid  | product | pinfo       | maximum_cost |
+------+---------+-------------+--------------+
|   27 | Mango   | fresh mango |         1000 |
+------+---------+-------------+--------------+
1 row in set (0.003 sec)
```

Fig. 16: Using MAX function

3) Finding the average cost of the different category of produce.

select pcat, AVG(price) as avg_cost
   -> from fproduct group by pcat;



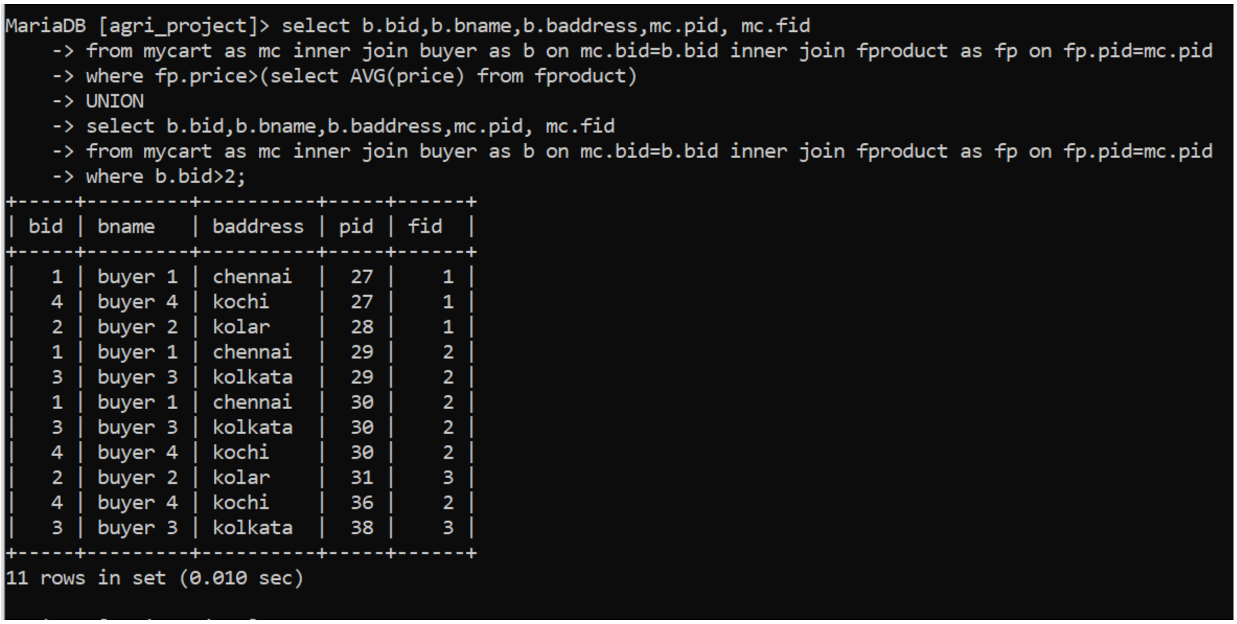
```
MariaDB [agri_project]> select pcat, AVG(price) as avg_cost
    -> from fproduct group by pcat;
+-----------+----------+
| pcat      | avg_cost |
+-----------+----------+
| Fruit     |      345 |
| Grains    |      375 |
| Vegetable |    312.5 |
+-----------+----------+
3 rows in set (0.000 sec)

MariaDB [agri_project]> _
```

Fig. 17: Using AVG function

Set Operations

1) Display Buyer ID, Name, Address & Product ID when price of product is greater than avg price for all produce or when buyer ID is greater than 2.

select b.bid,b.bname,b.baddress,mc.pid, mc.fid
-> from mycart as mc inner join buyer as b on mc.bid=b.bid inner join fproduct as fp on fp.pid=mc.pid
-> where fp.price>(select AVG(price) from fproduct)
-> UNION
-> select b.bid,b.bname,b.baddress,mc.pid, mc.fid
-> from mycart as mc inner join buyer as b on mc.bid=b.bid inner join fproduct as fp on fp.pid=mc.pid
-> where b.bid>2;

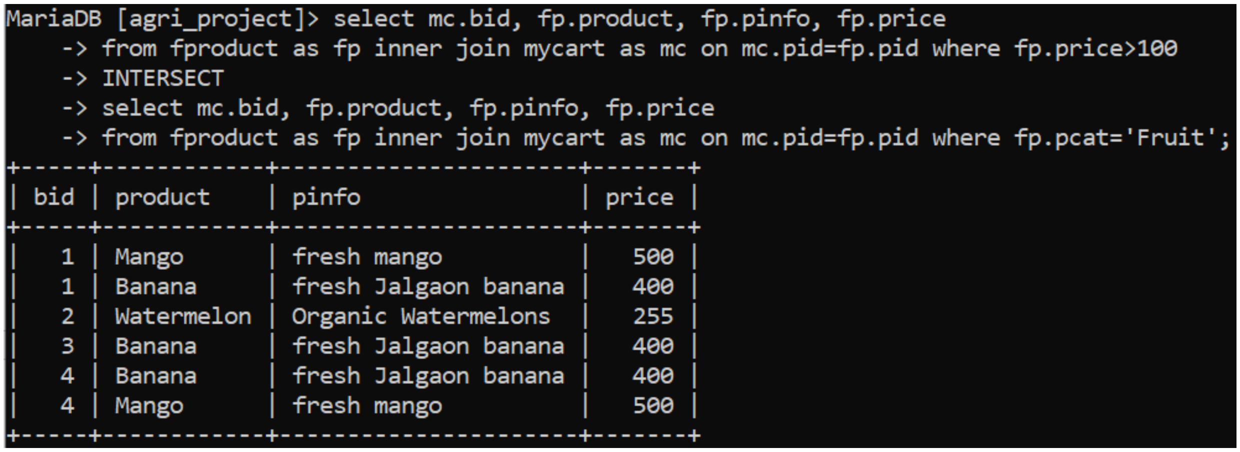


```
MariaDB [agri_project]> select b.bid,b.bname,b.baddress,mc.pid, mc.fid
    -> from mycart as mc inner join buyer as b on mc.bid=b.bid inner join fproduct as fp on fp.pid=mc.pid
    -> where fp.price>(select AVG(price) from fproduct)
    -> UNION
    -> select b.bid,b.bname,b.baddress,mc.pid, mc.fid
    -> from mycart as mc inner join buyer as b on mc.bid=b.bid inner join fproduct as fp on fp.pid=mc.pid
    -> where b.bid>2;
+-----+---------+----------+-----+------+
| bid | bname   | baddress | pid | fid  |
+-----+---------+----------+-----+------+
|   1 | buyer 1 | chennai  |  27 |    1 |
|   4 | buyer 4 | kochi    |  27 |    1 |
|   2 | buyer 2 | kolar    |  28 |    1 |
|   1 | buyer 1 | chennai  |  29 |    2 |
|   3 | buyer 3 | kolkata  |  29 |    2 |
|   1 | buyer 1 | chennai  |  30 |    2 |
|   3 | buyer 3 | kolkata  |  30 |    2 |
|   4 | buyer 4 | kochi    |  30 |    2 |
|   2 | buyer 2 | kolar    |  31 |    3 |
|   4 | buyer 4 | kochi    |  36 |    2 |
|   3 | buyer 3 | kolkata  |  38 |    3 |
+-----+---------+----------+-----+------+
11 rows in set (0.010 sec)
```

Fig. 17: Using UNION operation

2) Display Buyer ID, Product name, info, price when price is greater than Rs.100 & when the product is a Fruit.

```
select mc.bid, fp.product, fp.pinfo, fp.price
-> from fproduct as fp inner join mycart as mc on mc.pid=fp.pid where fp.price>100
-> INTERSECT
-> select mc.bid, fp.product, fp.pinfo, fp.price
-> from fproduct as fp inner join mycart as mc on mc.pid=fp.pid where fp.pcat='Fruit';
```

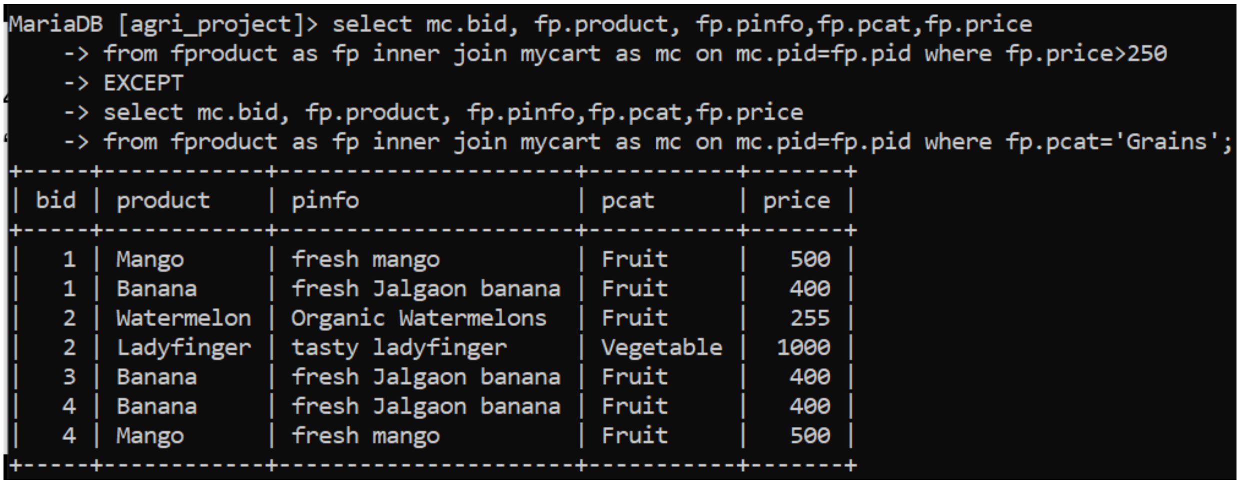


```
MariaDB [agri_project]> select mc.bid, fp.product, fp.pinfo, fp.price
    -> from fproduct as fp inner join mycart as mc on mc.pid=fp.pid where fp.price>100
    -> INTERSECT
    -> select mc.bid, fp.product, fp.pinfo, fp.price
    -> from fproduct as fp inner join mycart as mc on mc.pid=fp.pid where fp.pcat='Fruit';
+-----+------------+----------------------+-------+
| bid | product    | pinfo                | price |
+-----+------------+----------------------+-------+
|   1 | Mango      | fresh mango          |   500 |
|   1 | Banana     | fresh Jalgaon banana |   400 |
|   2 | Watermelon | Organic Watermelons  |   255 |
|   3 | Banana     | fresh Jalgaon banana |   400 |
|   4 | Banana     | fresh Jalgaon banana |   400 |
|   4 | Mango      | fresh mango          |   500 |
+-----+------------+----------------------+-------+
```

Fig. 18: Using INTERSECT operation

3. Display Buyer ID, Product name, info, price, category when price is greater than Rs.250 only when the product is not categorized as ‘Grains’.

