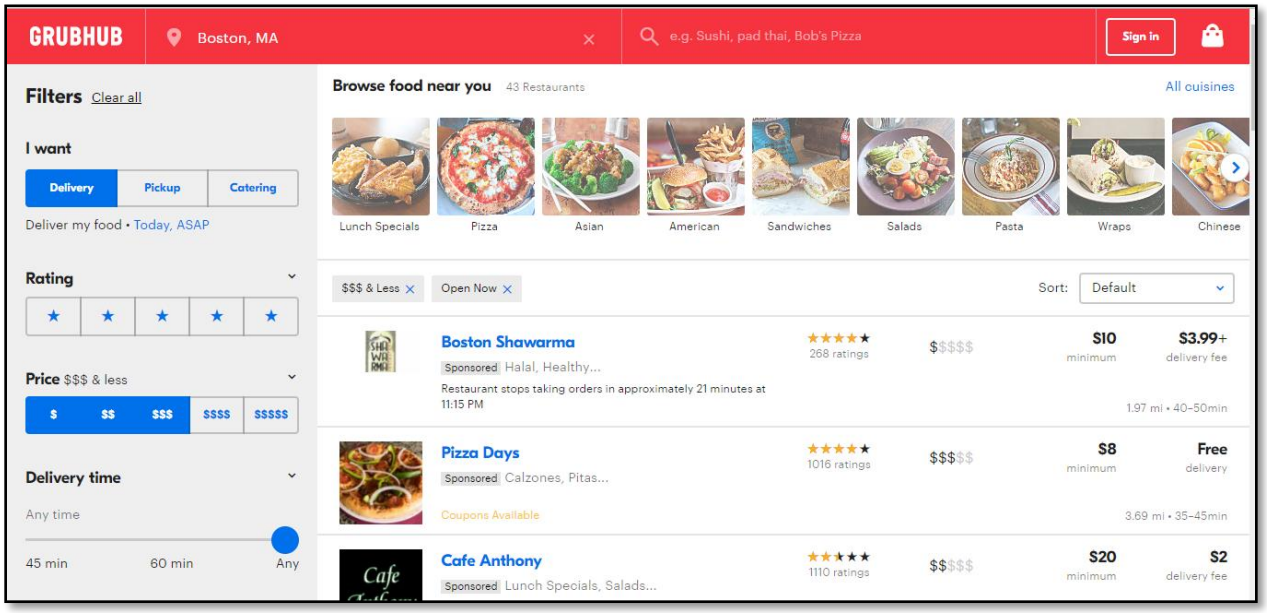
Krutika Kothawale

Final Project Report

This Project is a database for a **food ordering website** which has list of restaurants from which food can be ordered by customers.

