

DBMS Mini Project

E-Commerce

Database Management System

Submitted by :

Abhishek Patil

PES1UG20CS010

V Semester

Section A

Short Description and Scope of the Project

This project simulates an e-commerce database. Customers and suppliers can register by adding their details. Products are classified into categories. Suppliers can add products to multiple categories and Customers can order multiple products from different categories. Customers can also add multiple delivery address and choose one of them for each order that they make. Customers can also see their order history with price and count of each product from that order. Analytics is done with the help of various operators and functions and procedures. An UI is also developed as a part of this project which performs basic CRUD operations as required and also has an option to run queries on the database.

RDBMS - MySQL

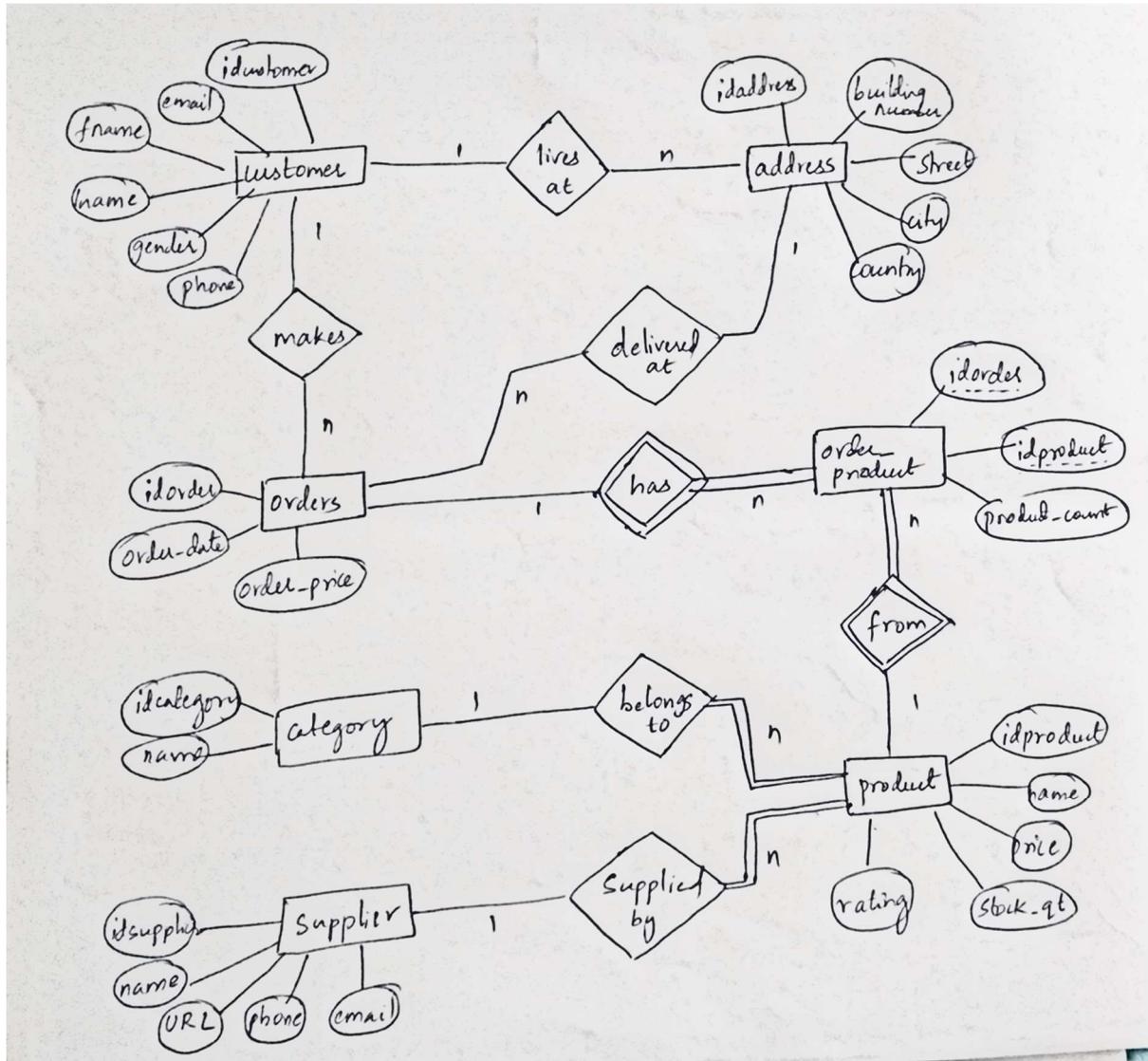
Query language - SQL

SQL editor - phpMyAdmin

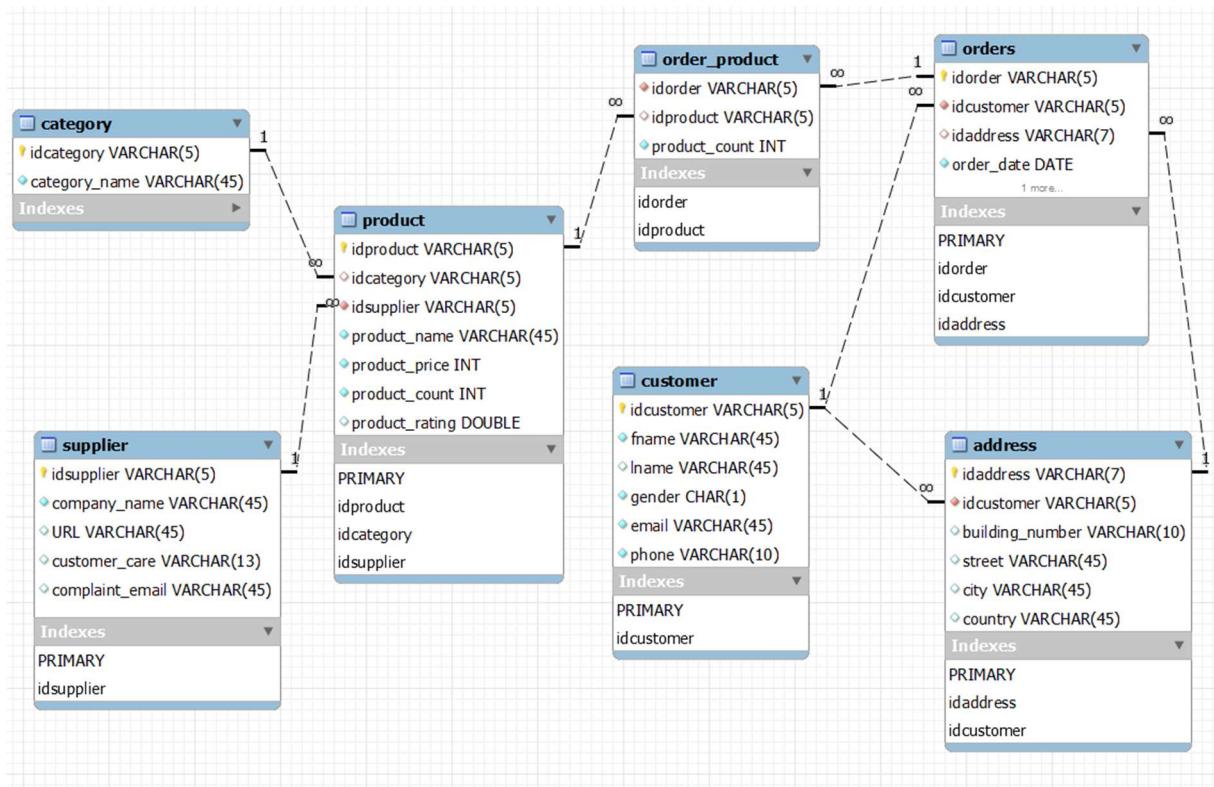
Relational Schema – MySQL Workbench

Frontend - python and Streamlit

ER diagram



Relational Schema



DDL statements - Building the database

Run SQL query/queries on database e-commerce_database_010: 

```
1 /*customer*/
2 CREATE TABLE `e-commerce_database_010`.`customer` (
3   `idcustomer` varchar(5) NOT NULL UNIQUE,
4   `fname` varchar(45) NOT NULL,
5   `lname` varchar(45) DEFAULT NULL,
6   `gender` char(1) NOT NULL,
7   `email` varchar(45) NOT NULL,
8   `phone` varchar(10) NOT NULL,
9   PRIMARY KEY (`idcustomer`)
10 );
11
12 /*supplier*/
13 CREATE TABLE `e-commerce_database_010`.`supplier` (
14   `idsupplier` VARCHAR(5) NOT NULL UNIQUE,
15   `company_name` VARCHAR(45) NOT NULL,
16   `URL` VARCHAR(45) DEFAULT NULL,
17   `customer_care` VARCHAR(13) DEFAULT NULL,
18   `complaint_email` VARCHAR(45) DEFAULT NULL,
19   PRIMARY KEY (`idsupplier`)
20 );
21
22 /*category*/
23 CREATE TABLE `e-commerce_database_010`.`category` (
24   `idcategory` VARCHAR(5) NOT NULL UNIQUE,
25   `category_name` VARCHAR(45) NOT NULL,
26   `product_count` INT NULL DEFAULT 0,
27   PRIMARY KEY (`idcategory`)
28 );
29
30 /*address*/
31 CREATE TABLE `e-commerce_database_010`.`address` (
32   `idaddress` VARCHAR(7) NOT NULL UNIQUE,
33   `idcustomer` VARCHAR(5) NOT NULL,
34   `building_number` VARCHAR(10) DEFAULT NULL,
35   `street` VARCHAR(45) DEFAULT NULL,
36   `city` VARCHAR(45) DEFAULT NULL,
37   `country` VARCHAR(45) DEFAULT "India",
38   PRIMARY KEY (`idaddress`),
39   FOREIGN KEY (`idcustomer`) REFERENCES `customer`(`idcustomer`)
40   ON DELETE CASCADE ON UPDATE CASCADE
41 );
```

```
42
43 /*product*/
44 CREATE TABLE `e-commerce_database_010`.`product` (
45   `idproduct` VARCHAR(5) NOT NULL UNIQUE,
46   `idcategory` VARCHAR(5),
47   `idsupplier` VARCHAR(5) NOT NULL,
48   `product_name` VARCHAR(45) NOT NULL,
49   `product_price` INT NOT NULL,
50   `product_count` INT NOT NULL,
51   `product_rating` DOUBLE DEFAULT 0,