```
select mc.bid, fp.product, fp.pinfo,fp.pcat,fp.price
-> from fproduct as fp inner join mycart as mc on mc.pid=fp.pid where fp.price>250
-> EXCEPT
-> select mc.bid, fp.product, fp.pinfo,fp.pcat,fp.price
-> from fproduct as fp inner join mycart as mc on mc.pid=fp.pid where fp.pcat='Grains';
```

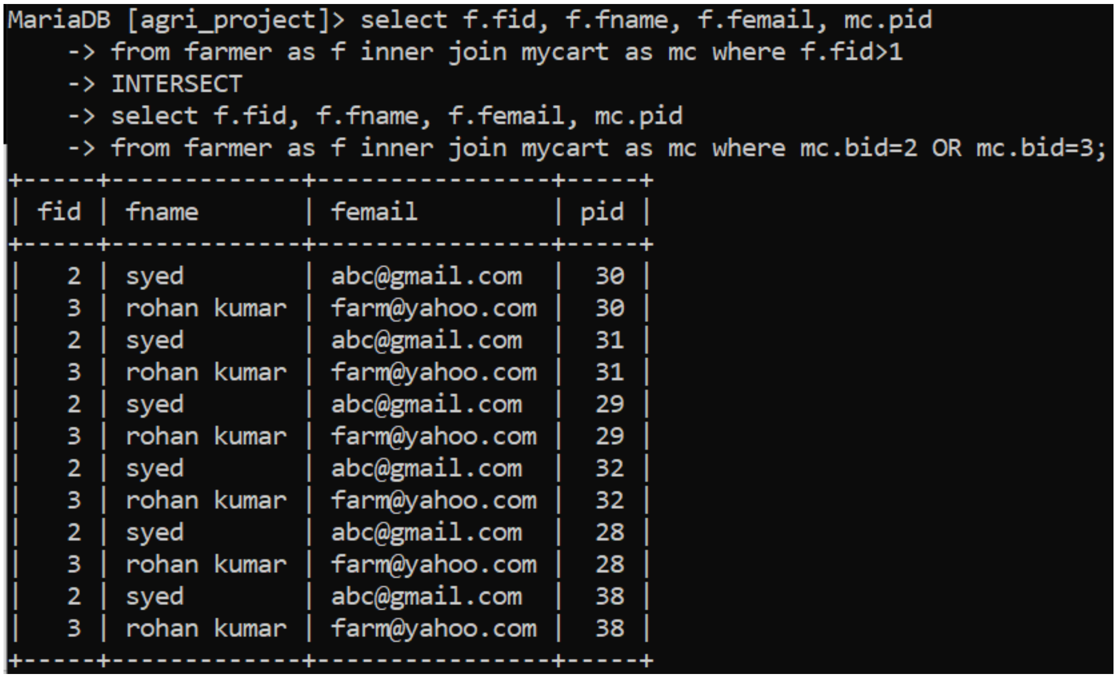


```
MariaDB [agri_project]> select mc.bid, fp.product, fp.pinfo,fp.pcat,fp.price
    -> from fproduct as fp inner join mycart as mc on mc.pid=fp.pid where fp.price>250
    -> EXCEPT
    -> select mc.bid, fp.product, fp.pinfo,fp.pcat,fp.price
    -> from fproduct as fp inner join mycart as mc on mc.pid=fp.pid where fp.pcat='Grains';
+-----+------------+---------------------+-----------+-------+
| bid | product    | pinfo               | pcat      | price |
+-----+------------+---------------------+-----------+-------+
|   1 | Mango      | fresh mango         | Fruit     |   500 |
|   1 | Banana     | fresh Jalgaon banana | Fruit     |   400 |
|   2 | Watermelon | Organic Watermelons | Fruit     |   255 |
|   2 | Ladyfinger | tasty ladyfinger    | Vegetable |  1000 |
|   3 | Banana     | fresh Jalgaon banana | Fruit     |   400 |
|   4 | Banana     | fresh Jalgaon banana | Fruit     |   400 |
|   4 | Mango      | fresh mango         | Fruit     |   500 |
+-----+------------+---------------------+-----------+-------+
```

Fig. 19: Using EXCEPT operation

4. Display the Farmer ID, Name, Email, Product ID when Farmer ID is greater than 1 and when only Buyer 2 or Buyer 3 buys the product.

select f.fid, f.fname, f.femail, mc.pid
-> from farmer as f inner join mycart as mc where f.fid>1
-> INTERSECT
-> select f.fid, f.fname, f.femail, mc.pid
-> from farmer as f inner join mycart as mc where mc.bid=2 OR mc.bid=3;



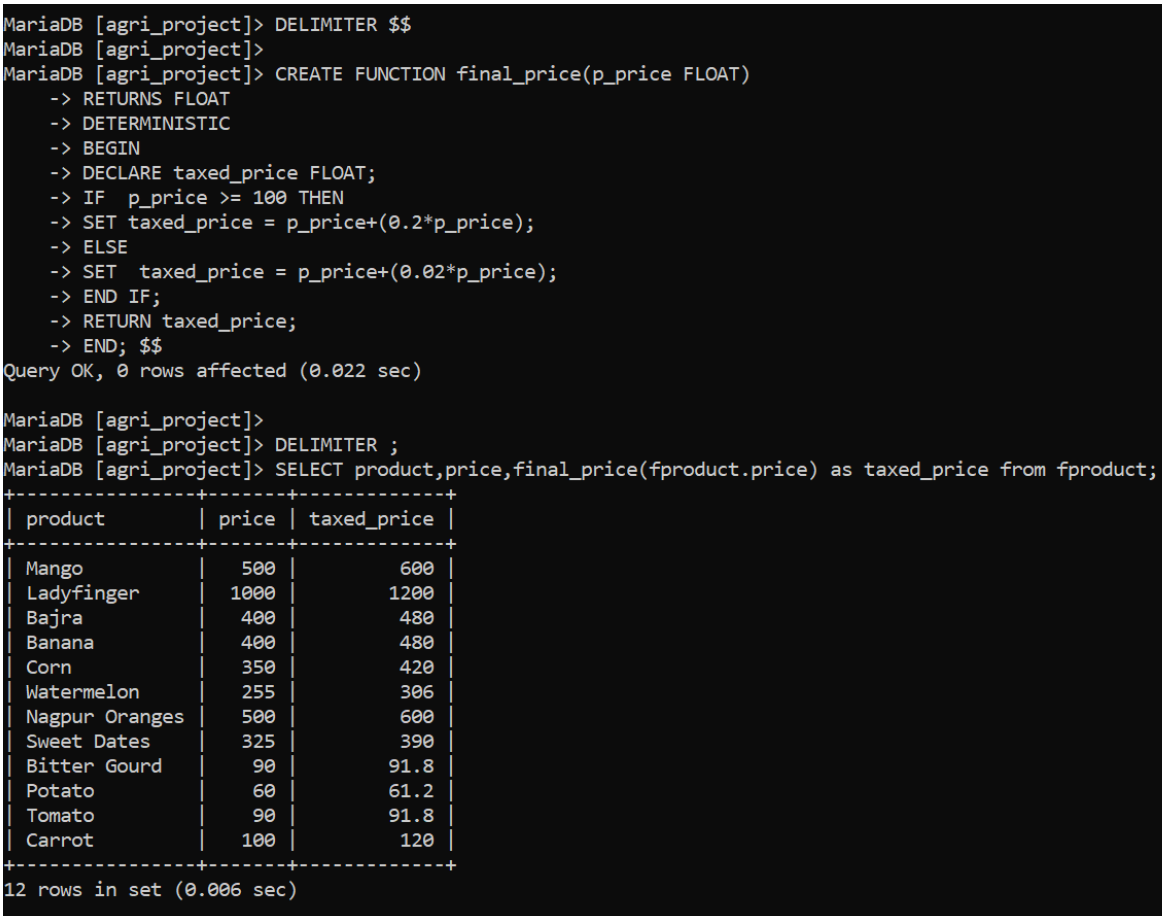
```
MariaDB [agri_project]> select f.fid, f.fname, f.femail, mc.pid
    -> from farmer as f inner join mycart as mc where f.fid>1
    -> INTERSECT
    -> select f.fid, f.fname, f.femail, mc.pid
    -> from farmer as f inner join mycart as mc where mc.bid=2 OR mc.bid=3;
+-----+------------+----------------+-----+
| fid | fname      | femail         | pid |
+-----+------------+----------------+-----+
|   2 | syed       | abc@gmail.com  |  30 |
|   3 | rohan kumar | farm@yahoo.com |  30 |
|   2 | syed       | abc@gmail.com  |  31 |
|   3 | rohan kumar | farm@yahoo.com |  31 |
|   2 | syed       | abc@gmail.com  |  29 |
|   3 | rohan kumar | farm@yahoo.com |  29 |
|   2 | syed       | abc@gmail.com  |  32 |
|   3 | rohan kumar | farm@yahoo.com |  32 |
|   2 | syed       | abc@gmail.com  |  28 |
|   3 | rohan kumar | farm@yahoo.com |  28 |
|   2 | syed       | abc@gmail.com  |  38 |
|   3 | rohan kumar | farm@yahoo.com |  38 |
+-----+------------+----------------+-----+
```

Fig. 20: Using INTERSECT operation

Functions and Procedures

1) Function to calculate the final taxed price of the products of the farm at rate of 20% when price is greater than Rs.100 & rate of 2% when price is lesser than Rs.100.