The food ordering website that I have considered here is **GRUBHUB**

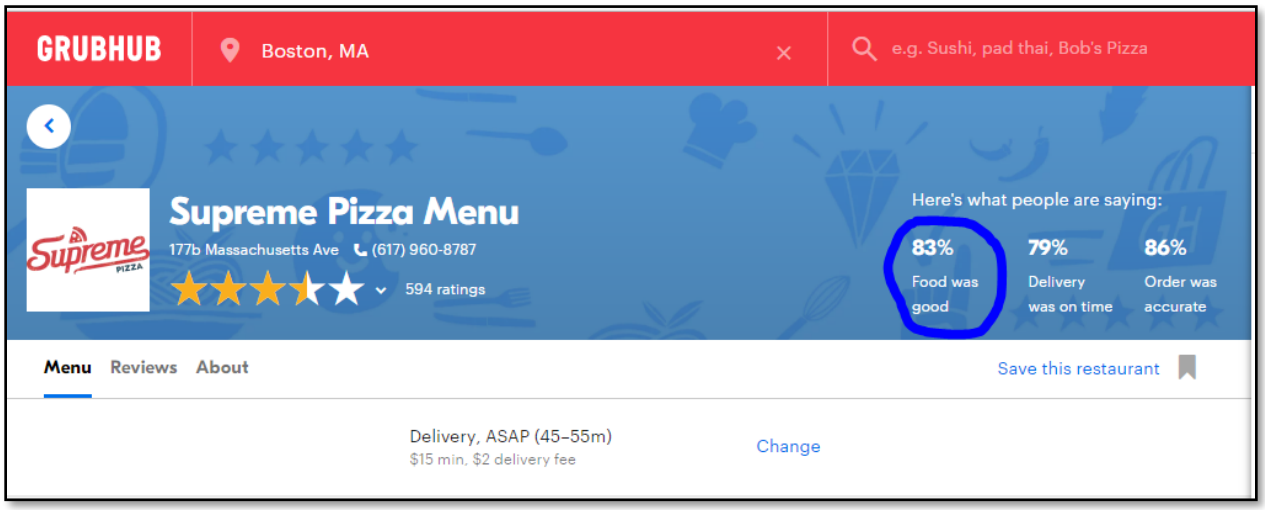


The database used for this project is MySQL
Database Name: mydb
ER Diagram is in separate PDF in the same zip file uploaded (Final ER Diagram of Project)
The project has:

a. Views
b. Stored Procedures
c. Triggers
d. Privileges

As an extra feature I have added natural language full text search in a stored procedure

This will help to calculate the percentage of customers who have said “Food is good” in the review of a particular restaurent like in the picture bellow



View

1. **List the details of all restaurants Like**
   - Restaurent Name
   - Timings
   - Minimum order price
   - Address
   - Discount
   - Delivery Or Pickup Available
   - Average of Rating given to the restaurant

```
CREATE VIEW V_Restaurant_List AS
Select   distinct Restaurant_Name,
         Opening_Time As Opens_At,
         Closing_Time As Closes_At,
         Min_Order_Price As Min_Order_Should_Be,
         Concat(Street, ' ', City, ' ',State, ' ',ZIP, ' ', Country) As Address,
         (Select (CASE WHEN(MAX(Discount_Percentage) > 0)
                    Then MAX(Discount_Percentage)
                    Else 'Discount Not Available'
                 END)
          from Rest_Discounts where Restaurant.Restaurant_ID = Rest_Discounts.Restaurant_ID) AS
          Discount,
          (Select CASE WHEN (Delivery_Facility = 'Y')
                      Then ' Delivery Available'
                   When (Pickup_Facility = 'Y')
                      Then ' Pickup Available'
                   When (Catering_Facility = 'Y')
                      Then ' Catering Available'
                 End) As `Delivery Or Pickup`,

           (Select Round(Avg(Rating),0)
            From Rating_Restaurent
            Where Rating_Restaurent.Restaurant_ID = Restaurant.Restaurant_ID) As
            Average_Rating_of_Restaurant
From Restaurant
INNER JOIN Address ON Restaurant.Address_ID = Address.Address_ID
LEFT OUTER JOIN Rest_Discounts ON Restaurant.Restaurant_ID = Rest_Discounts.Restaurant_ID;


select * from V_Restaurant_List;
```

Restaurant_Name	Opens_At	Closes_At	Min_Order_Should_Be	Address	Discount	Delivery Or Pickup	Average_Rating_of_Restaurant
Supreme Pizza	10:00:00	02:00:00	75.00	St. Germain Boston Massachusetts 02115 US	30	Delivery Available	3
Regina Pizzeria	10:00:00	02:00:00	0.00	Beacon Street Boston Massachusetts 02134 US	50	Delivery Available	2
New York Pizza	11:00:00	12:00:00	0.00	Boston Common Boston Massachusetts 02211 US	30	Delivery Available	3
Wok n Talk	12:00:00	23:00:00	0.00	10 Bosworth St Boston Massachusetts 02108 US	Discount Not Available	Pickup Available	4
SUBWAY	11:00:00	23:00:00	50.00	290 Washington St. Boston Massachusetts 021...	Discount Not Available	Delivery Available	3
Boston Shawarma	11:00:00	23:00:00	50.00	48 Temple Pl Boston Massachusetts 02111 US	Discount Not Available	Delivery Available	3
Cheesecake Factory	11:00:00	23:00:00	99.99	558 Washington St Boston Massachusetts 0211...	Discount Not Available	Delivery Available	2
IHOP	11:00:00	23:00:00	50.00	39 Dalton St Boston Massachusetts 02115 US	Discount Not Available	Delivery Available	NULL
BiMediterranean	11:00:00	23:00:00	50.00	800 Boylston St Boston Massachusetts 02118 US	Discount Not Available	Delivery Available	NULL