52   PRIMARY KEY (`idproduct`),
53   FOREIGN KEY (`idcategory`) REFERENCES `category` (`idcategory`)
54   ON DELETE SET NULL ON UPDATE SET NULL,
55   FOREIGN KEY (`idsupplier`) REFERENCES `supplier` (`idsupplier`)
56   ON DELETE CASCADE ON UPDATE CASCADE
57 );
58
59 /*order*/
60 CREATE TABLE `e-commerce_database_010`.`order` (
61   `idorder` VARCHAR(5) NOT NULL UNIQUE,
62   `idcustomer` VARCHAR(5) NOT NULL,
63   `idaddress` VARCHAR(7),
64   `order_date` DATE NOT NULL,
65   `order_price` INT NOT NULL,
66   PRIMARY KEY (`idorder`),
67   FOREIGN KEY (`idcustomer`) REFERENCES `customer` (`idcustomer`)
68   ON DELETE CASCADE ON UPDATE CASCADE,
69   FOREIGN KEY (`idaddress`) REFERENCES `address` (`idaddress`)
70   ON DELETE SET NULL ON UPDATE SET NULL
71 );
72
73 /*order_product*/
74 CREATE TABLE `e-commerce_database_010`.`order_product` (
75   `idorder` VARCHAR(5) NOT NULL,
76   `idproduct` VARCHAR(5),
77   `product_count` INT NOT NULL,
78   FOREIGN KEY (`idorder`) REFERENCES `order` (`idorder`)
79   ON DELETE CASCADE ON UPDATE CASCADE,
80   FOREIGN KEY (`idproduct`) REFERENCES `product` (`idproduct`)
81   ON DELETE SET NULL ON UPDATE SET NULL
82 );
```

Server: localhost » Database: e-commerce_database_010

Structure SQL Search Query Export Import Operations Privileges More

Show query box

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0251 seconds.)

```
/*customer*/ CREATE TABLE `e-commerce_database_010`.`customer` ( `idcustomer` varchar(5) NOT NULL UNIQUE, `fname` varchar(45) NOT NULL, `lname` varchar(45) DEFAULT NULL, `gender` char(1) NOT NULL, `email` varchar(45) NOT NULL, `phone` varchar(10) NOT NULL, PRIMARY KEY (`idcustomer`) );
```

[Edit inline] [Edit] [Create PHP code]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0315 seconds.)

```
/*supplier*/ CREATE TABLE `e-commerce_database_010`.`supplier` ( `idsupplier` VARCHAR(5) NOT NULL UNIQUE, `company_name` VARCHAR(45) NOT NULL, `URL` VARCHAR(45) DEFAULT NULL, `customer_care` VARCHAR(13) DEFAULT NULL, `complaint_email` VARCHAR(45) DEFAULT NULL, PRIMARY KEY (`idsupplier`) );
```

[Edit inline] [Edit] [Create PHP code]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0266 seconds.)

```
/*category*/ CREATE TABLE `e-commerce_database_010`.`category` ( `idcategory` VARCHAR(5) NOT NULL UNIQUE, `category_name` VARCHAR(45) NOT NULL, `product_count` INT NULL DEFAULT 0, PRIMARY KEY (`idcategory`) );
```

Console [Edit inline] [Edit] [Create PHP code]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0307 seconds.)

```
/*address*/ CREATE TABLE `e-commerce_database_010`.`address` ( `idaddress` VARCHAR(7) NOT NULL UNIQUE, `idcustomer` VARCHAR(5) NOT NULL, `building_number` VARCHAR(10) DEFAULT NULL, `street` VARCHAR(45) DEFAULT NULL, `city` VARCHAR(45) DEFAULT NULL, `country` VARCHAR(45) DEFAULT "India", PRIMARY KEY (`idaddress`), FOREIGN KEY (`idcustomer`) REFERENCES `customer` (`idcustomer`) ON DELETE CASCADE ON UPDATE CASCADE );
```