```
DELIMITER $$
CREATE FUNCTION final_price(p_price FLOAT)
RETURNS FLOAT
DETERMINISTIC
BEGIN
DECLARE taxed_price FLOAT;
IF  p_price >= 100 THEN
SET taxed_price = p_price+(0.2*p_price);
ELSE
SET  taxed_price = p_price+(0.02*p_price);
END IF;
RETURN taxed_price;
END; $$

DELIMITER ;
```

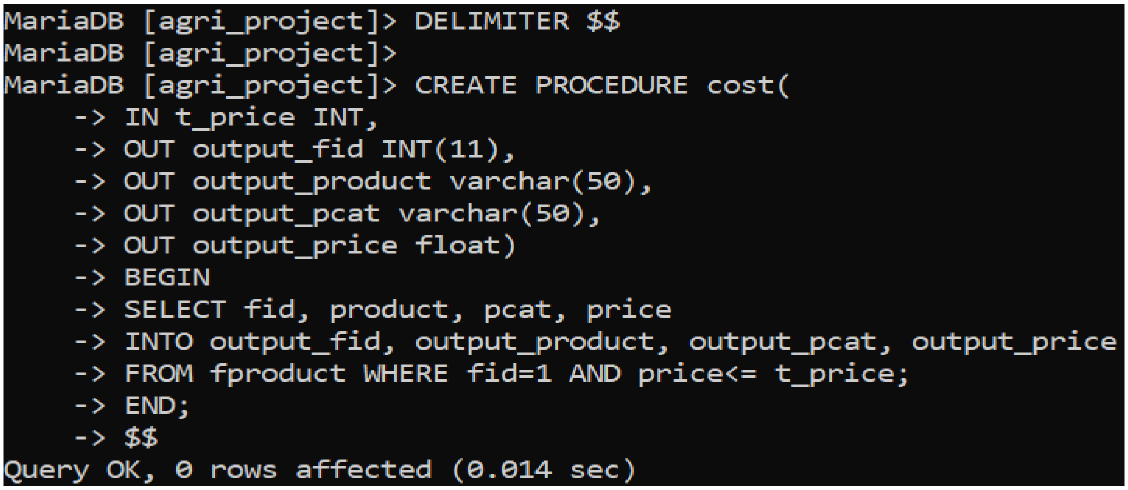


```
MariaDB [agri_project]> DELIMITER $$
MariaDB [agri_project]>
MariaDB [agri_project]> CREATE FUNCTION final_price(p_price FLOAT)
    -> RETURNS FLOAT
    -> DETERMINISTIC
    -> BEGIN
    -> DECLARE taxed_price FLOAT;
    -> IF  p_price >= 100 THEN
    -> SET taxed_price = p_price+(0.2*p_price);
    -> ELSE
    -> SET  taxed_price = p_price+(0.02*p_price);
    -> END IF;
    -> RETURN taxed_price;
    -> END; $$
Query OK, 0 rows affected (0.022 sec)

MariaDB [agri_project]>
MariaDB [agri_project]> DELIMITER ;
MariaDB [agri_project]> SELECT product,price,final_price(fproduct.price) as taxed_price from fproduct;
+----------------+-------+-------------+
| product        | price | taxed_price |
+----------------+-------+-------------+
| Mango          |   500 |         600 |
| Ladyfinger     |  1000 |        1200 |
| Bajra          |   400 |         480 |
| Banana         |   400 |         480 |
| Corn           |   350 |         420 |
| Watermelon     |   255 |         306 |
| Nagpur Oranges |   500 |         600 |
| Sweet Dates    |   325 |         390 |
| Bitter Gourd   |    90 |        91.8 |
| Potato         |    60 |        61.2 |
| Tomato         |    90 |        91.8 |
| Carrot         |   100 |         120 |
+----------------+-------+-------------+
12 rows in set (0.006 sec)
```

Fig. 21: Function for final taxed price

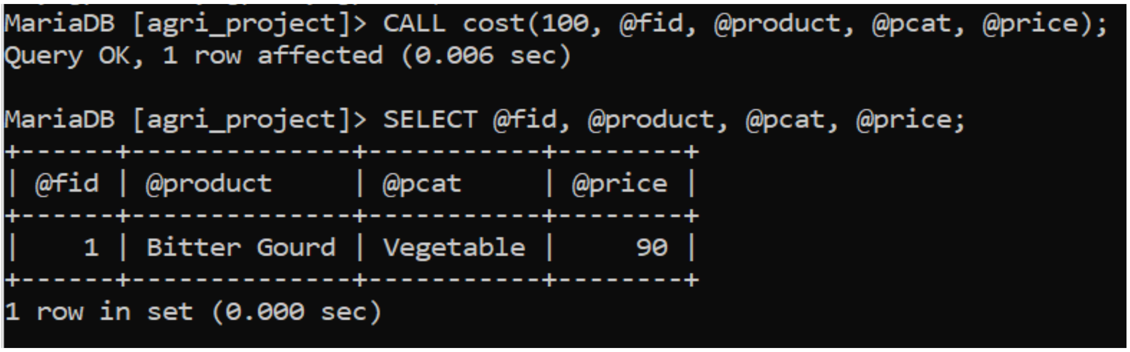
2) Procedure to list the farm produce which is from Farmer ID = 1 & is below the given limit price.

```
 DELIMITER $$
CREATE PROCEDURE cost(
IN t_price INT,
OUT output_fid INT(11),
OUT output_product varchar(50),
OUT output_pcat varchar(50),
OUT output_price float)
BEGIN
SELECT fid, product, pcat, price
INTO output_fid, output_product, output_pcat, output_price
FROM fproduct WHERE fid=1 AND price<= t_price;
END;
$$
```



```
MariaDB [agri_project]> DELIMITER $$
MariaDB [agri_project]>
MariaDB [agri_project]> CREATE PROCEDURE cost(
    -> IN t_price INT,
    -> OUT output_fid INT(11),
    -> OUT output_product varchar(50),
    -> OUT output_pcat varchar(50),
    -> OUT output_price float)
    -> BEGIN
    -> SELECT fid, product, pcat, price
    -> INTO output_fid, output_product, output_pcat, output_price
    -> FROM fproduct WHERE fid=1 AND price<= t_price;
    -> END;
    -> $$
Query OK, 0 rows affected (0.014 sec)
```

Fig. 21.1: Procedure for farm produce



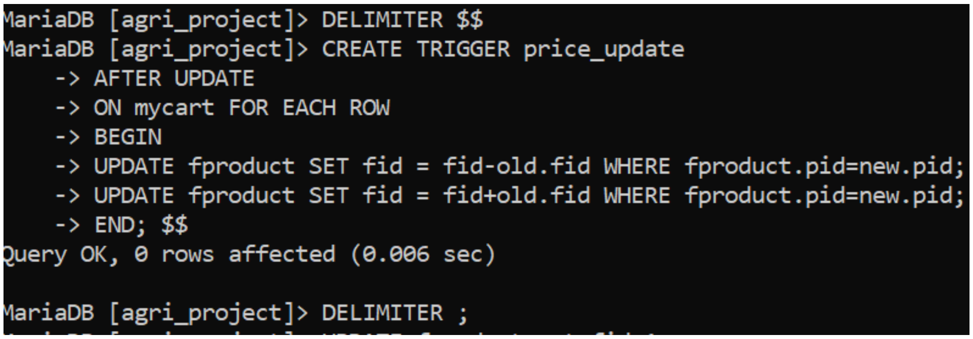
```
MariaDB [agri_project]> CALL cost(100, @fid, @product, @pcat, @price);
Query OK, 1 row affected (0.006 sec)

MariaDB [agri_project]> SELECT @fid, @product, @pcat, @price;
+------+-------------+-----------+--------+
| @fid | @product    | @pcat     | @price |
+------+-------------+-----------+--------+
|    1 | Bitter Gourd | Vegetable |     90 |
+------+-------------+-----------+--------+
1 row in set (0.000 sec)
```

Fig. 21.2: Procedure for farm produce

Triggers and Cursors

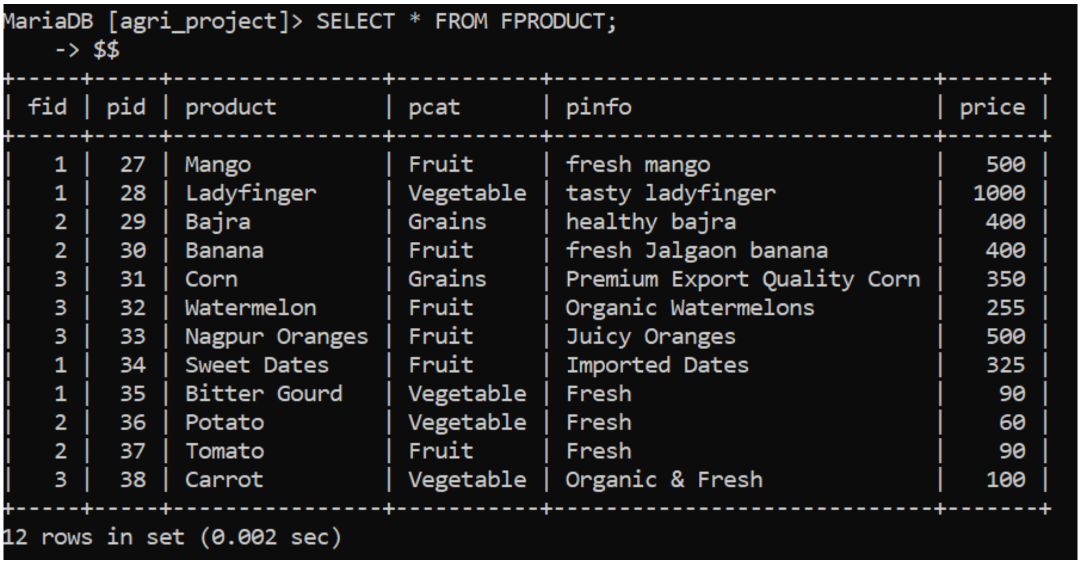
1) Trigger price_update updates fproduct table fid on updating fid in mycart table.

CREATE TRIGGER price_update
-> *AFTER UPDATE*
-> *ON mycart FOR EACH ROW*
-> *BEGIN*
-> *UPDATE fproduct SET fid = fid-old.fid WHERE fproduct.pid=new.pid;*
-> *UPDATE fproduct SET fid = fid+old.fid WHERE fproduct.pid=new.pid;*
-> *END; $$*



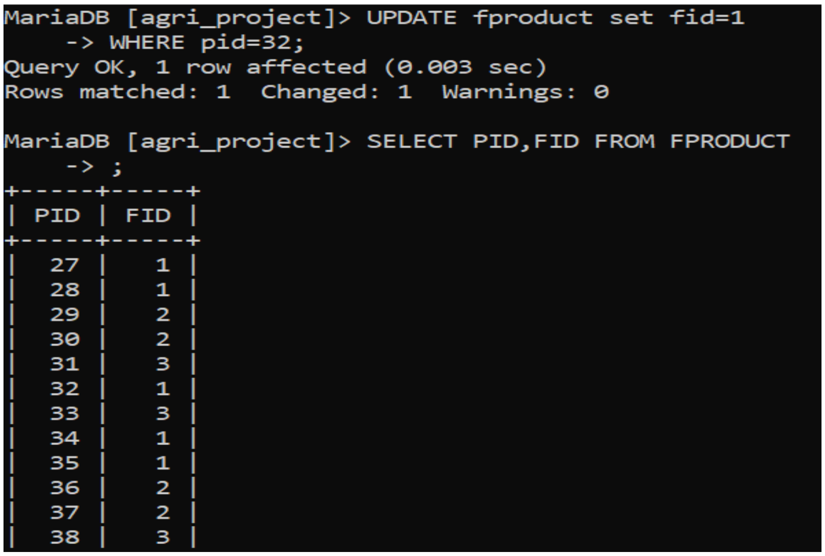
```
MariaDB [agri_project]> DELIMITER $$
MariaDB [agri_project]> CREATE TRIGGER price_update
    -> AFTER UPDATE
    -> ON mycart FOR EACH ROW
    -> BEGIN
    -> UPDATE fproduct SET fid = fid-old.fid WHERE fproduct.pid=new.pid;
    -> UPDATE fproduct SET fid = fid+old.fid WHERE fproduct.pid=new.pid;
    -> END; $$
Query OK, 0 rows affected (0.006 sec)

