## Q1/-

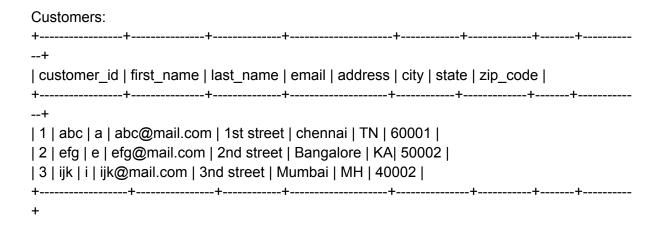
Cocoa Confections is a small bakery that sells brownies, cookies, pies, and other delicious treats to customers online. It keeps records of all of its online sales in an SQL database that is automatically populated as customers places orders on its site. In its database, Cocoa Confections has a customers table to keep track of customer contact information, and an orders table to keep track of various orders that those customers have placed. The schema of these tables is as follows:

```
CREATE TABLE customers (
customer id INT PRIMARY KEY,
first name VARCHAR(255) NOT NULL,
last name VARCHAR(255) NOT NULL,
email VARCHAR(255) NOT NULL,
address VARCHAR(255) DEFAULT NULL,
city VARCHAR(255) DEFAULT NULL,
state VARCHAR(2) DEFAULT NULL,
zip_code VARCHAR(5) DEFAULT NULL
);
CREATE TABLE orders (
order id INT PRIMARY KEY,
customer id INT NOT NULL,
order_placed_date DATE NOT NULL,
FOREIGN KEY (customer_id) REFERENCES customers
(customer_id)
);
```

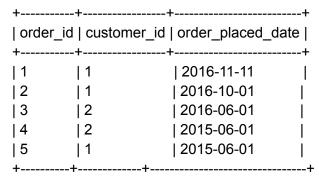
It's the end of 2016, and the owner of Cocoa Confections wants to write an SQL query that finds the COUNT of orders placed in 2016 by customer e-mail address. She wants to ORDER the results by the COUNT of orders placed in 2016, descending, so that she can personally send thank-you e-mails to Cocoa Confection's top customers by order volume.

Can you write a query that will help the owner of Cocoa Confections find the COUNT of all orders placed in 2016, by customer e-mail address, sorted Descending?

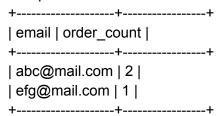
Sample Input:



## Orders:



## Sample OutPut:



## Q2/-Use the below queries to create a database and respective table.

CREATE DATABASE ORG; SHOW DATABASES; USE ORG;

```
CREATE TABLE Worker (
WORKER ID INT PRIMARY KEY
FIRST_NAME VARCHAR(25),
LAST NAME VARCHAR(25),
SALARY INT,
JOINING_DATE DATETIME,
DEPARTMENT VARCHAR(25)
);
INSERT INTO Worker
(WORKER ID, FIRST NAME, LAST NAME, SALARY, JOINING DATE, DEPARTMENT)
VALUES
(001, 'Monika', 'Arora', 100000, '14-02-20 09.00.00', 'HR'),
(002, 'Niharika', 'Verma', 80000, '14-06-11 09.00.00', 'Admin'),
(003, 'Vishal', 'Singhal', 300000, '14-02-20 09.00.00', 'HR'),
(004, 'Amitabh', 'Singh', 500000, '14-02-20 09.00.00', 'Admin'),
(005, 'Vivek', 'Bhati', 500000, '14-06-11 09.00.00', 'Admin'),
(006, 'Vipul', 'Diwan', 200000, '14-06-11 09.00.00', 'Account'),
(007, 'Satish', 'Kumar', 75000, '14-01-20 09.00.00', 'Account'),
(008, 'Geetika', 'Chauhan', 90000, '14-04-11 09.00.00', 'Admin');
CREATE TABLE Bonus (
      WORKER_REF_ID INT,
      BONUS_AMOUNT INT,
      BONUS DATE DATETIME,
      FOREIGN KEY (WORKER_REF_ID)
      REFERENCES Worker(WORKER_ID)
      ON DELETE CASCADE
);
INSERT INTO Bonus
(WORKER_REF_ID, BONUS_AMOUNT, BONUS_DATE) VALUES
(001, 5000, '16-02-20'),
(002, 3000, '16-06-11'),
(003, 4000, '16-02-20'),
(001, 4500, '16-02-20'),
(002, 3500, '16-06-11');
```

```
CREATE TABLE Title (
      WORKER_REF_ID INT,
      WORKER TITLE VARCHAR(25),
      AFFECTED_FROM DATETIME,
      FOREIGN KEY (WORKER REF ID)
      REFERENCES Worker(WORKER ID)
      ON DELETE CASCADE
);
INSERT INTO Title
(WORKER_REF_ID, WORKER_TITLE, AFFECTED_FROM) VALUES
(001, 'Manager', '2016-02-20 00:00:00'),
(002, 'Executive', '2016-06-11 00:00:00'),
(008, 'Executive', '2016-06-11 00:00:00'),
(005, 'Manager', '2016-06-11 00:00:00'),
(004, 'Asst. Manager', '2016-06-11 00:00:00'),
(007, 'Executive', '2016-06-11 00:00:00'),
(006, 'Lead', '2016-06-11 00:00:00'),
(003, 'Lead', '2016-06-11 00:00:00');
```

- 1. Write an SQL query to show the second highest salary from a Worker table.
- 2. Write an SQL query to determine the 5 highest salary from a Worker table.
- 3. Write an SQL query to show only even rows from a Worker table.
- 4. Write an SQL query to fetch the no. of workers for each department in descending order from the Worker table.
- 5. Write an SQL query to fetch the list of employees with the same salary from the Worker table.