[Edit inline] [Edit] [Create PHP code]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0344 seconds.)

```
/*product*/ CREATE TABLE `e-commerce_database_010`.`product` ( `idproduct` VARCHAR(5) NOT NULL UNIQUE, `idcategory` VARCHAR(5), `idsupplier` VARCHAR(5) NOT NULL, `product_name` VARCHAR(45) NOT NULL, `product_price` INT NOT NULL, `product_count` INT NOT NULL, `product_rating` DOUBLE DEFAULT 0, PRIMARY KEY (`idproduct`), FOREIGN KEY (`idcategory`) REFERENCES `category` (`idcategory`) ON DELETE SET NULL ON UPDATE SET NULL, FOREIGN KEY (`idsupplier`) REFERENCES `supplier` (`idsupplier`) ON DELETE CASCADE ON UPDATE CASCADE );
```

[Edit inline] [Edit] [Create PHP code]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0390 seconds.)

```
/*order*/ CREATE TABLE `e-commerce_database_010`.`order` ( `idorder` VARCHAR(5) NOT NULL UNIQUE, `idcustomer` VARCHAR(5) NOT NULL, `idaddress` VARCHAR(7), `order_date` DATE NOT NULL, `order_price` INT NOT NULL, PRIMARY KEY (`idorder`), FOREIGN KEY (`idcustomer`) REFERENCES `customer` (`idcustomer`) ON DELETE CASCADE ON UPDATE CASCADE, FOREIGN KEY (`idaddress`) REFERENCES `address` (`idaddress`) ON DELETE SET NULL ON UPDATE SET NULL );
```

[Edit inline] [Edit] [Create PHP code]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0337 seconds.)

```
/*order_product*/ CREATE TABLE `e-commerce_database_010`.`order_product` ( `idorder` VARCHAR(5) NOT NULL, `idproduct` VARCHAR(5), `product_count` INT NOT NULL, FOREIGN KEY (`idorder`) REFERENCES `order` (`idorder`) ON DELETE CASCADE ON UPDATE CASCADE, FOREIGN KEY (`idproduct`) REFERENCES `product` (`idproduct`) ON DELETE SET NULL ON UPDATE SET NULL );
```

[Edit inline] [Edit] [Create PHP code]

Populating the Database

a. Through INSERT command

```
Run SQL query/queries on database e-commerce_database_010: 
```

```
1 /*insert customers*/
2 INSERT INTO customer VALUES
3 ('C0001', 'Sherlock', 'Holmes', 'M', 'sherlock@gmail.com', '11111'),
4 ('C0002', 'Sheldon', 'Cooper', 'M', 'sheldon@gmail.com', '22222'),
5 ('C0003', 'Patrick', 'Jane', 'M', 'patrick@gmail.com', '33333'),
6 ('C0004', 'Red', 'John', 'M', 'redjohn@gmail.com', '44444'),
7 ('C0005', 'Rajesh', 'Koothrapalli', 'M', 'rajesh@gmail.com', '55555'),
8 ('C0006', 'Teresa', 'Lisbon', 'F', 'lisbon@gmail.com', '66666'),
9 ('C0007', 'Naruto', 'Uzumaki', 'M', 'narutouzumaki@gmail.com', '99999'),
10 ('C0008', 'Kakashi', 'Hatake', 'M', 'kakashi00@gmail.com', '10101');
11
12 /*insert suppliers*/
13 INSERT INTO supplier VALUES
14 ('S0001', 'Lenovo', 'www.lenovo.com', '1800-123-4545', 'lenovo.care@gmail.com'),
15 ('S0002', 'Dell', 'www.dell.com', '1800-123-4569', 'dell.care@gmail.com'),
16 ('S0003', 'Adidas', 'www.adidas.com', '1800-123-4200', 'adidas.care@gmail.com'),
17 ('S0004', 'Puma', 'www.puma.com', '1800-123-4848', 'puma.care@gmail.com'),
18 ('S0005', 'Nike', 'www.nike.com', '1800-123-4343', 'nike.care@gmail.com'),
19 ('S0006', 'Reebok', 'www.reebok.com', '1800-123-4646', 'reebok.care@gmail.com'),
20 ('S0007', 'Apple', 'www.apple.com', '1800-123-4567', 'apple.care@gmail.com');
21
22 /*insert categories*/
23 INSERT INTO category VALUES
24 ('C0001', 'Laptop'),
25 ('C0002', 'Smartphone'),
26 ('C0003', 'Shoes'),
27 ('C0004', 'Clothing');
28
29 /*insert address*/
30 INSERT INTO address VALUES
31 ('C0001-1', 'C0001', '#221B', 'Baker Street', 'London', 'UK'),
32 ('C0002-1', 'C0002', '#215S', 'Madison Ave', 'Pasadena', 'USA'),
33 ('C0003-1', 'C0003', '#1309', 'Cedars Street', 'Malibu', 'USA '),
34 ('C0003-2', 'C0003', '#3600', 'Riverside Blvd', 'Sacramento', 'USA'),
35 ('C0004-1', 'C0004', '#14', 'Street 1', 'Napa', 'USA'),
36 ('C0005-1', 'C0005', '#15', 'Madison Ave', 'Pasadena', 'USA'),
37 ('C0006-1', 'C0006', '#3600', 'Riverside Blvd', 'Sacramento', 'USA'),
38 ('C0007-1', 'C0007', '#02', 'Hokage Residence', 'Konohagakure', 'Japan'),
39 ('C0008-1', 'C0008', '#25', 'Hokage Residence', 'Konohagakure', 'Japan');
```

```

41 /*insert products*/
42 INSERT INTO product VALUES
43 ('P0001', 'C0001', 'S0001', 'Lenovo IdeaPad 330', 35000, 5, 3.5),
44 ('P0002', 'C0001', 'S0001', 'Lenovo ThinkPad E14', 75000, 10, 4.0),
45 ('P0003', 'C0001', 'S0002', 'Dell Inspiron 15', 40000, 15, 3.8),
46 ('P0004', 'C0001', 'S0002', 'Dell G15 Gaming', 90000, 10, 4.5),
47 ('P0005', 'C0001', 'S0002', 'Dell Latitude 9430', 200000, 5, 4.2),
48 ('P0006', 'C0002', 'S0007', 'iPhone 13', 70000, 25, 4.6),
49 ('P0007', 'C0002', 'S0007', 'iPhone 14', 90000, 25, 4.4),
50 ('P0008', 'C0001', 'S0007', 'MacBook Pro 2022', 150000, 10, 4.8),
51 ('P0009', 'C0003', 'S0005', 'Nike Air Force 1 Mid QS', 13000, 3, 4.0),
52 ('P0010', 'C0003', 'S0005', 'Nike Air Jordan 1 Retro ', 16000, 5, 4.0),
53 ('P0011', 'C0004', 'S0005', 'Nike Pro Dri-FIT Tshirt', 2000, 15, 3.5),
54 ('P0012', 'C0004', 'S0005', 'Nike Dri-FIT Tshirt', 2000, 15, 3.9),
55 ('P0013', 'C0003', 'S0004', 'Puma Redon Move Shoes', 6000, 5, 4.2),
56 ('P0014', 'C0003', 'S0004', 'Puma Electron E Pro Shoes', 4500, 12, 4.4),
57 ('P0015', 'C0004', 'S0003', 'Adidas Gameday Hoodie', 4800, 13, 4.5),
58 ('P0016', 'C0004', 'S0003', 'Adidas TraceRocker Jacket', 5000, 18, 4.6);
59
60 /*insert order*/
61 INSERT INTO orders VALUES
62 ('00001', 'C0002', 'C0002-1', '2022-08-11', 110000),
63 ('00002', 'C0005', 'C0005-1', '2022-07-12', 9800),
64 ('00003', 'C0002', 'C0002-1', '2020-05-14', 22000),
65 ('00004', 'C0003', 'C0003-1', '2021-06-15', 200000),
66 ('00005', 'C0003', 'C0003-2', '2022-07-16', 4000),
67 ('00006', 'C0005', 'C0005-1', '2020-08-17', 8000),
68 ('00007', 'C0004', 'C0004-1', '2020-04-09', 70000);
69
70 /*insert order details*/
71 INSERT INTO order_product VALUES
72 ('00001', 'P0001', 1),
73 ('00001', 'P0002', 1),
74 ('00002', 'P0015', 1),
75 ('00002', 'P0016', 1),
76 ('00003', 'P0010', 1),
77 ('00003', 'P0013', 1),
78 ('00004', 'P0005', 1),
79 ('00005', 'P0011', 2),
80 ('00006', 'P0012', 4),
81 ('00007', 'P0006', 1);
82