2. Create A View for customer to view his order history

- Here we are considering the view for order details for Customer = 1

```
CREATE VIEW V_Order_History AS
SELECT Restaurant_Name,
        `order`.Order_ID as `Order ID`,
        `order`.OrderTotal `Order total`,
        Date(`order`.Order_Timestamp) AS `Order Placed Date`,
        ref_order_status.Order_Status_Description AS `Status`,
        Concat(Street, ' ', City, ' ',State, ' ',ZIP, ' ', Country) As `Restaurant Address`,
        payment_Method_Name AS `Used Payment Method`
From  `Order`,
        Restaurant,
        ref_order_status,
        Address,
        payment_method_used,
        orderline,
        menu,
        customer
Where `order`.order_ID = orderline.order_ID
AND   orderline.menu_ID = menu.menu_ID
AND   menu.restaurant_ID = restaurant.restaurant_ID
AND   restaurant.address_ID = address.address_ID
AND   `order`.pay_Method_ID =  payment_method_used.pay_method_ID
AND   `order`.Order_Status_ID = ref_order_status.Order_Status_ID
AND  `order`.Customer_ID = Customer.Customer_ID
AND   Customer.Customer_ID = 1;

Select * from V_Order_History;
```

Restaurant_Name	Order ID	Order total	Order Placed Date	Status	Restaurant Address	Used Payment Method
Supreme Pizza	1	99.99	2017-02-01	Placed	St. Germain Boston Massachusetts 02115 US	Credit Card
Regina Pizzeria	2	99.99	2016-02-08	Placed	Beacon Street Boston Massachusetts 02134 US	Credit Card
SUBWAY	4	20.00	2017-04-09	Placed	290 Washington St. Boston Massachusetts 021...	Credit Card
SUBWAY	5	26.00	2017-12-13	Cancelled	290 Washington St. Boston Massachusetts 021...	Credit Card

Stored Procedure

1. Analyze the restaurants rating

-Text search analysis based on what people say in rating about the restaurent

Step1:

```
CREATE TABLE IF NOT EXISTS `mydb`.`Rating_Restaurent` (
        `Rating` INT NOT NULL,
        `Rating_Comments` VARCHAR(100) NULL,
        `Customer_ID` INT NOT NULL,
        `Restaurant_ID` INT NOT NULL,
        FULLTEXT (Rating_comments),
        INDEX `fk_Rating_Customer1_idx` (`Customer_ID` ASC),
        PRIMARY KEY (`Customer_ID`, `Restaurant_ID`),
        CONSTRAINT `fk_Rating_Customer1`
        FOREIGN KEY (`Customer_ID`)
        REFERENCES `mydb`.`Customer` (`Customer_ID`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION,
        CONSTRAINT `fk_Rating_Restaurant1`
        FOREIGN KEY (`Restaurant_ID`)
        REFERENCES `mydb`.`Restaurant` (`Restaurant_ID`)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Step2: INSERT data in table

```
select * from rating_restaurent;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