MariaDB [agri_project]> DELIMITER ;
```

Fig. 22.1: Trigger for price updating



```
MariaDB [agri_project]> SELECT * FROM FPRODUCT;
    -> $$
+-----+-----+----------------+-----------+------------------------------+-------+
| fid | pid | product        | pcat      | pinfo                        | price |
+-----+-----+----------------+-----------+------------------------------+-------+
|   1 |  27 | Mango          | Fruit     | fresh mango                  |   500 |
|   1 |  28 | Ladyfinger     | Vegetable | tasty ladyfinger             |  1000 |
|   2 |  29 | Bajra          | Grains    | healthy bajra                |   400 |
|   2 |  30 | Banana         | Fruit     | fresh Jalgaon banana         |   400 |
|   3 |  31 | Corn           | Grains    | Premium Export Quality Corn  |   350 |
|   3 |  32 | Watermelon     | Fruit     | Organic Watermelons          |   255 |
|   3 |  33 | Nagpur Oranges | Fruit     | Juicy Oranges                |   500 |
|   1 |  34 | Sweet Dates    | Fruit     | Imported Dates               |   325 |
|   1 |  35 | Bitter Gourd   | Vegetable | Fresh                        |    90 |
|   2 |  36 | Potato         | Vegetable | Fresh                        |    60 |
|   2 |  37 | Tomato         | Fruit     | Fresh                        |    90 |
|   3 |  38 | Carrot         | Vegetable | Organic & Fresh              |   100 |
+-----+-----+----------------+-----------+------------------------------+-------+
12 rows in set (0.002 sec)
```

Fig. 22.2: Trigger for price updating

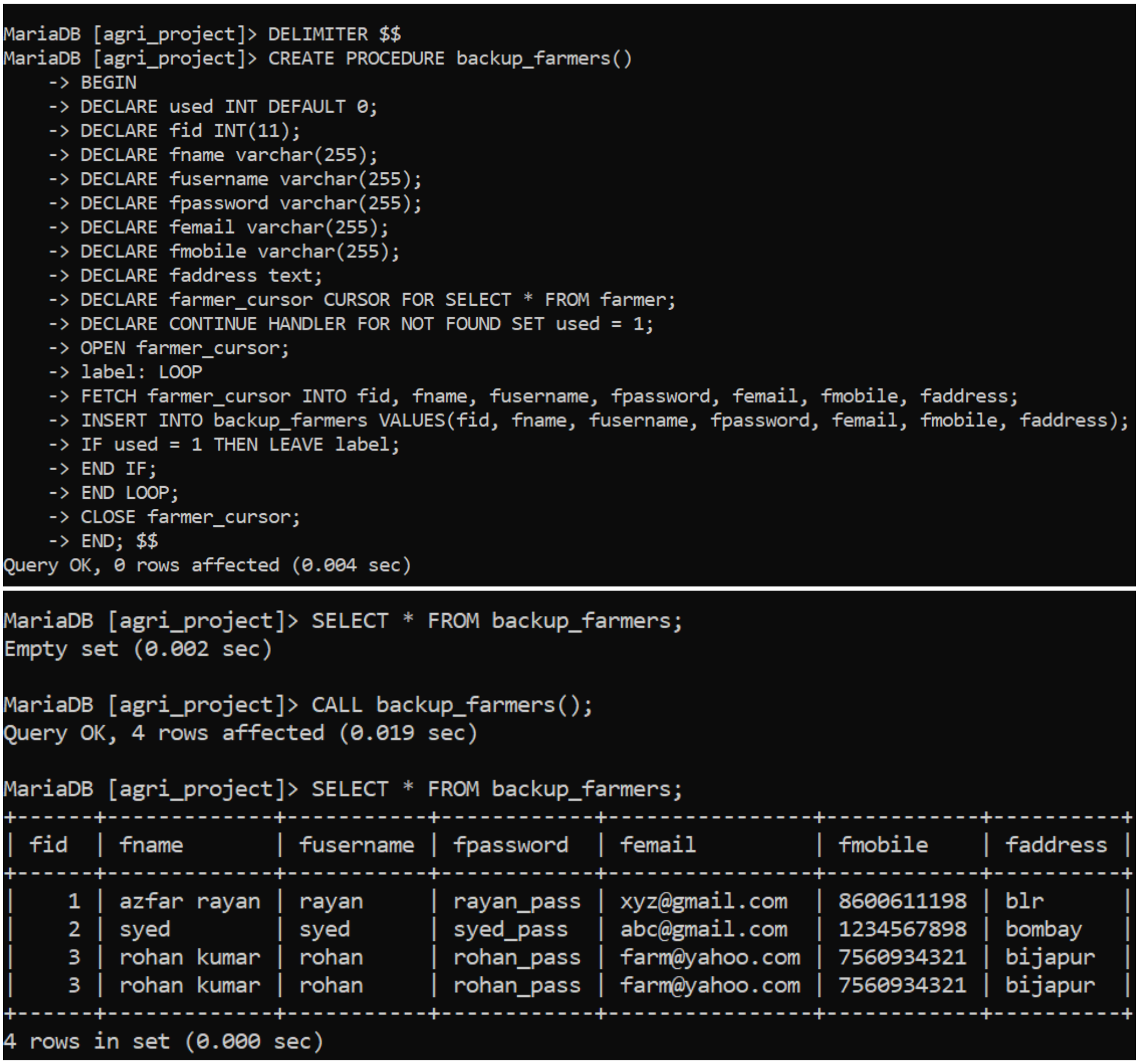


```
MariaDB [agri_project]> UPDATE fproduct set fid=1
    -> WHERE pid=32;
Query OK, 1 row affected (0.003 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [agri_project]> SELECT PID,FID FROM FPRODUCT
    -> ;
+-----+-----+
| PID | FID |
+-----+-----+
|  27 |   1 |
|  28 |   1 |
|  29 |   2 |
|  30 |   2 |
|  31 |   3 |
|  32 |   1 |
|  33 |   3 |
|  34 |   1 |
|  35 |   1 |
|  36 |   2 |
|  37 |   2 |
|  38 |   3 |
```

Fig. 22.3: Trigger for price updating

2) Cursor farmer_cursor declared in procedure backup_farmer( ) copies all details in farmer table which is inserted in the new backup_farmers table.

```
DELIMITER $$
CREATE PROCEDURE backup_farmers()
BEGIN
DECLARE used INT DEFAULT 0;
DECLARE fid INT(11);
DECLARE fname varchar(255);
DECLARE fusername varchar(255);
DECLARE fpassword varchar(255);
DECLARE femail varchar(255);
DECLARE fmobile varchar(255);
DECLARE faddress text;
DECLARE farmer_cursor CURSOR FOR SELECT * FROM farmer;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET used = 1;
OPEN farmer_cursor;
label: LOOP
FETCH farmer_cursor INTO fid, fname, fusername, fpassword, femail, fmobile, faddress;
INSERT INTO backup_farmers VALUES(fid, fname, fusername, fpassword, femail, fmobile, faddress);
IF used = 1 THEN LEAVE label;
END IF;
END LOOP;
CLOSE farmer_cursor;
END $$
DELIMITER ;
```