```

b. Through LOAD command

Run SQL query/queries on database e-commerce_database_010: [?](#)

```
1 LOAD DATA INFILE 'C:\\\\Users\\\\Abhishek Patil\\\\Downloads\\\\orders.csv'\n2 INTO TABLE orders\n3 FIELDS ENCLOSED BY ''"\n4 TERMINATED BY ';\n5 ESCAPED BY ''"\n6 LINES TERMINATED BY '\\n';\n7\n8 LOAD DATA INFILE 'C:\\\\Users\\\\Abhishek Patil\\\\Downloads\\\\order_product.csv'\n9 INTO TABLE order_product\n10 FIELDS ENCLOSED BY ''"\n11 TERMINATED BY ';\n12 ESCAPED BY ''"\n13 LINES TERMINATED BY '\\n';\n14
```

c. Through UI

← Server: localhost » Database: e-commerce_database_010 » Table: product

Browse Structure SQL Search Insert Export Import Privileges

Importing into the table "product"

File to import:

File may be compressed (gzip, bzip2, zip) or uncompressed.
A compressed file's name must end in **[format].[compression]**. Example: **.sql.zip**

Browse your computer: (Max: 8,192KiB)

Choose File **product.csv**

You may also drag and drop a file on any page.

Character set of the file:

utf-8

Server: localhost » Database: e-commerce_database_010

Structure SQL Search Query Export Import Operations Privileges Routines More

Show query box

8 rows inserted. (Query took 0.0032 seconds.)

```
/*insert customers*/ INSERT INTO customer VALUES ('C0001', 'Sherlock', 'Holmes', 'M', 'sherlock@gmail.com', '11111'), ('C0002', 'Sheldon', 'Cooper', 'M', 'sheldon@gmail.com', '22222'), ('C0003', 'Patrick', 'Jane', 'M', 'patrick@gmail.com', '33333'), ('C0004', 'Red', 'John', 'M', 'redjohn@gmail.com', '44444'), ('C0005', 'Rajesh', 'Koothrapalli', 'M', 'rajesh@gmail.com', '55555'), ('C0006', 'Teresa', 'Lisbon', 'F', 'lisbon@gmail.com', '66666'), ('C0007', 'Naruto', 'Uzumaki', 'M', 'narutouzumaki@gmail.com', '99999'), ('C0008', 'Kakashi', 'Hatake', 'M', 'kakashi00@gmail.com', '10101');
```

[Edit inline] [Edit] [Create PHP code]

7 rows inserted. (Query took 0.0020 seconds.)

```
/*insert suppliers*/ INSERT INTO supplier VALUES ('S0001', 'Lenovo', 'www.lenovo.com', '1800-123-4545', 'lenovo.care@gmail.com'), ('S0002', 'Dell', 'www.dell.com', '1800-123-4569', 'dell.care@gmail.com'), ('S0003', 'Adidas', 'www.adidas.com', '1800-123-4200', 'adidas.care@gmail.com'), ('S0004', 'Puma', 'www.puma.com', '1800-123-4848', 'puma.care@gmail.com'), ('S0005', 'Nike', 'www.nike.com', '1800-123-4343', 'nike.care@gmail.com'), ('S0006', 'Reebok', 'www.reebok.com', '1800-123-4646', 'reebok.care@gmail.com'), ('S0007', 'Apple', 'www.apple.com', '1800-123-4567', 'apple.care@gmail.com');
```

Console [Edit inline] [Edit] [Create PHP code]

4 rows inserted. (Query took 0.0025 seconds.)

```
/*insert categories*/ INSERT INTO category VALUES ('C0001', 'Laptop'), ('C0002', 'Smartphone'), ('C0003', 'Shoes'), ('C0004', 'Clothing');
```

[Edit inline] [Edit] [Create PHP code]

9 rows inserted. (Query took 0.0019 seconds.)

```
/*insert address*/ INSERT INTO address VALUES ('C0001-1', 'C0001', '#221B', 'Baker Street', 'London', 'UK'), ('C0002-1', 'C0002', '#2155', 'Madison Ave', 'Pasadena', 'USA'), ('C0003-1', 'C0003', '#1309', 'Cedars Street', 'Malibu', 'USA'), ('C0003-2', 'C0003', '#3600', 'Riverside Blvd', 'Sacramento', 'USA'), ('C0004-1', 'C0004', '#14', 'Street 1', 'Napa', 'USA'), ('C0005-1', 'C0005', '#15', 'Madison Ave', 'Pasadena', 'USA'), ('C0006-1', 'C0006', '#3600', 'Riverside Blvd', 'Sacramento', 'USA'), ('C0007-1', 'C0007', '#02', 'Hokage Residence', 'Konohagakure', 'Japan'), ('C0008-1', 'C0008', '#25', 'Hokage Residence', 'Konohagakure', 'Japan');
```

[Edit inline] [Edit] [Create PHP code]

