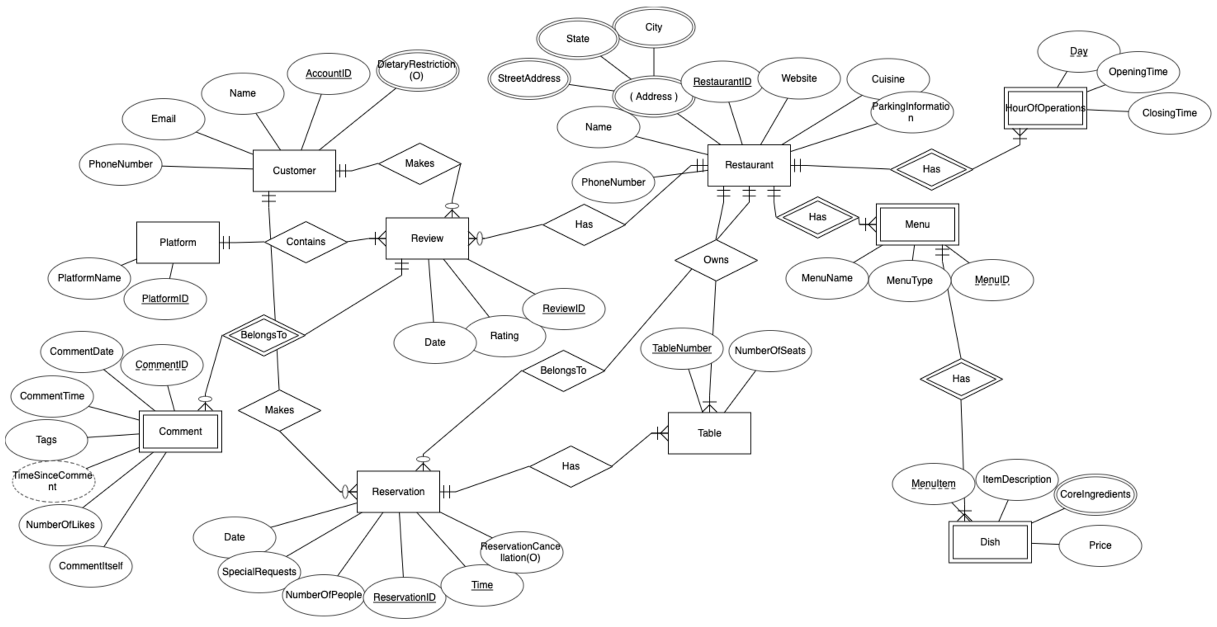
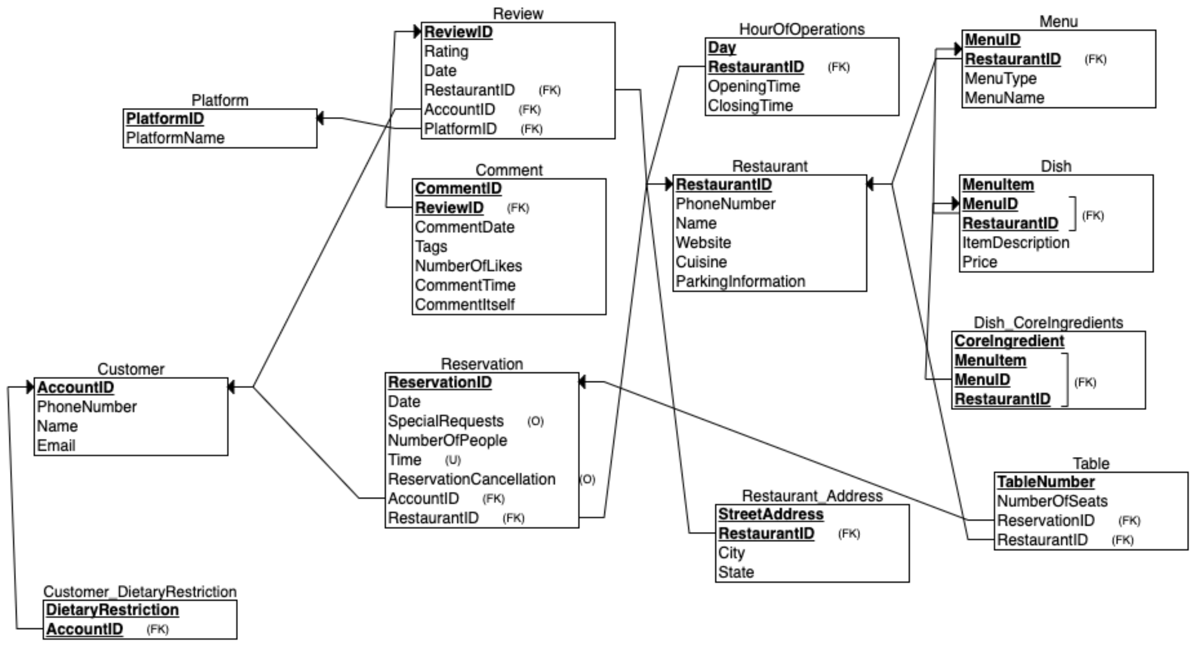
Tech Tables