Rating	Rating_Comments	Customer_ID	Restaurant_ID
5	Timing was accurate and Food was good. It wa...	1	1
5	Food was good. and was fresh	1	3
4	Order was accurate and on time	2	1
4	Order was accurate. It was on time as well	2	5
1	Worst Food	4	1
1	Worst Food	4	5
5	Food was good. I had shushi	5	1
2	Bellow Average	5	3
5	Quality was good and Food was good	5	5
2	Quality was Bellow Average	5	7
3	Food was not good I did not like it	6	3
4	I ordered from this reataurant and Order was a...	6	4
1	Worst Food i had	6	5
3	Food was not good. I enjoyed the food	6	6
3	Order came on time and Food was not good	7	3
4	Order was accurate. and also on time	7	5
2	it was not good at all Bellow Average	8	2
3	Food was not good. It was not delevered on ti...	8	6
1	it was the Worst Food I ever had	9	1
2	Bellow Average. I did not enjoy it	9	3
NULL	NULL	NULL	NULL

rating_restaurent 115

From the above table we can search the string "Food was good"

```
select Rating_comments
FROM rating_restaurent
inner join Restaurant
on Restaurant.Restaurant_ID = rating_restaurent.Restaurant_ID
WHERE MATCH (Rating_comments)
AGAINST ('"Food was good"' IN NATURAL LANGUAGE MODE)
and Restaurant.restaurant_name = 'Supreme Pizza';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Rating_comments
Timing was accurate and Food was good. It wa...
Food was good. I had shushi

Stored procedure to find Percentage of given text in database

Whatever text is to be searched to find how many people have said it about an restaurent can be passed as an input parameter.

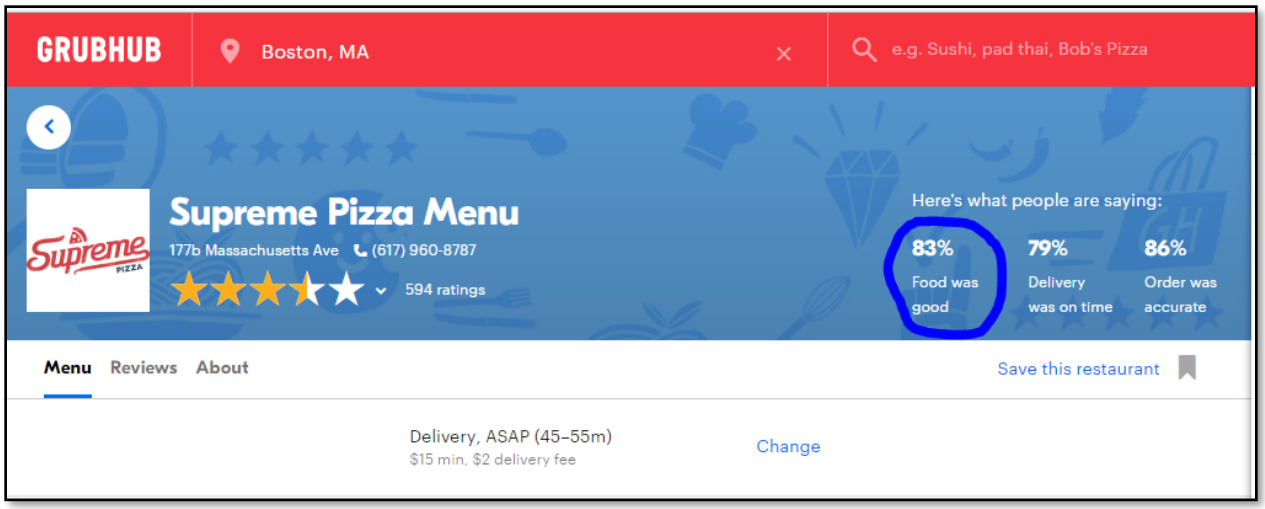
Here we are passing 2 inputs text “Food was good” and restaurant name “supreme pizza”

```
DELIMITER %%
CREATE PROCEDURE sp_rating_percentage
(IN text_to_search varchar(30),IN rest_name varchar(40), OUT percentage int)
BEGIN
select concat((round(((count(Rating_comments)/(select count(Rating_comments)
                                                FROM rating_restaurent
                                                inner join Restaurant
                                                on Restaurant.Restaurant_ID =
                                                rating_restaurent.Restaurant_ID
                                                where Restaurant_Name = Rest_name)) *
                                                100),0)),' %')
                                                AS `Precentage of given text in database`
FROM rating_restaurent
inner join Restaurant
on Restaurant.Restaurant_ID = rating_restaurent.Restaurant_ID
WHERE MATCH (Rating_comments)
AGAINST (text_to_search IN NATURAL LANGUAGE MODE)
AND Restaurant_Name = Rest_name;
 end %%
DELIMITER ;

call sp_rating_percentage('"Food was good"','Supreme Pizza', @percentage);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Precentage of given text in database
	40 %