```
MariaDB [agri_project]> DELIMITER $$
MariaDB [agri_project]> CREATE PROCEDURE backup_farmers()
    -> BEGIN
    -> DECLARE used INT DEFAULT 0;
    -> DECLARE fid INT(11);
    -> DECLARE fname varchar(255);
    -> DECLARE fusername varchar(255);
    -> DECLARE fpassword varchar(255);
    -> DECLARE femail varchar(255);
    -> DECLARE fmobile varchar(255);
    -> DECLARE faddress text;
    -> DECLARE farmer_cursor CURSOR FOR SELECT * FROM farmer;
    -> DECLARE CONTINUE HANDLER FOR NOT FOUND SET used = 1;
    -> OPEN farmer_cursor;
    -> label: LOOP
    -> FETCH farmer_cursor INTO fid, fname, fusername, fpassword, femail, fmobile, faddress;
    -> INSERT INTO backup_farmers VALUES(fid, fname, fusername, fpassword, femail, fmobile, faddress);
    -> IF used = 1 THEN LEAVE label;
    -> END IF;
    -> END LOOP;
    -> CLOSE farmer_cursor;
    -> END; $$
Query OK, 0 rows affected (0.004 sec)
```

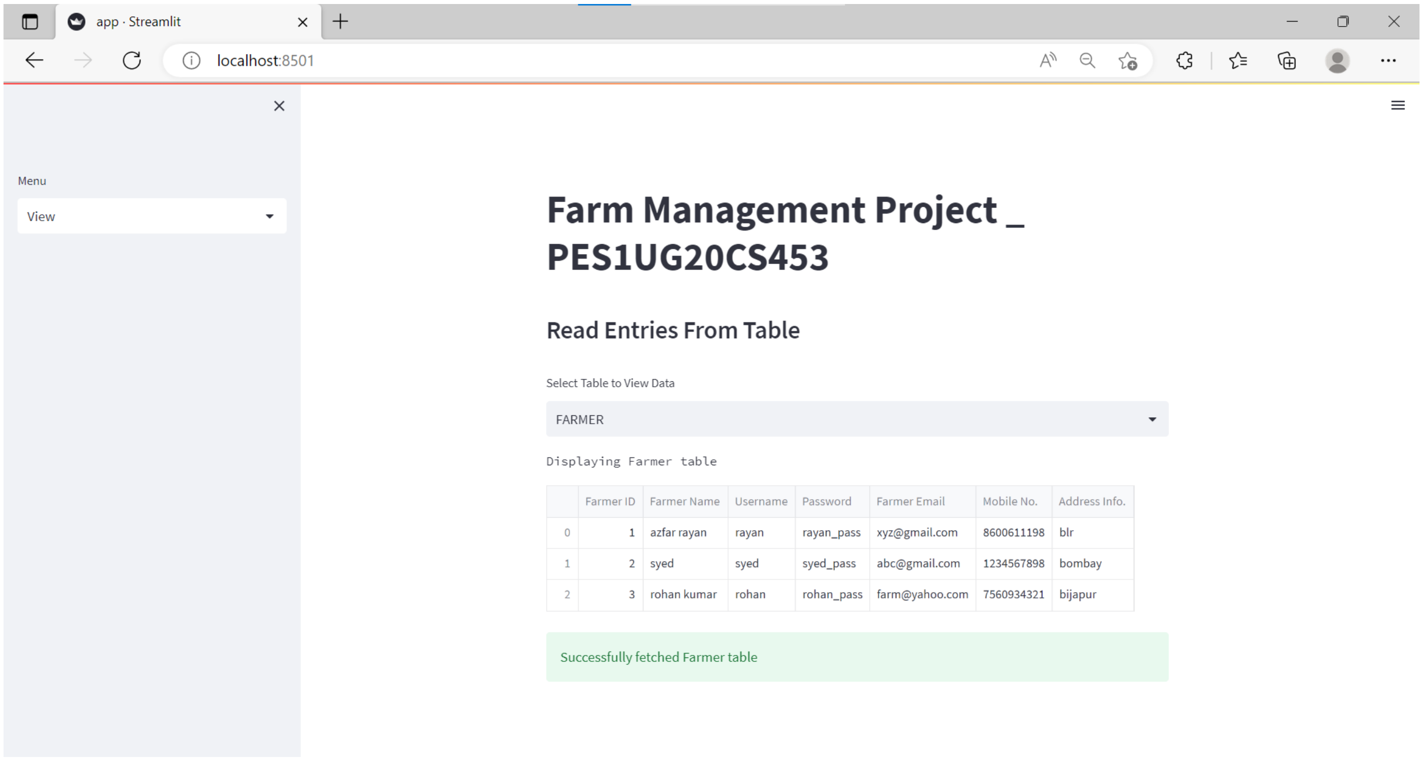
```
MariaDB [agri_project]> SELECT * FROM backup_farmers;
Empty set (0.002 sec)

MariaDB [agri_project]> CALL backup_farmers();
Query OK, 4 rows affected (0.019 sec)

MariaDB [agri_project]> SELECT * FROM backup_farmers;
+------+-------------+-----------+------------+----------------+------------+----------+
| fid  | fname       | fusername | fpassword  | femail         | fmobile    | faddress |
+------+-------------+-----------+------------+----------------+------------+----------+
|    1 | azfar rayan | rayan     | rayan_pass | xyz@gmail.com  | 8600611198 | blr      |
|    2 | syed        | syed      | syed_pass  | abc@gmail.com  | 1234567898 | bombay   |
|    3 | rohan kumar | rohan     | rohan_pass | farm@yahoo.com | 7560934321 | bijapur  |
|    3 | rohan kumar | rohan     | rohan_pass | farm@yahoo.com | 7560934321 | bijapur  |
+------+-------------+-----------+------------+----------------+------------+----------+
4 rows in set (0.000 sec)
```

Fig. 23: Cursor for extracting farmer details

Developing a Frontend

The frontend should support

1. Addition, Modification and Deletion of records from any chosen table
2. There should be an window to accept and run any SQL statement and display the result



Farm Management Project _ PES1UG20CS453

Read Entries From Table

Select Table to View Data

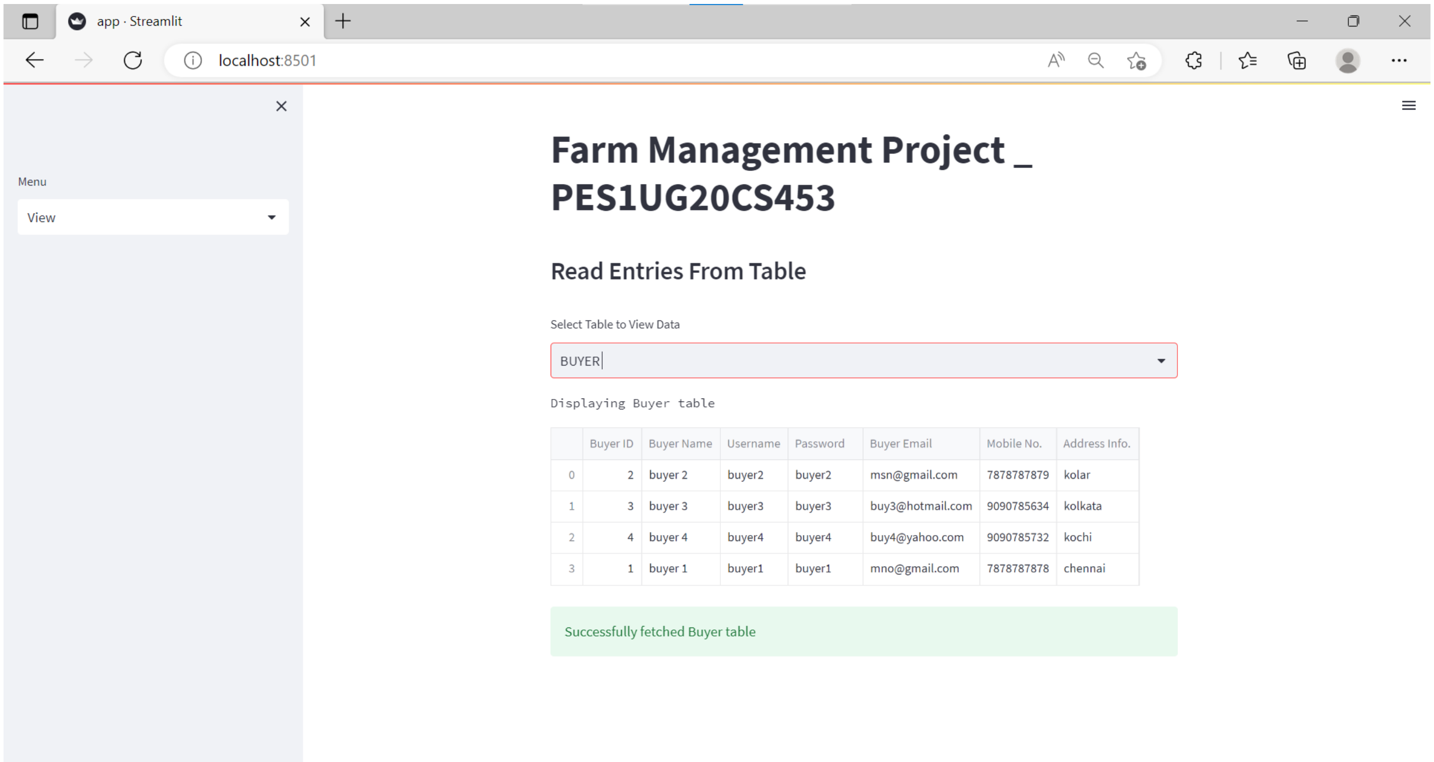
FARMER