16 rows inserted. (Query took 0.0032 seconds.)

```
'C0001', 'S0001', 'Lenovo ThinkPad E14', 75000, 10, 4.0), ('P0003', 'C0001', 'S0002', 'Dell Inspiron 15', 40000, 15, 3.8), ('P0004', 'C0001', 'S0002', 'Dell G15 Gaming', 90000, 10, 4.5), ('P0005', 'C0001', 'S0002', 'Dell Latitude 9430', 200000, 5, 4.2), ('P0006', 'C0002', 'S0007', 'iPhone 13', 70000, 25, 4.6), ('P0007', 'C0002', 'S0007', 'iPhone 14', 90000, 25, 4.4), ('P0008', 'C0001', 'S0007', 'MacBook Pro 2022', 150000, 10, 4.8), ('P0009', 'C0003', 'S0005', 'Nike Air Force 1 Mid QS', 13000, 3, 4.0), ('P0010', 'C0003', 'S0005', 'Nike Air Jordan 1 Retro ', 16000, 5, 4.0), ('P0011', 'C0004', 'S0005', 'Nike Pro Dri-FIT Tshirt', 2000, 15, 3.5), ('P0012', 'C0004', 'S0005', 'Nike Dri-FIT Tshirt', 2000, 15, 3.9), ('P0013', 'C0003', 'S0004', 'Puma Redon Move Shoes', 6000, 5, 4.2), ('P0014', 'C0003', 'S0004', 'Puma Electron F Pro Shoes', 4500, 12, 4.4);
```

[Edit]

7 rows inserted. (Query took 0.0031 seconds.)

```
/*insert order*/ INSERT INTO orders VALUES ('00001', 'C0002', 'C0002-1', '2022-08-11', 110000), ('00002', 'C0005', 'C0005-1', '2022-07-12', 9800), ('00003', 'C0002', 'C0002-1', '2020-05-14', 22000), ('00004', 'C0003', 'C0003-1', '2021-06-15', 20000), ('00005', 'C0003', 'C0003-2', '2022-07-16', 4000), ('00006', 'C0005', 'C0005-1', '2020-08-17', 8000), ('00007', 'C0004', 'C0004-1', '2020-04-09', 7000);
```

[Edit inline] [Edit] [Create PHP code]

10 rows inserted. (Query took 0.0030 seconds.)

```
/*insert order details*/ INSERT INTO order_product VALUES ('00001', 'P0001', 1), ('00001', 'P0002', 1), ('00002', 'P0015', 1), ('00002', 'P0016', 1), ('00003', 'P0010', 1), ('00003', 'P0013', 1), ('00004', 'P0005', 1), ('00005', 'P0011', 2), ('00006', 'P0012', 4), ('00007', 'P0006', 1);
```

[Edit inline] [Edit] [Create PHP code]

JOIN Queries

a. Get a list of all customers and their address(es)

LEFT JOIN:

Run SQL query/queries on database e-commerce_database_010: [?](#)

```
1 SELECT
2     customer.fname,
3     customer.lname,
4     address.building_number,
5     address.street,
6     address.city,
7     address.country
8 FROM
9     customer
10    LEFT JOIN
11     address ON address.idcustomer = customer.idcustomer
```

Console

Showing rows 0 - 8 (9 total, Query took 0.0006 seconds.)

```
SELECT customer.fname, customer.lname, address.building_number, address.street, address.city, address.country FROM customer LEFT JOIN address ON address.idcustomer = customer.idcustomer;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 ▾

Extra options

fname	lname	building_number	street	city	country
Sherlock	Holmes	#221B	Baker Street	London	UK
Sheldon	Cooper	#215S	Madison Ave	Pasadena	USA
Patrick	Jane	#1309	Cedars Street	Malibu	USA
Patrick	Jane	#3600	Riverside Blvd	Sacramento	USA
Red	John	#14	Street 1	Napa	USA
Rajesh	Koothrapalli	#15	Madison Ave	Pasadena	USA
Teresa	Lisbon	#3600	Riverside Blvd	Sacramento	USA
Naruto	Uzumaki	#02	Hokage Residence	Konohagakure	Japan
Kakashi	Hatake	#25	Hokage Residence	Konohagakure	Japan

b. GET a list of all laptops with their brand name, price, and rating.

RIGHT JOIN

Run SQL query/queries on database e-commerce_database_010:

```
1 SELECT
2     supplier.company_name Company,
3     laptops.product_name AS model,
4     laptops.product_price AS price,
5     laptops.product_rating AS rating
6 FROM
7     supplier
8     RIGHT JOIN (
9         SELECT
10            category.category_name,
11            product.idsupplier,
12            product.product_name,
13            product.product_price,
14            product.product_rating
15        FROM
16            product
17            RIGHT JOIN category ON category.idcategory = product.idcategory
18        WHERE
19            category.category_name = 'Laptop'
20    ) AS laptops ON laptops.idsupplier = supplier.idsupplier
```

Console

Showing rows 0 - 5 (6 total, Query took 0.0009 seconds.)

```
SELECT supplier.company_name Company, laptops.product_name AS model, laptops.product_price AS price, laptops.product_rating AS rating FROM supplier RIGHT JOIN ( SELECT category.category_name, product.idsupplier, product.product_name, product.product_price, product.product_rating FROM product RIGHT JOIN category ON category.idcategory = product.idcategory WHERE category.category_name = 'Laptop' ) AS laptops ON laptops.idsupplier = supplier.idsupplier;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Company	model	price	rating
Lenovo	Lenovo IdeaPad 330	35000	3.5
Lenovo	Lenovo ThinkPad E14	75000	4
Dell	Dell Inspiron 15	40000	3.8
Dell	Dell G15 Gaming	90000	4.5
Dell	Dell Latitude 9430	200000	4.2
Apple	MacBook Pro 2022	150000	4.8

c. GET a list of all products ordered by Patrick Jane

INNER JOIN

Run SQL query/queries on database e-commerce_database_010:

```
1 SELECT
2     prod.order_date AS `Order date`,
3     prod.product_count AS Quantity,
4     product.product_name AS Product
5 FROM
6     product
7     INNER JOIN (
8         SELECT
9             order_product.idproduct,
10            order_product.product_count,
11            ord.order_date
12        FROM
13            order_product
14            INNER JOIN (
15                SELECT
16                    orders.idorder,
17                    orders.order_date
18                FROM
19                    orders
20                    INNER JOIN customer ON customer.idcustomer = orders.idcustomer
21                WHERE
22                    customer.fname = 'Patrick'
23                    AND customer.lname = 'Jane'
24            ) AS ord ON ord.idorder = order_product.idorder
25        ) AS prod ON prod.idproduct = product.idproduct
```

Console

Showing rows 0 - 1 (2 total, Query took 0.0008 seconds.)

```
SELECT prod.order_date AS `Order date`, prod.product_count AS Quantity, product.product_name AS Product
FROM product
INNER JOIN (
    SELECT order_product.idproduct, order_product.product_count,
    ord.order_date
    FROM order_product
    INNER JOIN (
        SELECT orders.idorder, orders.order_date
        FROM orders
        INNER JOIN customer ON customer.idcustomer = orders.idcustomer
        WHERE customer.fname = 'Patrick' AND
        customer.lname = 'Jane'
    ) AS ord ON ord.idorder = order_product.idorder
) AS prod ON prod.idproduct = product.idproduct;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 ▾ Filter rows: Search this table

Extra options

Order date	Quantity	Product
2021-06-15	1	Dell Latitude 9430
2022-07-16	2	Nike Pro Dri-FIT Tshirt

Console

d. Get a list of all products with rating ≥ 4.5 with their supplier

JOIN

Run SQL query/queries on database **e-commerce_database_010**: 

```
1 SELECT
2     product.product_name as Name,
3     product.product_price as Price,
4     product.product_rating as Rating,
5     supplier.company_name as Supplier
6 from
7     product
8     JOIN supplier ON supplier.idsupplier = product.idsupplier
9 WHERE
10    product.product_rating >= 4.5
```

 Showing rows 0 - 4 (5 total, Query took 0.0008 seconds.)