Part 1)

1.





```sql
CREATE TABLE Customer
(
 PhoneNumber VARCHAR(15),
 CustomerName VARCHAR(50),
 AccountID INT NOT NULL,
 Email VARCHAR(255),
 PRIMARY KEY (AccountID)
);

CREATE TABLE Restaurant
(
 PhoneNumber VARCHAR(15),
 RestaurantName VARCHAR(50),
 RestaurantID INT NOT NULL,
 Website VARCHAR(255),
 Cuisine VARCHAR(50),
 ParkingInformation VARCHAR(1000),
 PRIMARY KEY (RestaurantID)
);

CREATE TABLE Platform
(
 PlatformID INT NOT NULL,
 PlatformName VARCHAR(50),
 PRIMARY KEY (PlatformID)
);

CREATE TABLE HourOfOperations
(
 Specificday VARCHAR(255) NOT NULL,
 OpeningTime Time,
 ClosingTime Time,
 RestaurantID INT NOT NULL,
 PRIMARY KEY (Specificday, RestaurantID),
 FOREIGN KEY (RestaurantID) REFERENCES Restaurant(RestaurantID)
);

CREATE TABLE Customer_DietaryRestriction
(
 DietaryRestriction VARCHAR(50) NOT NULL,
 AccountID INT NOT NULL,
 PRIMARY KEY (DietaryRestriction, AccountID),
 FOREIGN KEY (AccountID) REFERENCES Customer(AccountID)
);

CREATE TABLE Restaurant_Address
(
 StreetAddress VARCHAR(255) NOT NULL,
 City VARCHAR(100),
 Thestate VARCHAR(50),
 RestaurantID INT NOT NULL,
 PRIMARY KEY (StreetAddress, RestaurantID),
 FOREIGN KEY (RestaurantID) REFERENCES Restaurant(RestaurantID)
);

CREATE TABLE Review
(
 Rating INT,
 ReviewID INT NOT NULL,
 Reviewdate smalldatetime,
 RestaurantID INT NOT NULL,
 AccountID INT NOT NULL,
 PlatformID INT NOT NULL,
 PRIMARY KEY (ReviewID),
 FOREIGN KEY (RestaurantID) REFERENCES Restaurant(RestaurantID),
 FOREIGN KEY (AccountID) REFERENCES Customer(AccountID),
 FOREIGN KEY (PlatformID) REFERENCES Platform(PlatformID)
);

CREATE TABLE Menu
(
 MenuID INT NOT NULL,
 MenuType VARCHAR(50),
 MenuName VARCHAR(50),
 RestaurantID INT NOT NULL,
 PRIMARY KEY (MenuID, RestaurantID),
 FOREIGN KEY (RestaurantID) REFERENCES Restaurant(RestaurantID)
);

CREATE TABLE Reservation
(
```