So 40% people say that Supreme Pizza Restaurant has good food



2. Get list of all menu items sold in restaurants

- List of all details contain
  1. Restaurant Name
  2. The Category of menu in it
  3. Actual items to be sold
  4. Price of each menu
  5. Cuisine

```
DELIMITER %%
CREATE PROCEDURE sp_find_rest_details()
BEGIN
        DROP view IF EXISTS V_details_of_rest;
        Create View V_details_of_rest AS
        SELECT  Restaurant_Name,
                Menu.Menu_Type `Menu catagory`,
                Menu.menu_name `Menu`,
                Menu.Price `cost of item`,
                Cuisine.Cuisine_name `Cuisine`
        From    Restaurant
        Left outer join Menu on menu.restaurant_ID = restaurant.restaurant_ID
        Left outer join Cuisine on menu.cuisine_ID = cuisine.cuisine_ID;
 end %%
DELIMITER ;

call sp_find_rest_details();

Select * from V_details_of_rest;
```

Restaurant_Name	Menu catagory	Menu	cost of item	Cuisine
Supreme Pizza	Pizza	Veggie delight pizza	14.00	Pizza
Regina Pizzeria	Pizza	Buffalo Chicken Supreme Pizza	12.00	Pizza
Regina Pizzeria	Pizza	Buffalo Chicken Supreme Pizza	17.00	Pizza
New York Pizza	Specialty Pizza and Calzones	Cheese	14.00	Pizza
Wok n Talk	Specialty Pizza and Calzones	Chicken Broccoli Ziti	20.00	Pizza
SUBWAY	Appetizers	Preserved Egg with Chilled Tofu	12.95	Asian
SUBWAY	Soup	Clam Soup	11.95	Asian
Boston Shawarma	Drinks	Almond Bubble Tea	4.50	Asian
Boston Shawarma	Salad Bowls	Caesar Salad Bowl	6.49	American
Cheesecake Factory	Family Meals	Meal for 6	46.49	American
Cheesecake Factory	Espresso	Flat White Coffee	12.99	Bakery
IHOP	Fresh Salads	Shawarma Salad	9.00	Mediterranean
BiMediterranean	Signature Sushi Burritos	House Tempura Burrito	10.00	Japanesse
BiMediterranean	Signature Sushi Burritos	House veg Burrito	10.00	Japanesse

3. Procedure to calculate revenue a REATAURENT

- The procedure calculates revenue of a restaurent generated on any particular date

```
DELIMITER %%
CREATE PROCEDURE Revenue (  IN Restaurant_Name varchar(30),
                            IN Date1 varchar(10),
                            OUT total_Revenue int)
BEGIN
        Select sum(`order`.OrderTotal) as `Revenue of Restaurant`
        From `Order`
        INNER join orderline ON `Order`.Order_ID = OrderLine.Order_ID
        INNER join menu ON OrderLine.Menu_ID = Menu.Menu_ID
        INNER JOIN Restaurant ON Restaurant.Restaurant_ID = Menu.Restaurant_ID
        WHERE Restaurant.Restaurant_Name = Restaurant_Name
        AND Date(`order`.Order_timestamp) = date1
        AND `order`.Order_status_ID = (SELECT Order_Status_ID
                                       FROM ref_order_status
                                       WHERE Order_Status_Description = 'Placed');
 END %%
DELIMITER ;

CALL Revenue('Supreme Pizza','2017-02-01', @total_Revenue);
```