```
Displaying Farmer table
```

	Farmer ID	Farmer Name	Username	Password	Farmer Email	Mobile No.	Address Info.
0	1	azfar rayan	rayan	rayan_pass	xyz@gmail.com	8600611198	blr
1	2	syed	syed	syed_pass	abc@gmail.com	1234567898	bombay
2	3	rohan kumar	rohan	rohan_pass	farm@yahoo.com	7560934321	bijapur

Successfully fetched Farmer table

Fig. 24.1: Viewing tables



Menu

View

Farm Management Project _ PES1UG20CS453

Read Entries From Table

Select Table to View Data

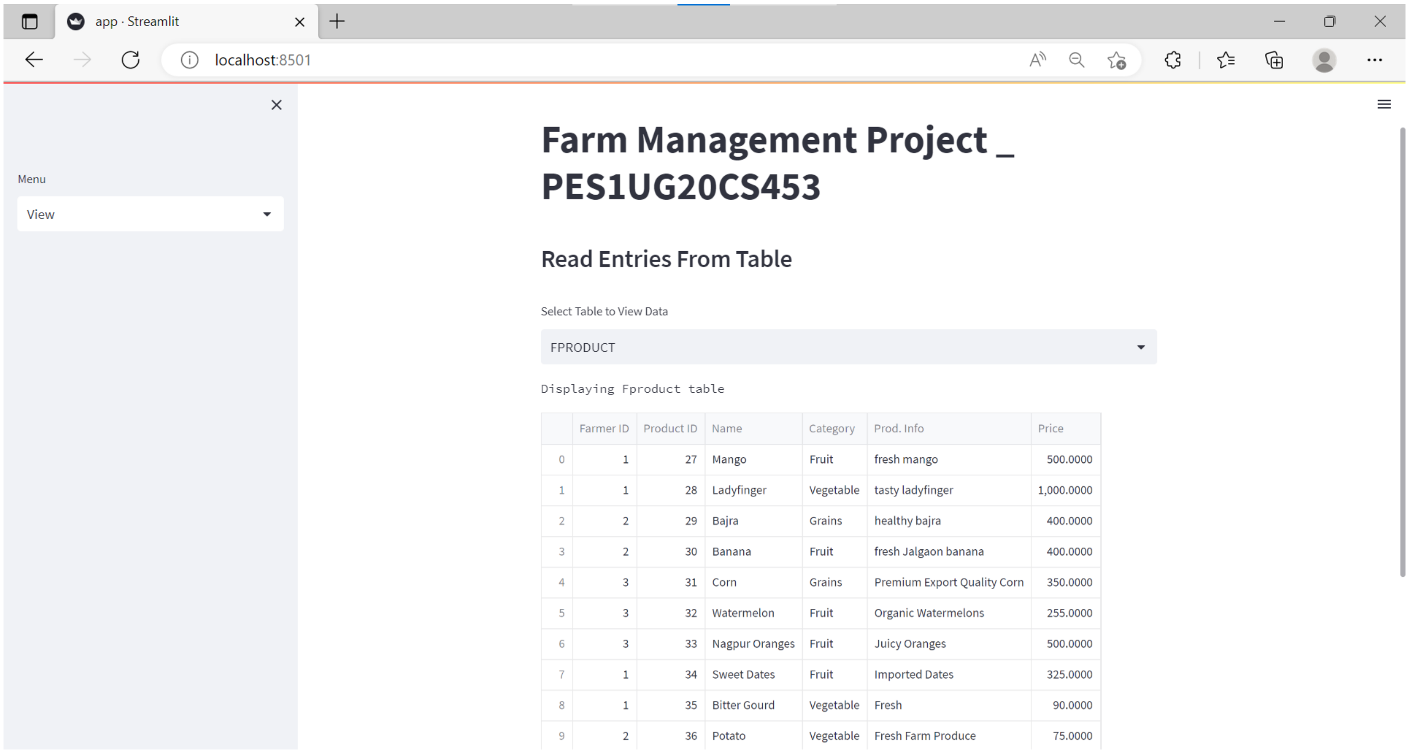
BUYER

```
Displaying Buyer table
```

	Buyer ID	Buyer Name	Username	Password	Buyer Email	Mobile No.	Address Info.
0	2	buyer 2	buyer2	buyer2	msn@gmail.com	7878787879	kolar
1	3	buyer 3	buyer3	buyer3	buy3@hotmail.com	9090785634	kolkata
2	4	buyer 4	buyer4	buyer4	buy4@yahoo.com	9090785732	kochi
3	1	buyer 1	buyer1	buyer1	mno@gmail.com	7878787878	chennai

Successfully fetched Buyer table

Fig. 24.2: Viewing tables



Menu

View

Farm Management Project _ PES1UG20CS453

Read Entries From Table

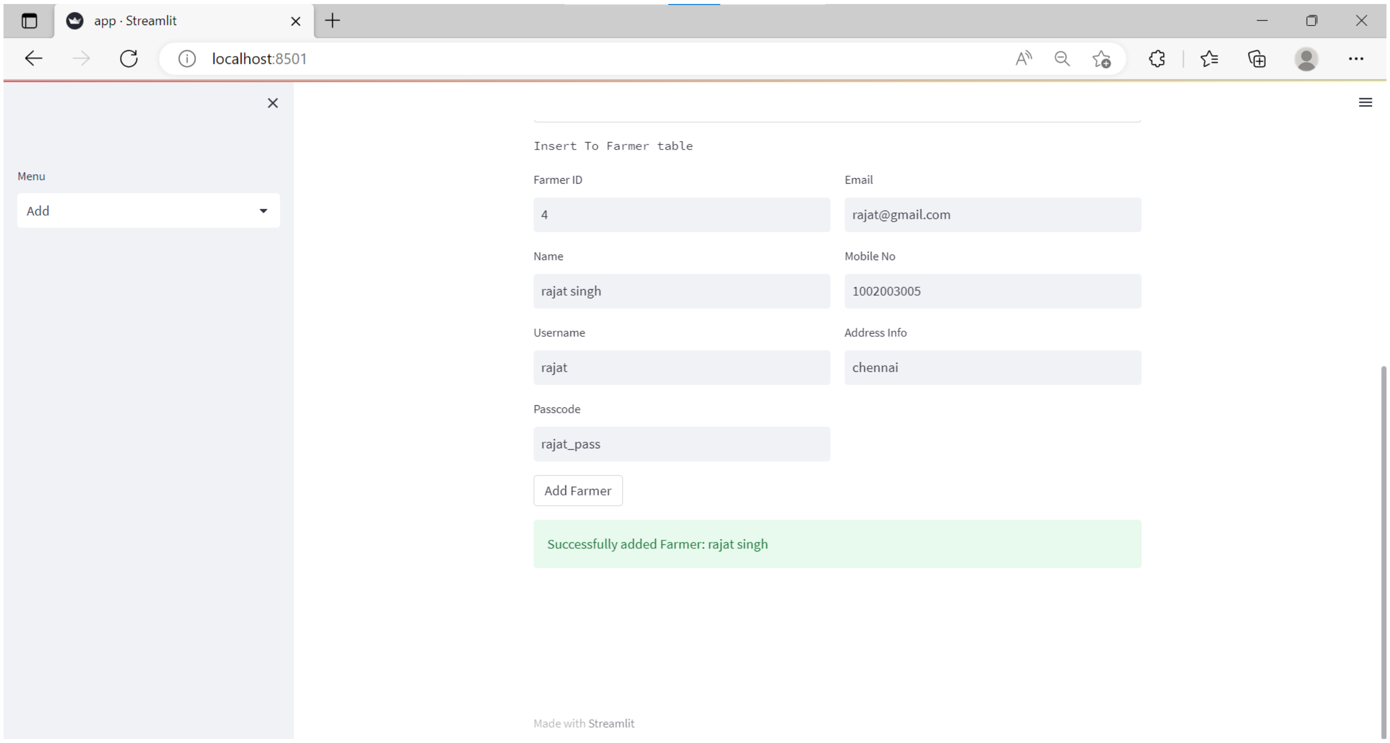
Select Table to View Data

FPRODUCT