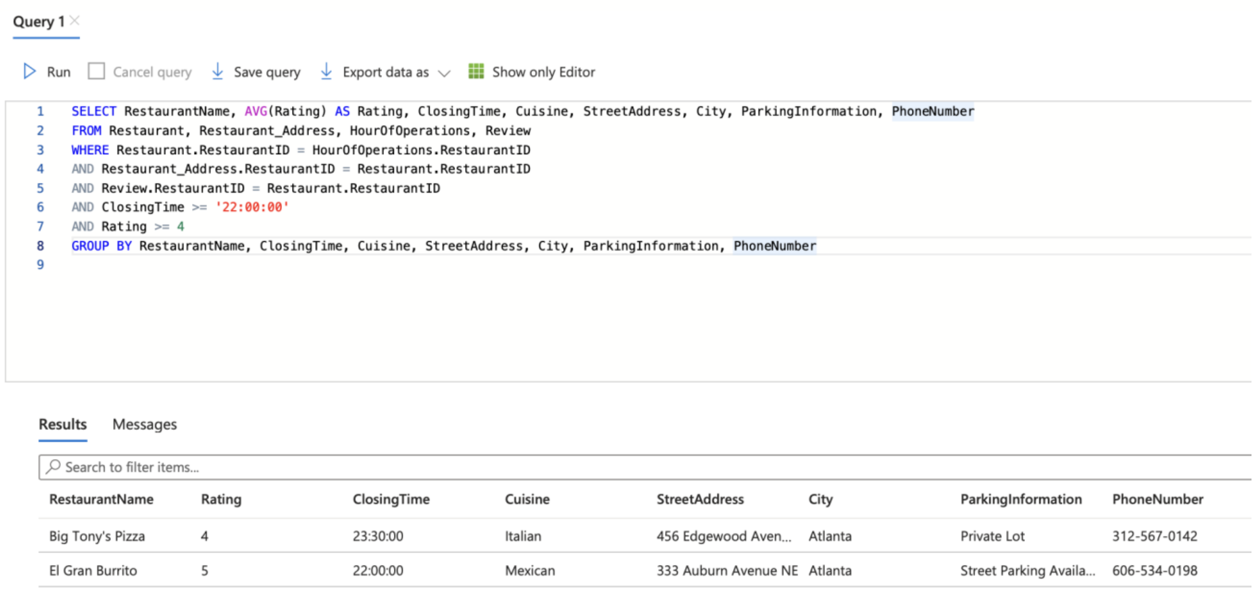
```sql
 Reservationdate date,
 SpecialRequests VARCHAR(255),
 NumberOfPeople INT,
 ReservationID INT NOT NULL,
 Reservationtime time NOT NULL,
 ReservationCancellation CHAR(1),
 AccountID INT NOT NULL,
 RestaurantID INT NOT NULL,
 PRIMARY KEY (ReservationID),
 FOREIGN KEY (AccountID) REFERENCES Customer(AccountID),
 FOREIGN KEY (RestaurantID) REFERENCES Restaurant(RestaurantID),
 UNIQUE (Reservationtime)
);

CREATE TABLE Dish
(
 MenuItem VARCHAR(100) NOT NULL,
 ItemDescription VARCHAR(255),
 Price NUMERIC(8,2),
 MenuID INT NOT NULL,
 RestaurantID INT NOT NULL,
 PRIMARY KEY (MenuItem, MenuID, RestaurantID),
 FOREIGN KEY (MenuID, RestaurantID) REFERENCES Menu(MenuID, RestaurantID)
);

CREATE TABLE Dish_CoreIngredients
(
 CoreIngredient VARCHAR(50) NOT NULL,
 MenuItem VARCHAR(100) NOT NULL,
 MenuID INT NOT NULL,
 RestaurantID INT NOT NULL,
 PRIMARY KEY (CoreIngredient, MenuItem, MenuID, RestaurantID),
 FOREIGN KEY (MenuItem, MenuID, RestaurantID) REFERENCES Dish(MenuItem,
MenuID, RestaurantID)
);

CREATE TABLE Comment
(
 CommentDate date,
 Tags VARCHAR(100),
 NumberOfLikes INT,
 CommentTime time,
 CommentID INT NOT NULL,
 CommentItself VARCHAR(1000),
 ReviewID INT NOT NULL,
 PRIMARY KEY (CommentID, ReviewID),
 FOREIGN KEY (ReviewID) REFERENCES Review(ReviewID)
);

CREATE TABLE Tablefacts
(
 TableNumber INT NOT NULL,
 NumberOfSeats INT,
 ReservationID INT NOT NULL,
 RestaurantID INT NOT NULL,
 PRIMARY KEY (TableNumber),
 FOREIGN KEY (ReservationID) REFERENCES Reservation(ReservationID),
 FOREIGN KEY (RestaurantID) REFERENCES Restaurant(RestaurantID)
);
```

```
INSERT INTO CUSTOMER VALUES ('857-555-0193', 'Tyler Johnson', 0001,
'example123@gmail.com')
INSERT INTO CUSTOMER VALUES ('312-555-0142', 'Jennifer Brown', 0002,
'jennifer.brown@yahoo.uk')
INSERT INTO CUSTOMER VALUES ('212-555-1083', 'Kyle Smith', 0003,
'kyle.smith@web.de')
INSERT INTO CUSTOMER VALUES ('906-945-2381', 'Anna Green', 0004,
'anna.green@hotmail.com')
INSERT INTO CUSTOMER VALUES ('706-993-6567', 'Jessica Foster', 0005,
'jessica.foster@gmail.com')
INSERT INTO CUSTOMER VALUES ('245-345-7531', 'Donald Trump', 0006,
'yaboiDonnie@gmail.com')
INSERT INTO Restaurant VALUES ('617-555-0135', 'The Green Elephant', 1,
'www.greenelephant.com', 'Vegan', 'Street')
INSERT INTO Restaurant VALUES ('312-567-0142', 'Big Tony''s Pizza', 2,
'www.bigtonyspizza.com', 'Italian', 'Private Lot')
INSERT INTO Restaurant VALUES ('212-521-0183', 'Mama Mia''s Trattoria', 3,
'www.mamamiastrattoria.com', 'Italian', 'Valet')
INSERT INTO Restaurant VALUES ('317-489-9820', 'Sushi House', 4,
'www.sushihouse.com', 'Japanese', 'Garage')
INSERT INTO Restaurant VALUES ('713-995-0125', 'Georgia BBQ Company', 5,
'www.georgiabbqco.com', 'Barbecue', 'Parking Lot')
INSERT INTO Restaurant VALUES ('606-534-0198', 'El Gran Burrito', 6,
'www.elgranburrito.com', 'Mexican', 'Street Parking Available')
INSERT INTO Platform VALUES (1, 'Yelp')
INSERT INTO Platform VALUES (2, 'Google Reviews')
```

Query 1:

Business Justification: This query is designed for platform users to quickly access a list of restaurants that are open late night and have a 4 star rating or higher. This is useful for students who are up late at night studying and want to see what is good and available. The user will also see the address and phone number so they can decide to order at the closest restaurant that meets their criteria.

```
SELECT RestaurantName, AVG(Rating) AS Rating, ClosingTime, Cuisine, StreetAddress, City,
ParkingInformation, PhoneNumber
FROM Restaurant, Restaurant_Address, HourOfOperations, Review
WHERE Restaurant.RestaurantID = HourOfOperations.RestaurantID AND
Restaurant_Address.RestaurantID = Restaurant.RestaurantID AND Review.RestaurantID =
Restaurant.RestaurantID AND ClosingTime >= '22:00:00' AND Rating >= 4
GROUP BY RestaurantName, ClosingTime, Cuisine, StreetAddress, City, ParkingInformation,
PhoneNumber
```