```
SELECT product.product_name as Name, product.product_price as Price,
product.product_rating as Rating, supplier.company_name as Supplier from product
JOIN supplier ON supplier.idsupplier = product.idsupplier WHERE
product.product_rating >= 4.5;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all

Number of rows:

All 

Filter rows:

Search this table

Sort by key:

Extra options

Name	Price	Rating	Supplier
Dell G15 Gaming	90000	4.5	Dell
iPhone 13	70000	4.6	Apple
MacBook Pro 2022	150000	4.8	Apple
Adidas Gameday Hoodie	4800	4.5	Adidas
Adidas TraceRocker Jacket	5000	4.6	Adidas

Aggregate Functions

a. Get total money spent by customers on the platform

SUM

Run SQL query/queries on database e-commerce_database_010: [?](#)

```
1 SELECT
2     customer.fname,
3     customer.lname,
4     SUM(orders.order_price) as `Total Spent`
5 FROM
6     customer
7     INNER JOIN orders ON customer.idcustomer = orders.idcustomer
8 GROUP BY
9     orders.idcustomer
```

Showing rows 0 - 3 (4 total, Query took 0.0009 seconds.)

```
SELECT customer.fname, customer.lname, SUM(orders.order_price) as `Total Spent`
FROM customer INNER JOIN orders ON customer.idcustomer = orders.idcustomer GROUP BY
orders.idcustomer;
```

Profiling [[Edit inline](#)] [[Edit](#)] [[Explain SQL](#)] [[Create PHP code](#)] [[Refresh](#)]

Show all | Number of rows: All ▾ Filter rows:

Extra options

fname	lname	Total Spent
Sheldon	Cooper	132000
Rajesh	Koothrapalli	17800
Patrick	Jane	204000
Red	John	70000

b. GET Average price of all products in each category

AVG

Run SQL query/queries on database **e-commerce_database_010**: 

```
1 SELECT
2     category.category_name,
3     AVG(product.product_price) as `Average Price`
4 FROM
5     product
6     JOIN category ON category.idcategory = product.idcategory
7 GROUP BY
8     product.idcategory
```

 Showing rows 0 - 3 (4 total, Query took 0.0045 seconds.)

```
SELECT category.category_name, AVG(product.product_price) as `Average Price` FROM
product JOIN category ON category.idcategory = product.idcategory GROUP BY
product.idcategory;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: All ▾ Filter rows: Search this table

Extra options

category_name	Average Price
Laptop	98333.3333
Smartphone	80000.0000
Shoes	9875.0000
Clothing	3450.0000

c. GET number of products supplied by each supplier

COUNT

Run SQL query/queries on database e-commerce_database_010: 

```
1 SELECT
2     supplier.company_name as Supplier,
3     COUNT(product.idproduct) as `Number of products supplied`
4 from
5     product
6     JOIN supplier ON supplier.idsupplier = product.idsupplier
7 GROUP BY
8     supplier.idsupplier
9 ORDER BY
10    supplier.company_name|
```

 Showing rows 0 - 5 (6 total, Query took 0.0013 seconds.)

```
SELECT supplier.company_name as Supplier, COUNT(product.idproduct) as `Number of products supplied` from product JOIN supplier ON supplier.idsupplier = product.idsupplier GROUP BY supplier.idsupplier ORDER BY supplier.company_name;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all

| Number of rows:

All 

Filter rows:

Search this table

Extra options

Supplier	Number of products supplied
Adidas	2
Apple	3
Dell	3
Lenovo	2
Nike	4
Puma	2

d. GET the highest priced product in each category

MAX

Run SQL query/queries on database e-commerce_database_010: 

```
1 SELECT
2     product.product_name AS `Product Name`,
3     maxi.Category,
4     maxi.`Max Price`
5 FROM
6     product
7     INNER JOIN (
8         SELECT
9             category.category_name AS Category,
10            category.idcategory,
11            MAX(product.product_price) as `Max Price`
12        from
13            product
14            INNER JOIN category ON category.idcategory = product.idcategory
15        GROUP BY
16            product.idcategory|
17        ORDER BY
18            category.category_name
19    ) AS maxi ON maxi.`Max Price` = product.product_price
20    AND maxi.idcategory = product.idcategory
```

 Showing rows 0 - 3 (4 total, Query took 0.0009 seconds.)

```
SELECT product.product_name AS `Product Name`, maxi.Category, maxi.`Max Price` FROM product
INNER JOIN ( SELECT category.category_name AS Category, category.idcategory,
MAX(product.product_price) as `Max Price` from product INNER JOIN category ON
category.idcategory = product.idcategory GROUP BY product.idcategory ORDER BY
category.category_name ) AS maxi ON maxi.`Max Price` = product.product_price AND
maxi.idcategory = product.idcategory;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Product Name	Category	Max Price
Dell Latitude 9430	Laptop	200000
iPhone 14	Smartphone	90000
Nike Air Jordan 1 Retro	Shoes	16000
Adidas TraceRocker Jacket	Clothing	5000

SET OPERATIONS

a. GET the list of users who haven't ordered products - EXCEPT

Run SQL query/queries on database e-commerce_database_010:

```
1 SELECT customer.idcustomer, customer.fname, customer.lname FROM customer NATURAL JOIN
2 ((SELECT customer.idcustomer FROM customer)
3 EXCEPT
4 (SELECT orders.idcustomer FROM orders)) AS not_ordered;
```

Showing rows 0 - 3 (4 total, Query took 0.0027 seconds.)