```
Displaying Fproduct table
```

	Farmer ID	Product ID	Name	Category	Prod. Info	Price
0	1	27	Mango	Fruit	fresh mango	500.0000
1	1	28	Ladyfinger	Vegetable	tasty ladyfinger	1,000.0000
2	2	29	Bajra	Grains	healthy bajra	400.0000
3	2	30	Banana	Fruit	fresh Jalgaon banana	400.0000
4	3	31	Corn	Grains	Premium Export Quality Corn	350.0000
5	3	32	Watermelon	Fruit	Organic Watermelons	255.0000
6	3	33	Nagpur Oranges	Fruit	Juicy Oranges	500.0000
7	1	34	Sweet Dates	Fruit	Imported Dates	325.0000
8	1	35	Bitter Gourd	Vegetable	Fresh	90.0000
9	2	36	Potato	Vegetable	Fresh Farm Produce	75.0000

Fig. 24.3: Viewing tables



Insert To Farmer table

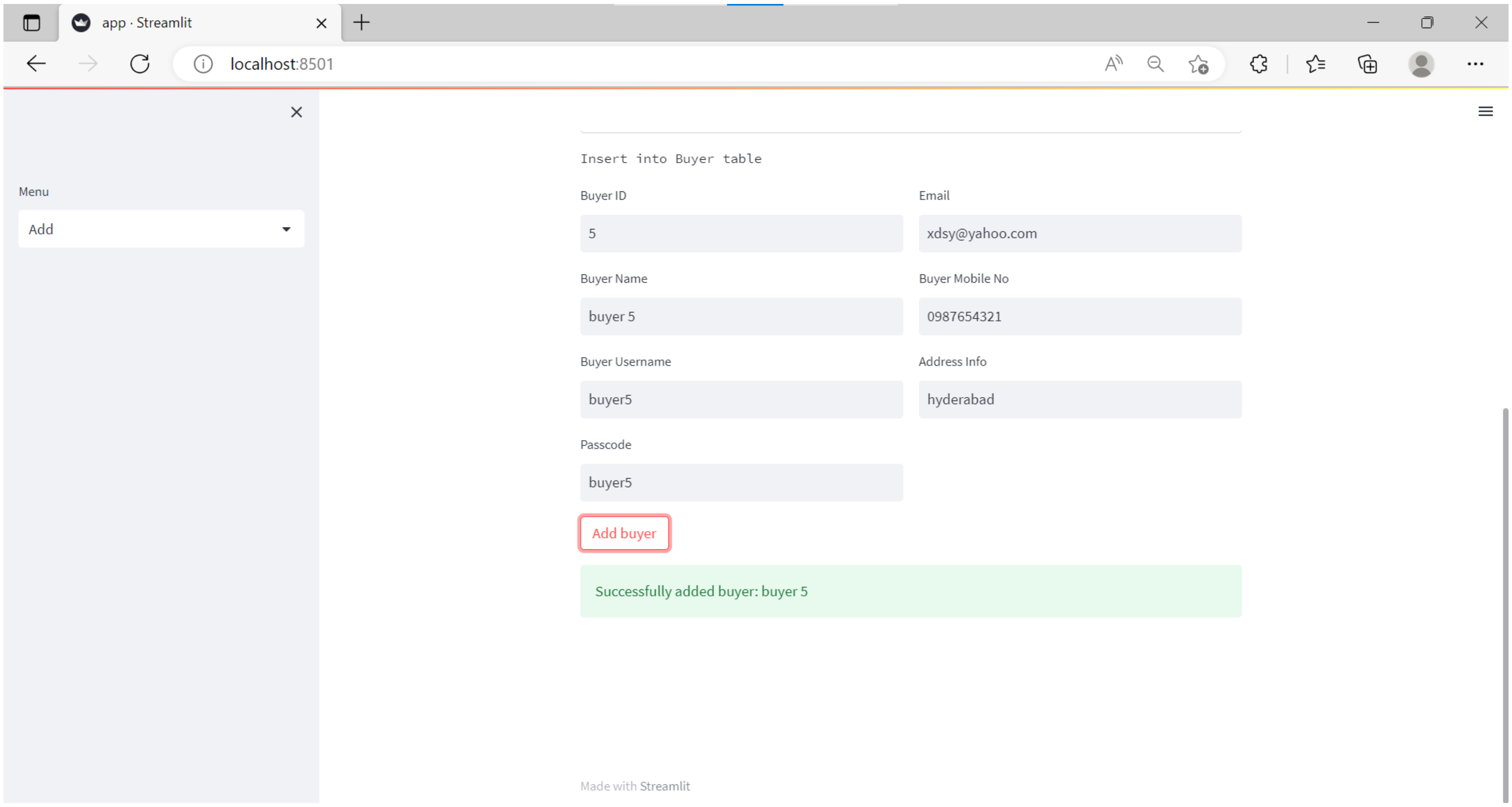
Farmer ID: 4
Email: rajat@gmail.com
Name: rajat singh
Mobile No: 1002003005
Username: rajat
Address Info: chennai
Passcode: rajat_pass

Add Farmer

Successfully added Farmer: rajat singh

Made with Streamlit

Fig. 25.1: Inserting values in tables



Insert into Buyer table

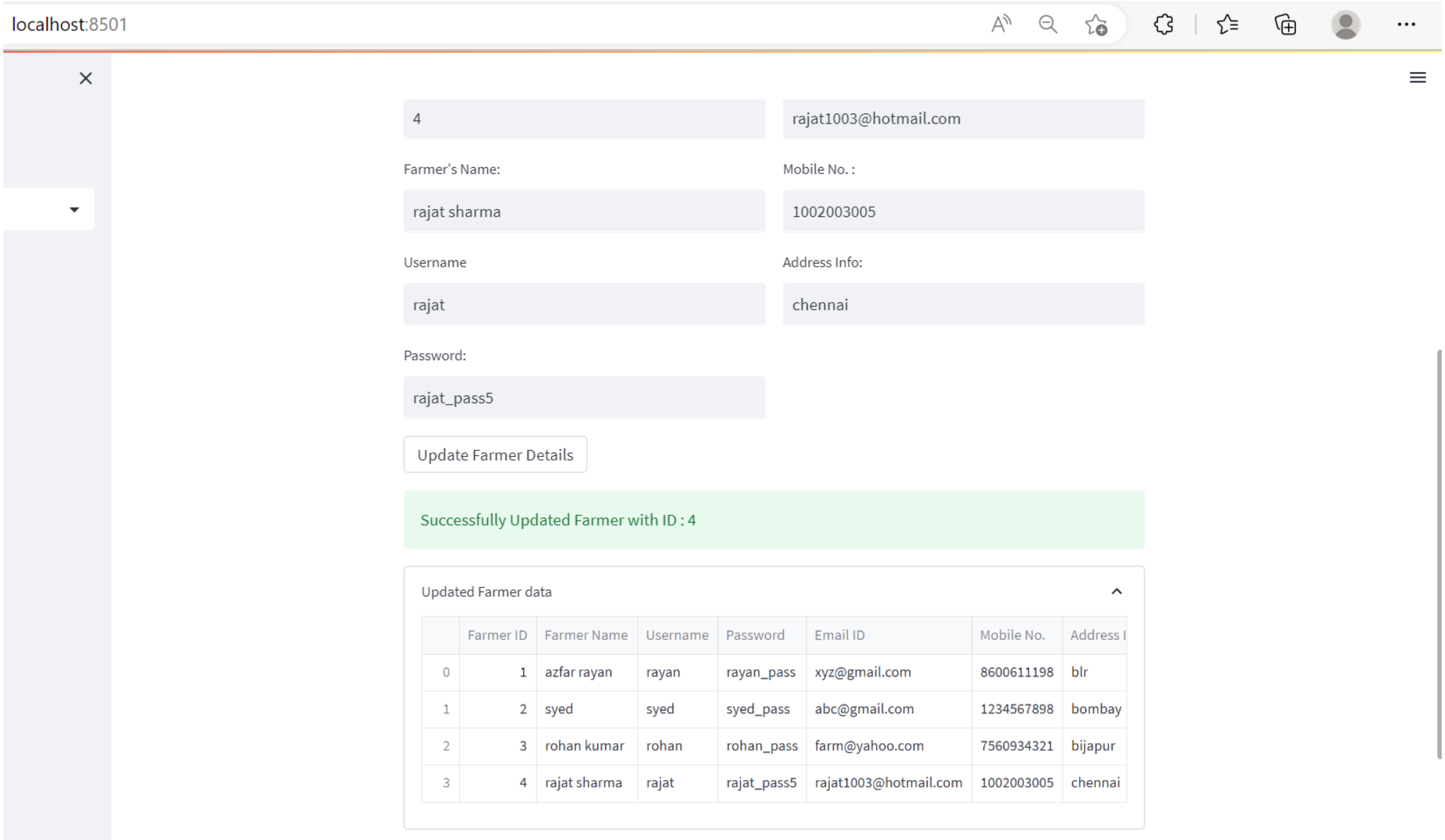
Buyer ID: 5
Email: xdsy@yahoo.com
Buyer Name: buyer 5
Buyer Mobile No: 0987654321
Buyer Username: buyer5
Address Info: hyderabad
Passcode: buyer5

Add buyer

Successfully added buyer: buyer 5

Made with Streamlit

Fig. 25.2: Inserting values in tables



4

rajat1003@hotmail.com

Farmer's Name: rajat sharma

Mobile No. : 1002003005

Username rajat

Address Info: chennai

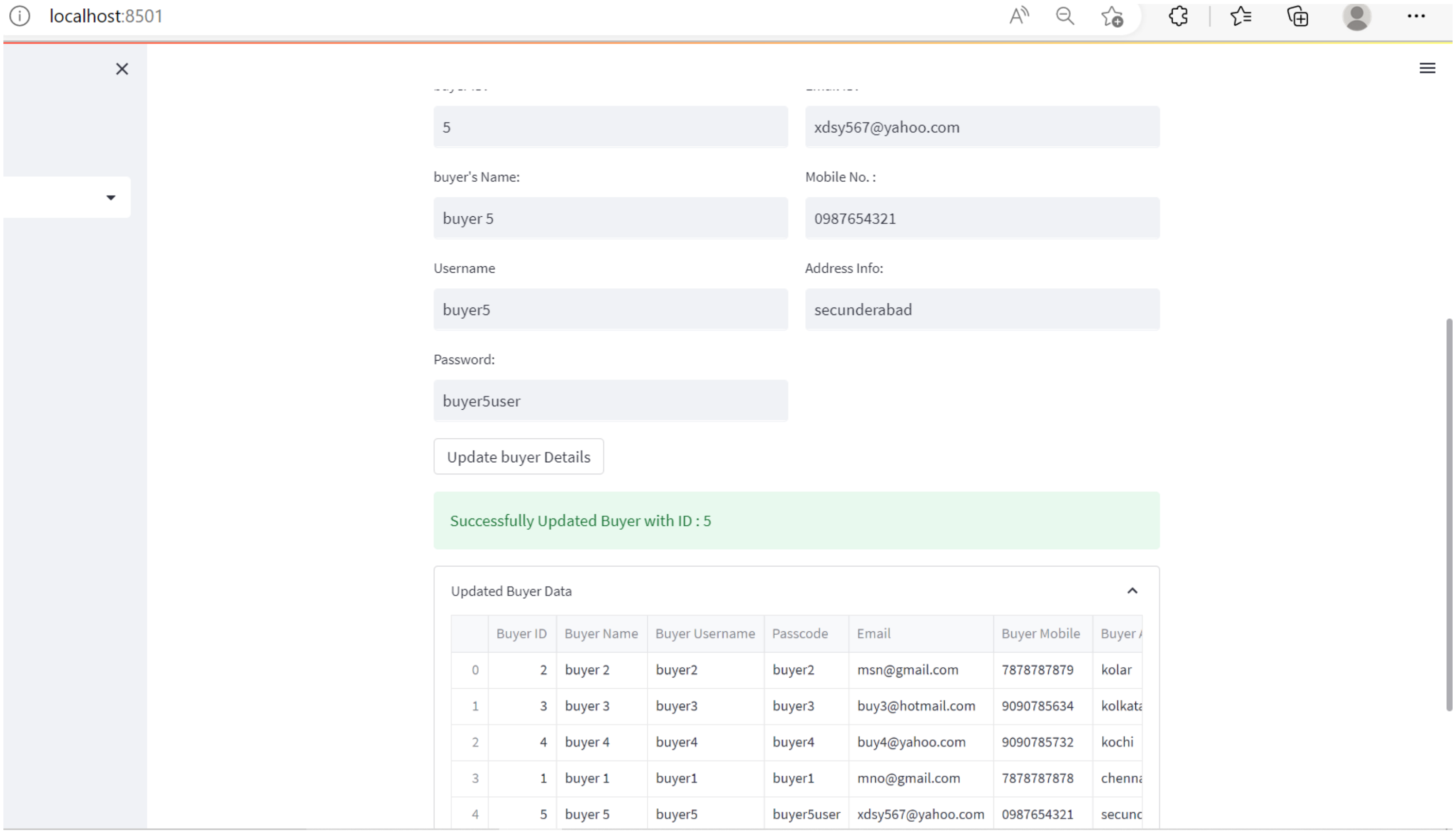
Password: rajat_pass5

Successfully Updated Farmer with ID : 4

Updated Farmer data

	Farmer ID	Farmer Name	Username	Password	Email ID	Mobile No.	Address I
0	1	azfar rayan	rayan	rayan_pass	xyz@gmail.com	8600611198	blr
1	2	syed	syed	syed_pass	abc@gmail.com	1234567898	bombay
2	3	rohan kumar	rohan	rohan_pass	farm@yahoo.com	7560934321	bijapur
3	4	rajat sharma	rajat	rajat_pass5	rajat1003@hotmail.com	1002003005	chennai

Fig. 26.1: Updating values in tables



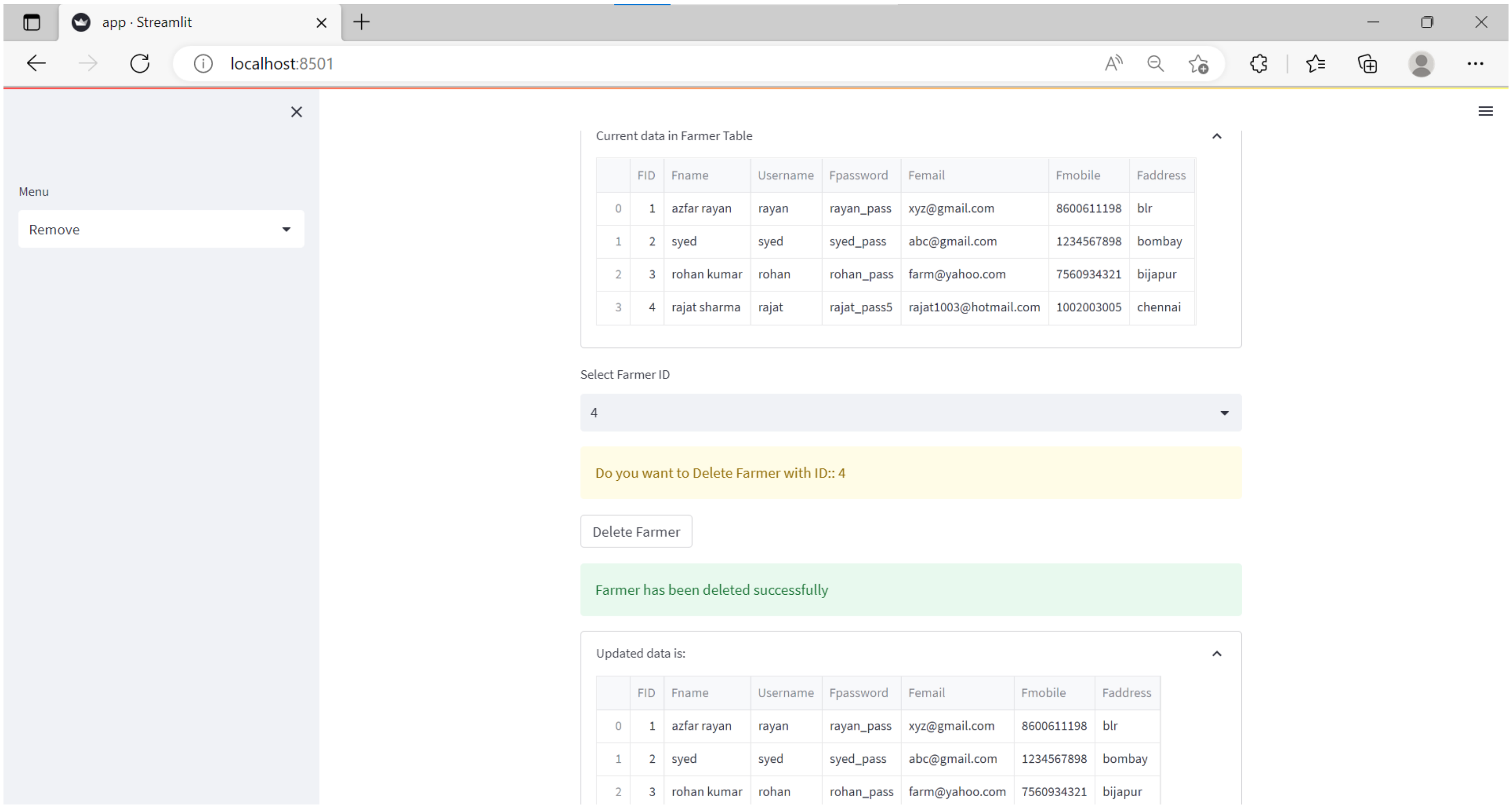
5	xdsy567@yahoo.com
buyer's Name:	Mobile No. :
buyer 5	0987654321
Username	Address Info:
buyer5	secunderabad
Password:	
buyer5user	