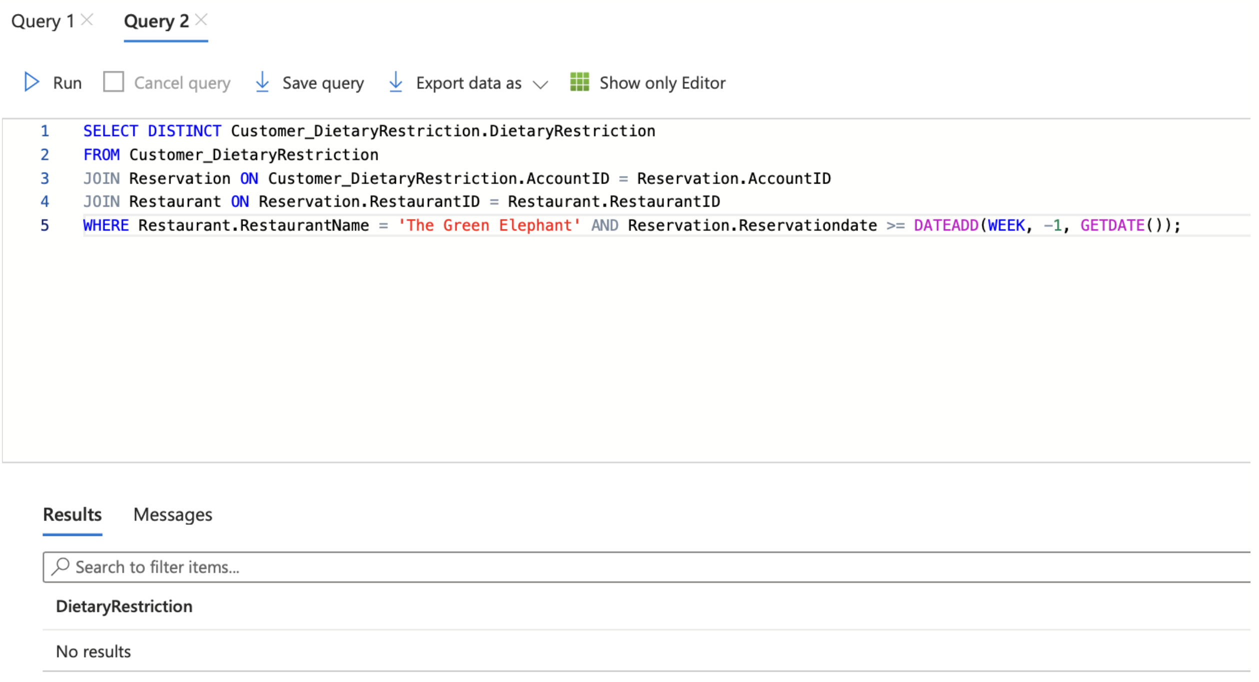


```
SELECT RestaurantName, AVG(Rating) AS Rating, ClosingTime, Cuisine, StreetAddress, City, ParkingInformation, PhoneNumber
FROM Restaurant, Restaurant_Address, HourOfOperations, Review
WHERE Restaurant.RestaurantID = HourOfOperations.RestaurantID
AND Restaurant_Address.RestaurantID = Restaurant.RestaurantID
AND Review.RestaurantID = Restaurant.RestaurantID
AND ClosingTime >= '22:00:00'
AND Rating >= 4
GROUP BY RestaurantName, ClosingTime, Cuisine, StreetAddress, City, ParkingInformation, PhoneNumber
```

RestaurantName	Rating	ClosingTime	Cuisine	StreetAddress	City	ParkingInformation	PhoneNumber
Big Tony's Pizza	4	23:30:00	Italian	456 Edgewood Aven...	Atlanta	Private Lot	312-567-0142
El Gran Burrito	5	22:00:00	Mexican	333 Auburn Avenue NE	Atlanta	Street Parking Availa...	606-534-0198

Query 2: **This query does NOT have an output based on our current data and it's more of a hypothetical example of what could be possible.**
Business Justification: This query can be useful for helping identify the dietary restrictions of customers who have recently made a reservation at a particular restaurant (in this case "The Green Elephant"). This information will help the restaurant prepare for future customer needs in advance and ensure they have appropriate menu items available and help order the necessary ingredients.

```sql
SELECT DISTINCT Customer_DietaryRestriction.DietaryRestriction
FROM Customer_DietaryRestriction
JOIN Reservation ON Customer_DietaryRestriction.AccountID = Reservation.AccountID
JOIN Restaurant ON Reservation.RestaurantID = Restaurant.RestaurantID
WHERE Restaurant.RestaurantName = 'The Green Elephant' AND Reservation.Reservationdate >=
DATEADD(WEEK, -1, GETDATE());
```