```
SELECT customer.idcustomer, customer.fname, customer.lname FROM customer NATURAL JOIN ((SELECT customer.idcustomer FROM customer) EXCEPT (SELECT
orders.idcustomer FROM orders)) AS not_ordered;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

idcustomer	fname	lname
C0001	Sherlock	Holmes
C0006	Teresa	Lisbon
C0007	Naruto	Uzumaki
C0008	Kakashi	Hatake

b. Get the list of suppliers who supply both laptops and smartphones - INTERSECT

Run SQL query/queries on database e-commerce_database_010:

```
1 SELECT supplier.idsupplier, supplier.company_name, supplier.URL FROM supplier NATURAL JOIN
2 (
3     (SELECT product.idsupplier FROM product WHERE product.idcategory IN
4         (SELECT category.idcategory FROM category WHERE category.category_name = "Laptop"))
5 INTERSECT
6     (SELECT product.idsupplier FROM product WHERE product.idcategory IN
7         (SELECT category.idcategory FROM category WHERE category.category_name = "Smartphone"))
8 ) AS electronics;
```

Showing rows 0 - 0 (1 total, Query took 0.0015 seconds.)

```
SELECT supplier.idsupplier, supplier.company_name, supplier.URL FROM supplier NATURAL JOIN ( (SELECT product.idsupplier FROM product WHERE
product.idcategory IN (SELECT category.idcategory FROM category WHERE category.category_name = "Laptop")) INTERSECT (SELECT product.idsupplier
FROM product WHERE product.idcategory IN (SELECT category.idcategory FROM category WHERE category.category_name = "Smartphone")) ) AS
electronics;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

idsupplier	company_name	URL
S0007	Apple	www.apple.com

c. GET the list of products that either weren't bought by customers or have a rating less than 4 - UNION

Run SQL query/queries on database e-commerce_database_010: 

```
1 SELECT product.idproduct, product.product_name FROM product WHERE product.idproduct NOT IN
2     (SELECT order_product.idproduct FROM order_product)
3 UNION
4 SELECT product.idproduct, product.product_name FROM product WHERE product.product_rating < 4;
```

Showing rows 0 - 8 (9 total, Query took 0.0052 seconds.)

```
SELECT product.idproduct, product.product_name FROM product WHERE product.idproduct NOT IN (SELECT order_product.idproduct FROM order_product)
UNION SELECT product.idproduct, product.product_name FROM product WHERE product.product_rating < 4;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

idproduct	product_name
P0003	Dell Inspiron 15
P0004	Dell G15 Gaming
P0007	iPhone 14
P0008	MacBook Pro 2022
P0009	Nike Air Force 1 Mid QS
P0014	Puma Electron E Pro Shoes
P0001	Lenovo IdeaPad 330
P0011	Nike Pro Dri-FIT Tshirt
P0012	Nike Dri-FIT Tshirt

d. Get the list of suppliers who are not supplying any products currently - EXCEPT

Run SQL query/queries on database e-commerce_database_010: 

```
1 SELECT supplier.idsupplier, supplier.company_name, supplier.URL FROM supplier NATURAL JOIN
2 ((SELECT supplier.idsupplier FROM supplier)
3 EXCEPT
4 (SELECT product.idsupplier FROM product)) AS supplier;
```

Showing rows 0 - 0 (1 total, Query took 0.0050 seconds.)

```
SELECT supplier.idsupplier, supplier.company_name, supplier.URL FROM supplier NATURAL JOIN ((SELECT supplier.idsupplier
FROM supplier) EXCEPT (SELECT product.idsupplier FROM product)) AS supplier;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

idsupplier	company_name	URL
S0006	Reebok	www.reebok.com

Functions and Procedures

Function

Objective: Supplier can input product ID and know the total sales of the product. This gives the supplier a good analysis of his product's performance on the e-commerce site.

Run SQL query/queries on database e-commerce_database_010: 

```
1 DELIMITER $$  
2 CREATE FUNCTION total_sales (ProductID VARCHAR(5))  
3 RETURNS INT  
4 READS SQL DATA  
5 DETERMINISTIC  
6 BEGIN  
7     DECLARE sale_amount INT;  
8     DECLARE total_sold INT;  
9     DECLARE price INT;  
10    SET total_sold = (SELECT SUM(product_count) FROM order_product  
11                      WHERE idproduct = ProductID);  
12    SET price = (SELECT product_price FROM product  
13                      WHERE idproduct = ProductID);  
14    SET sale_amount = total_sold * price;  
15    RETURN sale_amount;  
16 END; $$
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0127 seconds.)

```
CREATE FUNCTION total_sales (ProductID VARCHAR(5)) RETURNS INT READS SQL DATA DETERMINISTIC BEGIN DECLARE sale_amount INT;  
DECLARE total_sold INT; DECLARE price INT; SET total_sold = (SELECT SUM(product_count) FROM order_product WHERE idproduct =  
ProductID); SET price = (SELECT product_price FROM product WHERE idproduct = ProductID); SET sale_amount = total_sold *  
price; RETURN sale_amount; END;
```

[Edit inline] [Edit] [Create PHP code]

Sales of Nike – S0005 (*NULL indicates quantity_sold = 0*)

```
1 SELECT product_name, total_sales(idproduct) as `Total Sales`  
2 FROM product WHERE product.idsupplier = 'S0005';
```

product_name	Total Sales
Nike Air Force 1 Mid QS	NULL
Nike Air Jordan 1 Retro	16000
Nike Pro Dri-FIT Tshirt	4000
Nike Dri-FIT Tshirt	8000

Procedure

Objective: Customer can input Category name and get products with >4.5 rating from the category along with their prices. Helps the customer in keeping tabs on good products on the ecommerce site.

```
1 DELIMITER $$  
2 CREATE PROCEDURE display_good_products(  
3     IN categoryname VARCHAR(45),  
4     IN rating DOUBLE  
5 )  
6 BEGIN  
7     DECLARE categoryID VARCHAR(5);  
8     SET categoryID = (SELECT idcategory FROM category  
9                         WHERE category_name = categoryName);  
10    SELECT product_name, product_price, product_rating FROM product  
11      WHERE idcategory = categoryID AND product_rating >= rating;  
12 END; $$
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0052 seconds.)

```
CREATE PROCEDURE display_good_products( IN categoryname VARCHAR(45), IN rating DOUBLE ) BEGIN DECLARE categoryID VARCHAR(5); SET categoryID = (SELECT idcategory FROM category WHERE category_name = categoryName); SELECT product_name, product_price, product_rating FROM product WHERE idcategory = categoryID AND product_rating >= rating; END;
```

[Edit inline] [Edit] [Create PHP code]

Customer wants to see laptops with rating>=4.0

```
1 CALL display_good_products("Laptop", 4.0);
```

Showing rows 0 - 3 (4 total, Query took 0.0027 seconds.)

```
CALL display_good_products("Laptop", 4.0);
```

[Edit inline] [Edit] [Create PHP code]

Show all | Number of rows: 25 ▾ Filter rows: Search this table

Extra options

product_name	product_price	product_rating
Lenovo ThinkPad E14	75000	4
Dell G15 Gaming	90000	4.5
Dell Latitude 9430	200000	4.2
MacBook Pro 2022	150000	4.8

TRIGGER

Objective: A new product which hasn't had any sale yet cannot have >0 rating. Hence, this trigger shows an error when supplier tries to add product with >0 rating.