Revenue of Restaurant
99.99

Trigger

1. Trigger to update delivery details after an order is places

Below figure shows status of table before order is placed (before trigger runs):

```
select * from delivery;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Delivery_ID	Delivery_Start_Time	Delivery_End_Time	Employee_ID	Delivery_Status_ID	Address_ID	Order_ID
1	2017-02-01 11:20:20	2017-02-01 11:20:30	4	1	5	1
2	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	7	2
3	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	8	3
4	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	9	4
5	2017-12-13 02:34:11	0000-00-00 00:00:00	2	1	10	7
6	2017-12-13 02:35:15	0000-00-00 00:00:00	2	1	10	7

elivery 64

Event of trigger is when order is placed , i.e. when order table is inserted with data

```
delimiter //
create trigger t_after_order_placed
after insert on `order`
for each row
begin
  INSERT INTO delivery
        select '', now(),'',2,1, Address.Address_ID,`order`.Order_ID
        from `order`, customer, address, customer_address
        where `order`.Customer_ID = customer.Customer_ID
        AND customer.Customer_ID = customer_address.Customer_ID
        and customer_address.Address_ID = address.Address_ID
        order by Order_Timestamp desc
        limit 1;
end; //
```

The below screenshot shows Delivery table which has a new row added according to trigger

```
445 • INSERT INTO `ORDER` values (8,29.00,'2017-12-14 02:39:00',1,1,1);
446   //
447 • select * from delivery;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Delivery_ID	Delivery_Start_Time	Delivery_End_Time	Employee_ID	Delivery_Status_ID	Address_ID	Order_ID
1	2017-02-01 11:20:20	2017-02-01 11:20:30	4	1	5	1
2	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	7	2
3	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	8	3
4	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	9	4
5	2017-12-13 02:34:11	0000-00-00 00:00:00	2	1	10	7
6	2017-12-13 02:35:15	0000-00-00 00:00:00	2	1	10	7
7	2017-12-13 02:42:11	0000-00-00 00:00:00	2	1	10	8
NULL	NULL	NULL	NULL	NULL	NULL	NULL

2. Trigger to update delivery details after an order is cancelled

When an order is cancelled delivery should be stopped by cancelling it. So at this event we are updating the delivery table by cancelled status.

Delivery_Status_ID = 3 -- cancelled
Order_Status_ID = 2 = -- cancelled

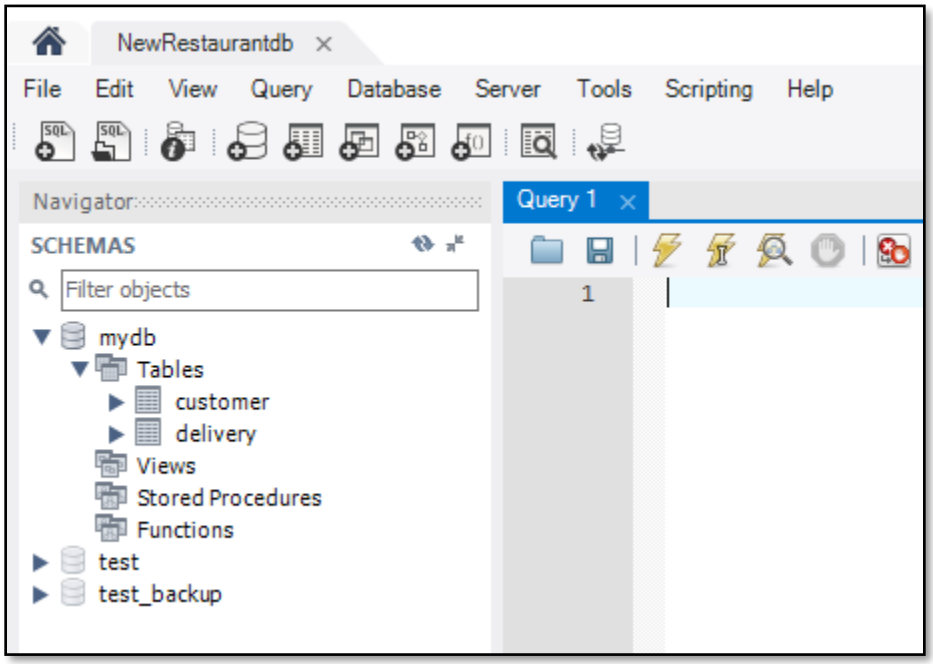
```
DROP TRIGGER IF EXISTS t_after_order_updated;

delimiter //
create trigger t_after_order_updated
after update on `order`
for each row
begin
 if (select Order_Status_ID from `order` order by Order_Timestamp desc limit 1) =2 then
        update delivery
        Set Delivery_Status_ID = 3
        where Order_ID = (select * from `order` order by Order_Timestamp desc limit 1);
        end if;
end; //

Update mydb.`order`
set Order_Status_ID = 2
where Order_ID = 12;
```

```
select * from delivery;
```