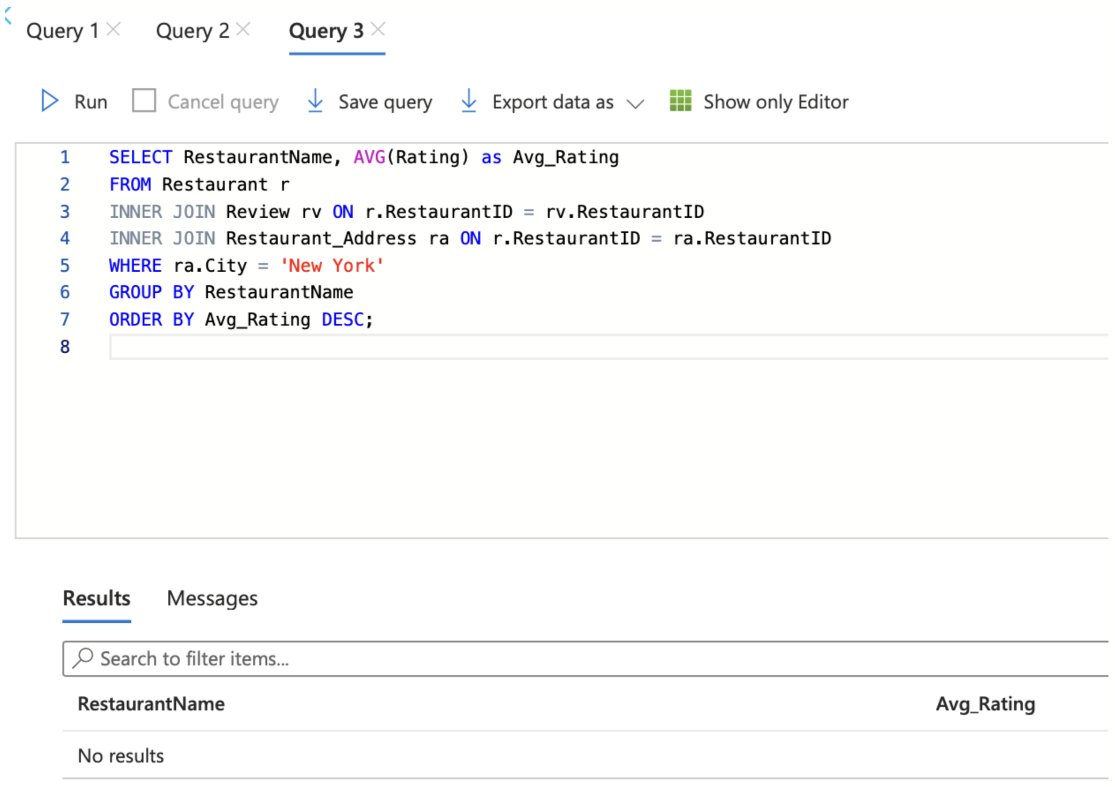
```sql
SELECT DISTINCT Customer_DietaryRestriction.DietaryRestriction
FROM Customer_DietaryRestriction
JOIN Reservation ON Customer_DietaryRestriction.AccountID = Reservation.AccountID
JOIN Restaurant ON Reservation.RestaurantID = Restaurant.RestaurantID
WHERE Restaurant.RestaurantName = 'The Green Elephant' AND Reservation.Reservationdate >= DATEADD(WEEK, -1, GETDATE());
```

```
DietaryRestriction
No results
```

Query 3:
Business Justification: **This query does NOT have an output based on our current data and it's more of a hypothetical example of what could be possible.**
This query will find the top-rated restaurants in a specific city, helping a restaurant aggregator platform to find the top-rated restaurants in a specific city. The platform can use this information to recommend these restaurants to its users, thereby

enhancing their user experience. For instance, when visiting a new town a user may want to pick a restaurant that has the best reviews (highly rated) within the city.

```
SELECT RestaurantName, AVG(Rating) as Avg_Rating
FROM Restaurant r
INNER JOIN Review rv ON r.RestaurantID = rv.RestaurantID
INNER JOIN Restaurant_Address ra ON r.RestaurantID = ra.RestaurantID
WHERE ra.City = 'New York'
GROUP BY RestaurantName
ORDER BY Avg_Rating DESC;
```