```
1 DELIMITER $$  
2 CREATE TRIGGER update_product_rating  
3 BEFORE INSERT  
4 ON `product` FOR EACH ROW  
5 BEGIN  
6     DECLARE error_msg VARCHAR(100);  
7     SET error_msg = ('New product must have 0 rating');  
8     IF NEW.product_rating > 0 THEN  
9         SIGNAL SQLSTATE '45000'  
10        SET MESSAGE_TEXT = error_msg;  
11    END IF;  
12 END $$
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0137 seconds.)

```
CREATE TRIGGER update_product_rating BEFORE INSERT ON `product` FOR EACH ROW BEGIN DECLARE error_msg VARCHAR(100); SET error_msg = ('New product must have 0 rating'); IF NEW.product_rating > 0 THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = error_msg; END IF; END;
```

[Edit inline] [Edit] [Create PHP code]

Reebok Tries to add shoes with rating 4.0

```
1 INSERT INTO product VALUES('P0017', 'C0003', 'S0006', 'Reebok Shoes', 4000, 10, 4.0);
```

Error

SQL query: [Copy](#)

```
INSERT INTO product VALUES('P0017', 'C0003', 'S0006', 'Reebok Shoes', 4000, 10, 4.0);
```

MySQL said: 

```
#1644 - New product must have 0 rating
```

FRONTEND

CRUD operations :

Create/Add data

The screenshot shows a web application interface for performing CRUD operations on a database. On the left, a sidebar menu displays 'CRUD' under 'Menu', 'Customer' under 'Table', and 'Create' under 'Operation'. The main content area is titled 'E-commerce Database Management System - PES1UG20CS010' and prompts 'Choose Table to perform CRUD operations'. Below this, a specific 'Customer' form is shown with fields for Customer ID (C0020), Gender (M), First Name (Abhishek), Email ID (abhijpatil2003@gmail.com), Last Name (Patil), and Phone number (9632714472). A button labeled 'Add Customer' is present. At the bottom, a green success message box states 'Successfully added Customer: Abhishek Patil'.

Read data

The screenshot shows the same web application interface after a record has been added. The sidebar menu now shows 'Read' under 'Operation'. The main content area displays a table titled 'Customer' with columns: Customer ID, First Name, Last Name, Gender, Email ID, and Phone Number. The table lists nine records, with the last record (Customer ID C0020, First Name Abhishek, Last Name Patil, Gender M, Email abhijpatil2003@gmail.com, Phone number 9632714472) highlighted by a red border.

	Customer ID	First Name	Last Name	Gender	Email ID	Phone Number
0	C0001	Sherlock	Holmes	M	sherlock@gmail.com	11111
1	C0002	Sheldon	Cooper	M	sheldon@gmail.com	22222
2	C0003	Patrick	Jane	M	patrick@gmail.com	33333
3	C0004	Red	John	M	redjohn@gmail.com	44444
4	C0005	Rajesh	Koothrapalli	M	rajesh@gmail.com	55555
5	C0006	Teresa	Lisbon	F	lisbon@gmail.com	66666
6	C0007	Naruto	Uzumaki	M	narutouzumaki@gmail.com	99999
7	C0008	Kakashi	Hatake	M	kakashi00@gmail.com	10101
8	C0020	Abhishek	Patil	M	abhijpatil2003@gmail.com	9632714472

The highlighted record is the one which was added through the frontend.

Update data

Before updating email of Sherlock Holmes

The screenshot shows the 'Customer' table with the following data:

	Customer ID	First Name	Last Name	Gender	Email ID	Phone Number
0	C0001	Sherlock	Holmes	M	sherlock@gmail.com	11111
1	C0002	Sheldon	Cooper	M	sheldon@gmail.com	22222
2	C0003	Patrick	Jane	M	patrick@gmail.com	33333
3	C0004	Red	John	M	redjohn@gmail.com	44444
4	C0005	Rajesh	Koothrapalli	M	rajesh@gmail.com	55555

The update form for Customer C0001 shows the following fields:

- First Name: Sherlock
- Last Name: Holmes
- Gender: M
- Email: holmes@gmail.com
- Phone Number: 11111

A success message at the bottom states: "Successfully updated customer details of Customer C0001".

After updating email of Sherlock Holmes

The screenshot shows the 'Customer' table with the following data, reflecting the update:

	Customer ID	First Name	Last Name	Gender	Email ID	Phone Number
0	C0001	Sherlock	Holmes	M	holmes@gmail.com	11111
1	C0002	Sheldon	Cooper	M	sheldon@gmail.com	22222
2	C0003	Patrick	Jane	M	patrick@gmail.com	33333
3	C0004	Red	John	M	redjohn@gmail.com	44444
4	C0005	Rajesh	Koothrapalli	M	rajesh@gmail.com	55555

Delete data

Before deleting second address of customer C0003

Address

Current data

	Address ID	Customer ID	Building Number	Street	City	Country
0	C0001-1	C0001	#221B	Baker Street	London	UK
1	C0002-1	C0002	#215S	Madison Ave	Pasadena	USA
2	C0003-1	C0003	#1309	Cedars Street	Malibu	USA
3	C0003-2	C0003	#3600	Riverside Blvd	Sacramento	USA
4	C0004-1	C0004	#14	Street 1	Napa	USA
5	C0005-1	C0005	#15	Madison Ave	Pasadena	USA
6	C0006-1	C0006	#3600	Riverside Blvd	Sacramento	USA
7	C0007-1	C0007	#02	Hokage Residence	Konohagakure	Japan
8	C0008-1	C0008	#25	Hokage Residence	Konohagakure	Japan

Address to Delete

After deleting second address - C0003-2, only address C0003-1 remains for customer C0003

Address to Delete

[C0003-2, 'C0003']

Do you want to delete Address['C0003-2', 'C0003']? This action cannot be reversed!

DELETE

Address C0003-2 of customer C0003 has been deleted successfully, see Updated data below.

Current data

	Address ID	Customer ID	Building Number	Street	City	Country
0	C0001-1	C0001	#221B	Baker Street	London	UK
1	C0002-1	C0002	#215S	Madison Ave	Pasadena	USA
2	C0003-1	C0003	#1309	Cedars Street	Malibu	USA
3	C0004-1	C0004	#14	Street 1	Napa	USA
4	C0005-1	C0005	#15	Madison Ave	Pasadena	USA

Query Box

Calling function total_sales

The screenshot shows the 'Query Box' application interface. On the left is a dark sidebar with a 'Menu' section containing a dropdown menu set to 'Query Box'. The main area has a title 'Query Box' and a sub-section 'Enter Query' containing a code editor with the following SQL query:

```
SELECT
    product_name,
    total_sales(idproduct) as `Total Sales`
FROM
    product
WHERE
    product.idsupplier = 'S0005';
```

Below the code editor is a button labeled 'Execute'. A green success message box at the bottom states 'Query Executed Successfully'. Below this message is a table displaying the results of the query:

	0	1
0	Nike Air Force 1 Mid QS	<NA>
1	Nike Air Jordan 1 Retro	16,000.0000
2	Nike Pro Dri-FIT Tshirt	4,000.0000
3	Nike Dri-FIT Tshirt	8,000.0000