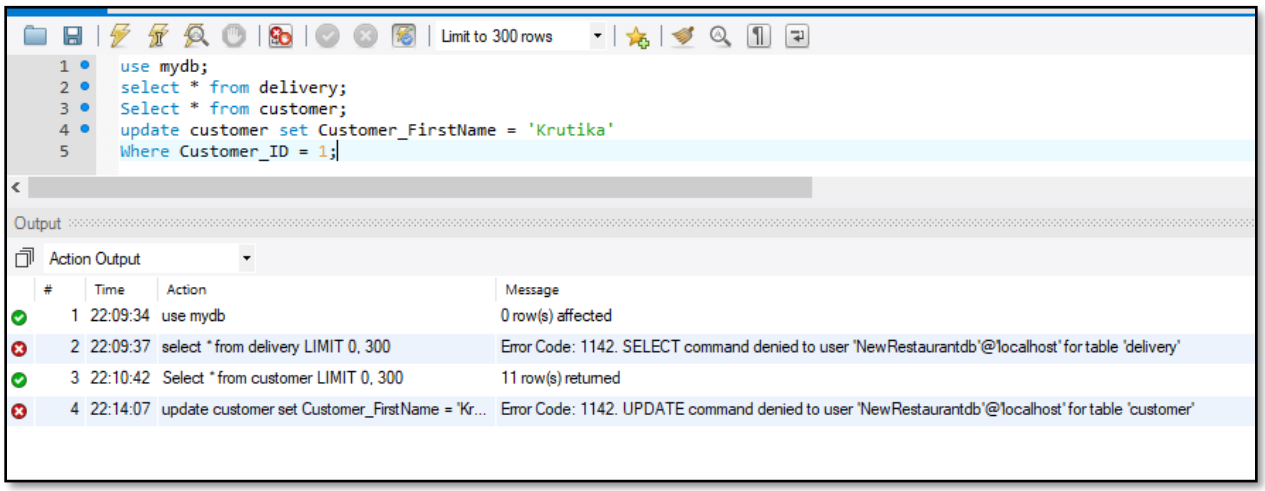
Delivery_ID	Delivery_Start_Time	Delivery_End_Time	Employee_ID	Delivery_Status_ID	Address_ID	Order_ID
1	2017-02-01 11:20:20	2017-02-01 11:20:30	4	1	5	1
2	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	7	2
3	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	8	3
4	2017-02-01 11:20:20	2017-02-01 11:20:30	4	2	9	4
7	2017-12-13 20:10:29	0000-00-00 00:00:00	2	1	10	11
8	2017-12-13 20:21:44	0000-00-00 00:00:00	2	3	10	12

Privileges

```
create user 'NewRestaurantdb'@'localhost';

revoke all privileges, grant option from 'NewRestaurantdb'@'localhost';

grant select on mydb.customer to 'NewRestaurantdb'@'localhost';
grant insert on mydb.delivery to 'NewRestaurantdb'@'localhost';

Flush privileges;
```



From the new user crested we have given select access for customer and insert access for delivery table so we get the below output for priviliges



SQL for different requirments

1. List open and closed restaurants

```
Select Restaurant_Name,
        Opening_Time,
        Closing_Time,
        current_time,
        (CASE WHEN (current_time > Opening_Time && current_time < Closing_Time)
                Then ' Restaurent is open'
            ELSE
                    ' Restaurent is Closed'
            End ) AS Restaurant_Status
From Restaurant
where Restaurant_ID IN (select Restaurant_ID from Restaurant);
```