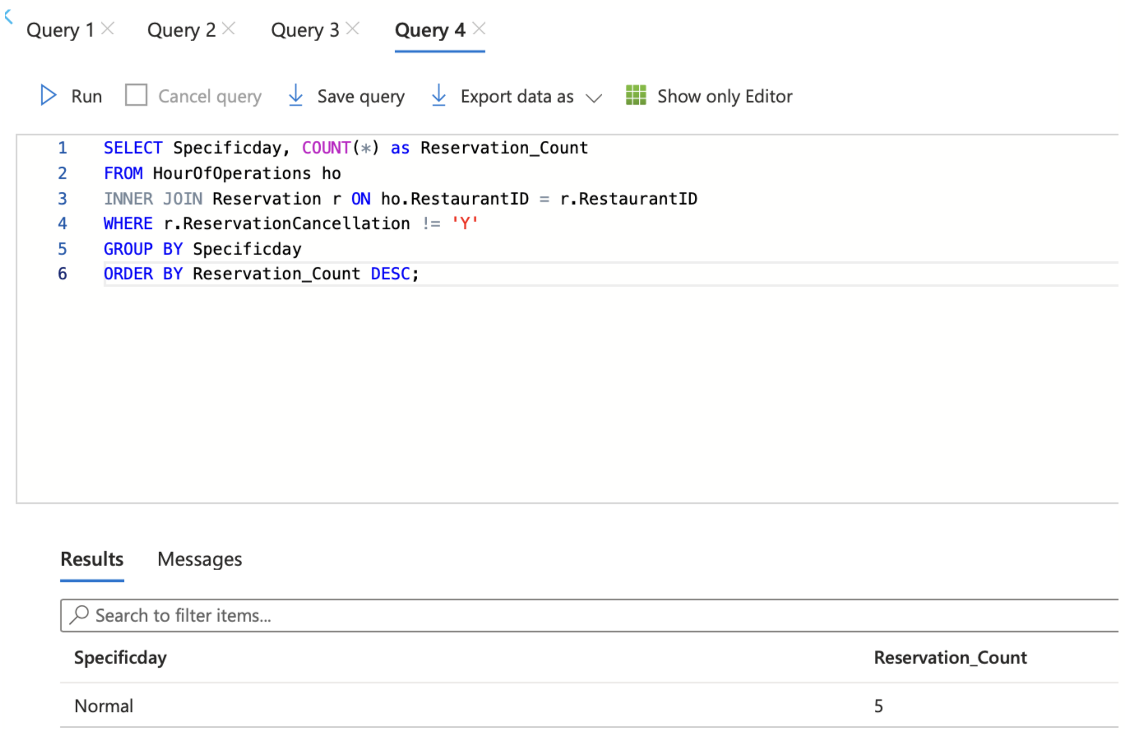


```
SELECT RestaurantName, AVG(Rating) as Avg_Rating
FROM Restaurant r
INNER JOIN Review rv ON r.RestaurantID = rv.RestaurantID
INNER JOIN Restaurant_Address ra ON r.RestaurantID = ra.RestaurantID
WHERE ra.City = 'New York'
GROUP BY RestaurantName
ORDER BY Avg_Rating DESC;
```

RestaurantName	Avg_Rating
No results	

<u>Query 4:</u>
<u>B</u>usiness Justification:
This query will help a restaurant to determine the busiest days of the week. The restaurant can use this information to allocate resources effectively, manage staff schedules, and improve the customer experience. A business may want to offer discounts for when days it isn't the busiest.

```
SELECT Specificday, COUNT(*) as Reservation_Count
FROM HourOfOperations ho
INNER JOIN Reservation r ON ho.RestaurantID = r.RestaurantID
WHERE r.ReservationCancellation != 'Y'
GROUP BY Specificday
ORDER BY Reservation_Count DESC;
```



```
SELECT Specificday, COUNT(*) as Reservation_Count
FROM HourOfOperations ho
INNER JOIN Reservation r ON ho.RestaurantID = r.RestaurantID
WHERE r.ReservationCancellation != 'Y'
GROUP BY Specificday
ORDER BY Reservation_Count DESC;
```

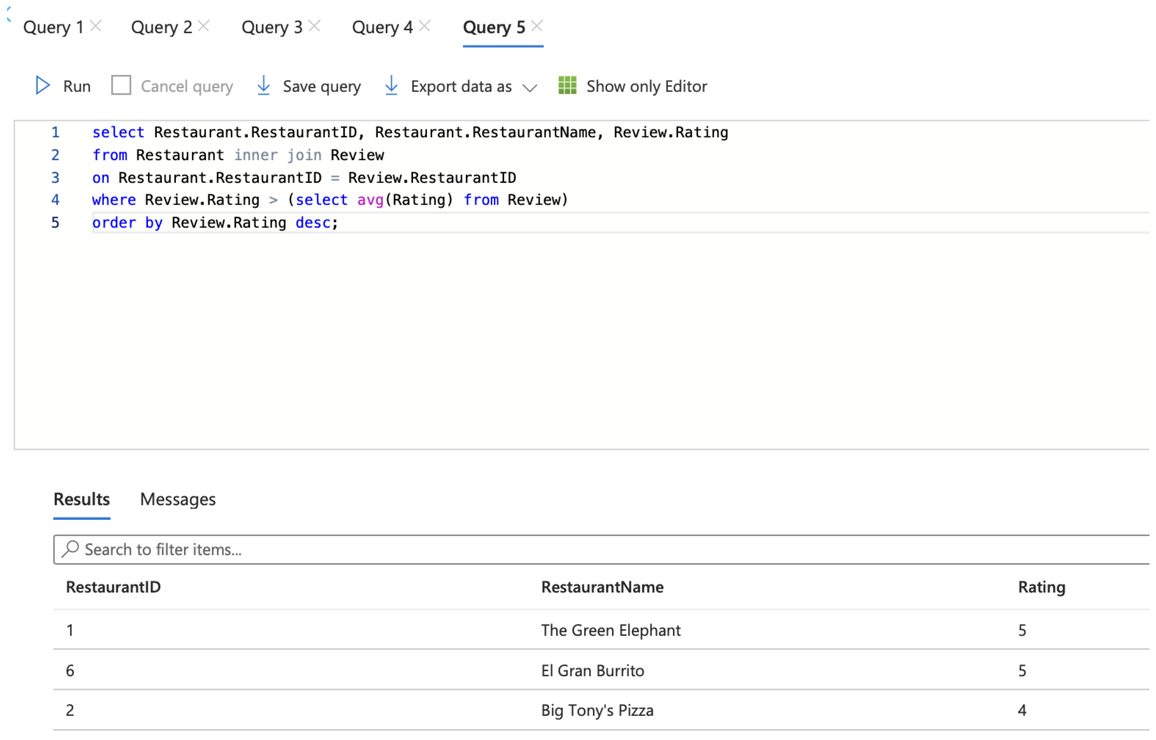
```
Specificday                                                    Reservation_Count
Normal                                                         5
```