Successfully Updated Buyer with ID : 5

Updated Buyer Data

	Buyer ID	Buyer Name	Buyer Username	Passcode	Email	Buyer Mobile	Buyer A
0	2	buyer 2	buyer2	buyer2	msn@gmail.com	7878787879	kolar
1	3	buyer 3	buyer3	buyer3	buy3@hotmail.com	9090785634	kolkata
2	4	buyer 4	buyer4	buyer4	buy4@yahoo.com	9090785732	kochi
3	1	buyer 1	buyer1	buyer1	mno@gmail.com	7878787878	chenna
4	5	buyer 5	buyer5	buyer5user	xdsy567@yahoo.com	0987654321	secunc

Fig. 26.2: Updating values in tables



Current data in Farmer Table

	FID	Fname	Username	Fpassword	Femail	Fmobile	Faddress
0	1	azfar rayan	rayan	rayan_pass	xyz@gmail.com	8600611198	blr
1	2	syed	syed	syed_pass	abc@gmail.com	1234567898	bombay
2	3	rohan kumar	rohan	rohan_pass	farm@yahoo.com	7560934321	bijapur
3	4	rajat sharma	rajat	rajat_pass5	rajat1003@hotmail.com	1002003005	chennai

Select Farmer ID

4

Do you want to Delete Farmer with ID:: 4

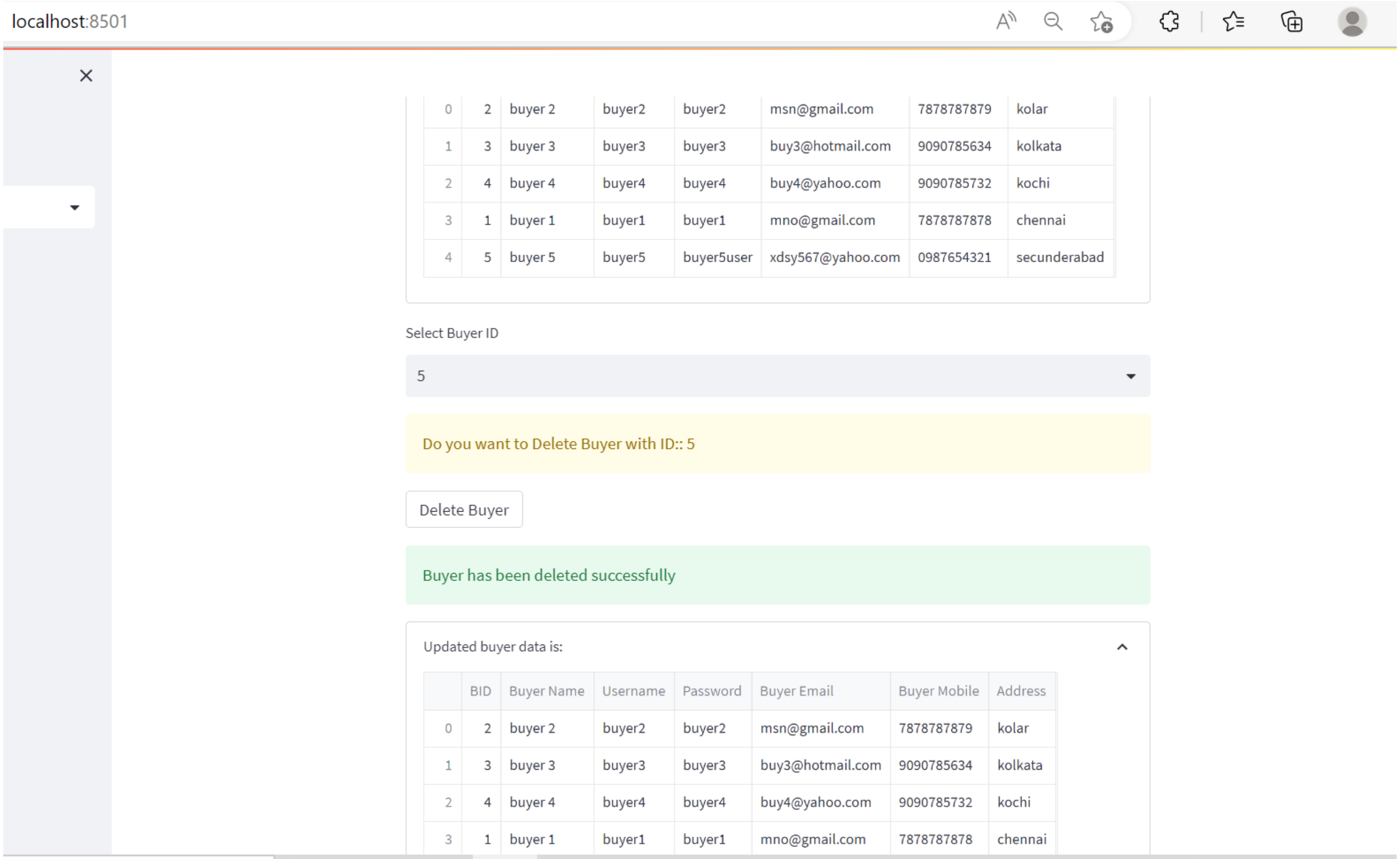
Delete Farmer

Farmer has been deleted successfully

Updated data is:

	FID	Fname	Username	Fpassword	Femail	Fmobile	Faddress
0	1	azfar rayan	rayan	rayan_pass	xyz@gmail.com	8600611198	blr
1	2	syed	syed	syed_pass	abc@gmail.com	1234567898	bombay
2	3	rohan kumar	rohan	rohan_pass	farm@yahoo.com	7560934321	bijapur

Fig. 27.1: Deleting values in tables



0	2	buyer 2	buyer2	buyer2	msn@gmail.com	7878787879	kolar
1	3	buyer 3	buyer3	buyer3	buy3@hotmail.com	9090785634	kolkata
2	4	buyer 4	buyer4	buyer4	buy4@yahoo.com	9090785732	kochi
3	1	buyer 1	buyer1	buyer1	mno@gmail.com	7878787878	chennai
4	5	buyer 5	buyer5	buyer5user	xdsy567@yahoo.com	0987654321	secunderabad

Select Buyer ID

5

Do you want to Delete Buyer with ID:: 5

Buyer has been deleted successfully

Updated buyer data is:

	BID	Buyer Name	Username	Password	Buyer Email	Buyer Mobile	Address
0	2	buyer 2	buyer2	buyer2	msn@gmail.com	7878787879	kolar
1	3	buyer 3	buyer3	buyer3	buy3@hotmail.com	9090785634	kolkata
2	4	buyer 4	buyer4	buyer4	buy4@yahoo.com	9090785732	kochi
3	1	buyer 1	buyer1	buyer1	mno@gmail.com	7878787878	chennai

Fig. 27.2: Deleting values in tables