Restaurant_Name	Opening_Time	Closing_Time	current_time	Restaurant_Status
Supreme Pizza	10:00:00	02:00:00	23:14:32	Restaurent is Closed
Regina Pizzeria	10:00:00	02:00:00	23:14:32	Restaurent is Closed
New York Pizza	11:00:00	12:00:00	23:14:32	Restaurent is Closed
Wok n Talk	12:00:00	23:00:00	23:14:32	Restaurent is Closed
SUBWAY	11:00:00	23:00:00	23:14:32	Restaurent is Closed
Boston Shawarma	11:00:00	23:00:00	23:14:32	Restaurent is Closed
Cheesecake Factory	11:00:00	23:00:00	23:14:32	Restaurent is Closed
IHOP	11:00:00	23:00:00	23:14:32	Restaurent is Closed
BiMediterranean	11:00:00	23:00:00	23:14:32	Restaurent is Closed

2. List Restaurants in the customer area (i.e near his home or office address)

- Here we are searching restaurants that are in customer 1's area

```
SELECT Restaurant_Name, Address.ZIP
FROM Restaurant
INNER JOIN Address ON restaurant.Address_ID = Address.Address_ID
Where Address.ZIP IN (SELECT Address.ZIP
                      FROM Customer
                      INNER JOIN customer_address
                      ON Customer.CUSTOMER_ID = customer_address.Customer_ID
                      INNER JOIN Address
                      ON customer_address.Address_ID = address.Address_ID
                      INNER JOIN ref_address_type
                      ON customer_address.ref_address_type_ID =
                      ref_address_type.ref_address_type_ID
                      Where Customer.CUSTOMER_ID = 1
                      AND Address_type_description IN ('Home Address','Office Address'));
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Restaurant_Name	ZIP
Supreme Pizza	02115
Boston Shawarma	02111
Cheesecake Factory	02111
IHOP	02115

3. Calculate price of order after applying discount in the cart

```
SELECT      `Order`.Order_ID,
            Round(((SUM(OrderLIne.Line_Price) * OrderLIne.Quantity) *
            (1 - (rest_Discounts.Discount_Percentage)/100)),2)
             AS Final_Price_after_discount
FROM        `Order`, OrderLine,
             Menu,
            Restaurant,
            rest_Discounts
WHERE       `Order`.Order_ID = OrderLine.Order_ID
AND         OrderLine.Menu_ID = Menu.Menu_ID
AND         Restaurant.Restaurant_ID = Menu.Restaurant_ID
AND         Restaurant.Restaurant_ID = rest_Discounts.Restaurant_ID;
```

Order_ID	Final_Price_after_discount
1	455.00

4. Top 5 Restaurants

```
Select Restaurant_Name, rating
From restaurant
Left outer join rating_restaurent
on Restaurant.Restaurant_ID = rating_restaurent.Restaurant_ID
order by Rating desc
LIMIT 5;
```

Result Grid Filter Rows:

Restaurant_Name	rating
Supreme Pizza	5
SUBWAY	4
Boston Shawarma	3
New York Pizza	2
SUBWAY	1

5. Number of orders a person has made from a particular Restaurant

```
SELECT concat(Customer_FirstName,' ',Customer_LastName) `Customer Name`,
                Restaurant_Name,
        count(`order`.Order_ID) as `number of orders`
        From Customer
         inner join `Order` on customer.Customer_ID = `order`.customer_ID
         inner join orderline on orderline.Order_ID = `order`.order_ID
         inner join menu on orderline.Menu_ID = menu.Menu_ID
         inner join restaurant on restaurant.Restaurant_ID = menu.Restaurant_ID
         where order_status_ID = (select order_status_ID from ref_order_status where
Order_Status_Description = 'Placed')
AND customer.Customer_ID = 1
AND restaurant.Restaurant_ID = 1;
```

Customer Name	Restaurant_Name	number of orders
Lia Shelton	Supreme Pizza	1