Query 5:

Business Justification:

This query could help users find the top rated restaurants, filtering only the restaurants that have a higher rating than the average rating of all restaurants. After getting the results, the user can then look through them and make a choice, knowing that they're only looking at highly rated restaurants.

```sql
select Restaurant.RestaurantID, Restaurant.RestaurantName, Review.Rating
from Restaurant inner join Review
on Restaurant.RestaurantID = Review.RestaurantID
where Review.Rating > (select avg(Rating) from Review)
order by Review.Rating desc;
```



```sql
select Restaurant.RestaurantID, Restaurant.RestaurantName, Review.Rating
from Restaurant inner join Review
on Restaurant.RestaurantID = Review.RestaurantID
where Review.Rating > (select avg(Rating) from Review)
order by Review.Rating desc;
```

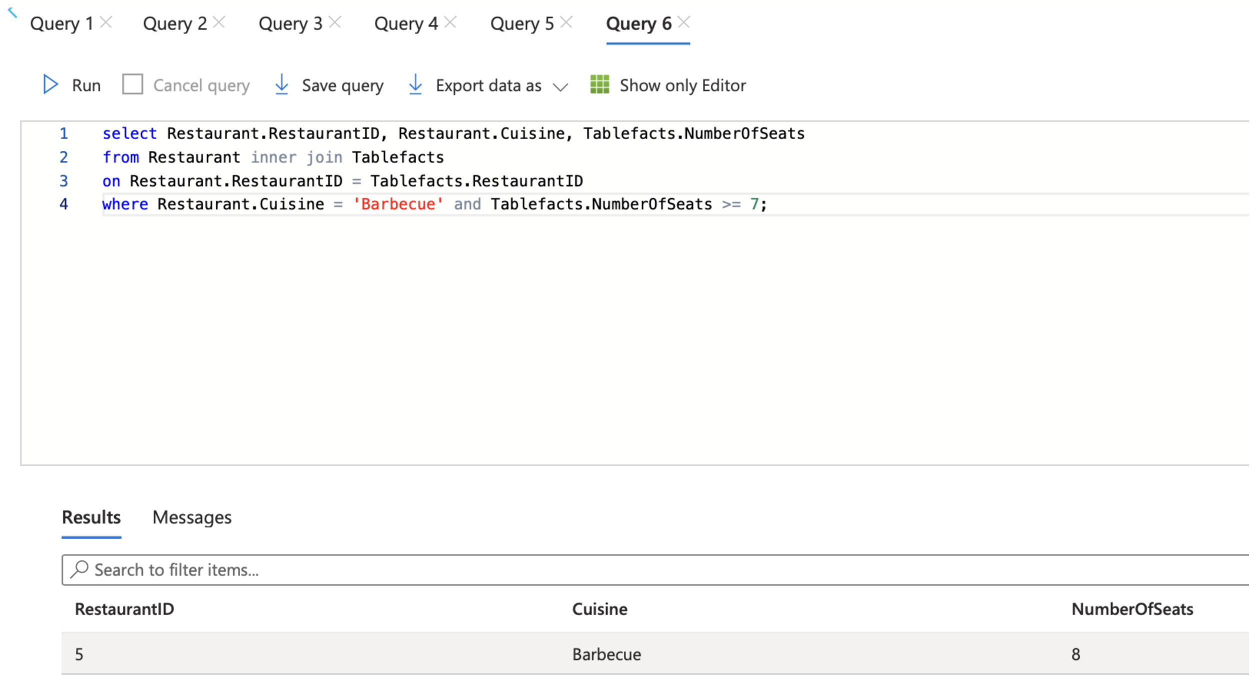
RestaurantID	RestaurantName	Rating
1	The Green Elephant	5
6	El Gran Burrito	5
2	Big Tony's Pizza	4

Query 6:

Business Justification:

This query serves as an example for a potential customer's search. In this example, the user is looking for a restaurant that serves a specific cuisine and has enough space for their own party. This query could be customized in many ways depending on what the individual user specifically wants.

```
select Restaurant.RestaurantID, Restaurant.Cuisine, Tablefacts.NumberOfSeats
from Restaurant inner join Tablefacts
on Restaurant.RestaurantID = Tablefacts.RestaurantID
where Restaurant.Cuisine = 'Barbecue' and Tablefacts.NumberOfSeats >= 7;
```



```
select Restaurant.RestaurantID, Restaurant.Cuisine, Tablefacts.NumberOfSeats
from Restaurant inner join Tablefacts
on Restaurant.RestaurantID = Tablefacts.RestaurantID
where Restaurant.Cuisine = 'Barbecue' and Tablefacts.NumberOfSeats >= 7;
```

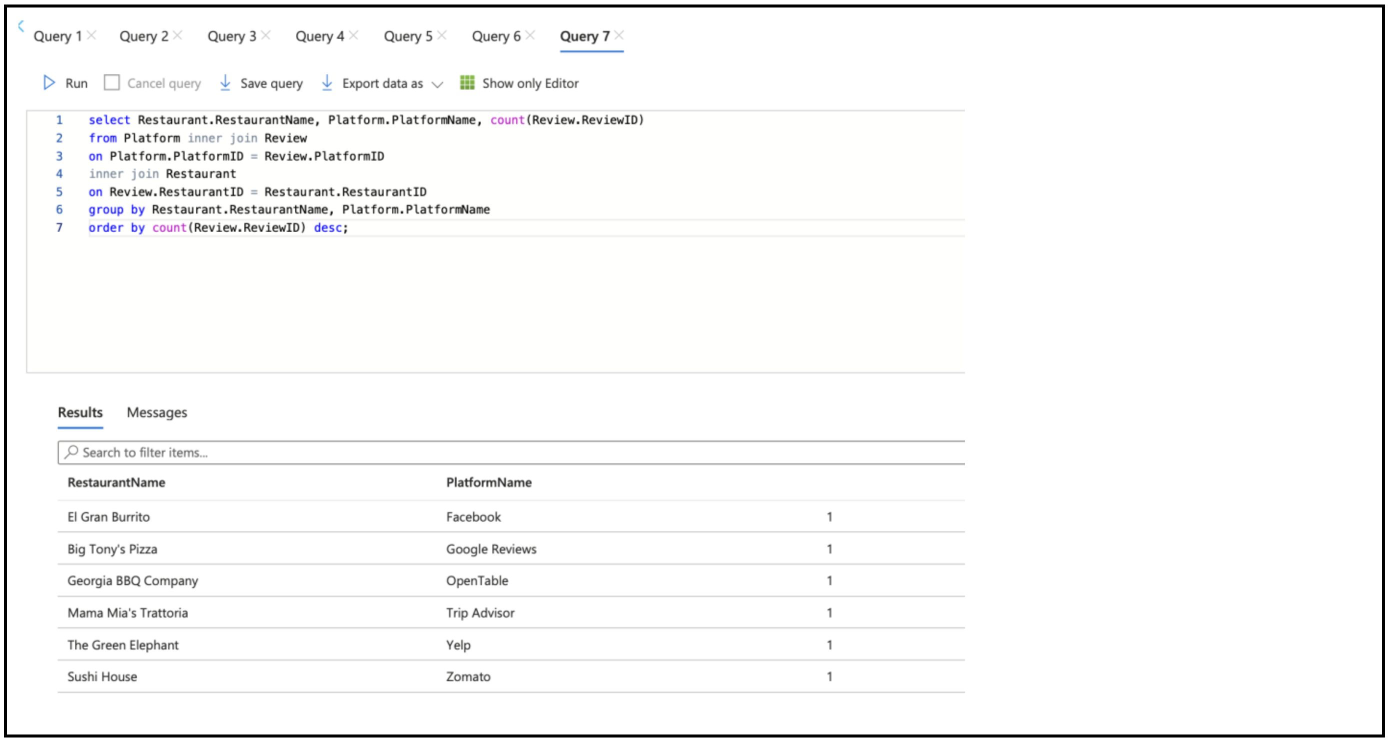
RestaurantID	Cuisine	NumberOfSeats
5	Barbecue	8

Query 7:

Business Justification:

This query simply shows the number of reviews on each platform for each individual restaurant. This could be useful information for restaurants to know as they would be able to see on what platform their customers are most active on as well as having the ability to develop various marketing strategies to keep pushing on the platforms they're already strong on.

```
select Restaurant.RestaurantName, Platform.PlatformName, count(Review.ReviewID)
from Platform inner join Review
on Platform.PlatformID = Review.PlatformID
inner join Restaurant
on Review.RestaurantID = Restaurant.RestaurantID
group by Restaurant.RestaurantName, Platform.PlatformName
order by count(Review.ReviewID) desc;
```


Query 1 Query 2 Query 3 Query 4 Query 5 Query 6 Query 7

Run | Cancel query | Save query | Export data as | Show only Editor

```sql
select Restaurant.RestaurantName, Platform.PlatformName, count(Review.ReviewID)
from Platform inner join Review
on Platform.PlatformID = Review.PlatformID
inner join Restaurant
on Review.RestaurantID = Restaurant.RestaurantID
group by Restaurant.RestaurantName, Platform.PlatformName
order by count(Review.ReviewID) desc;
```

Results | Messages

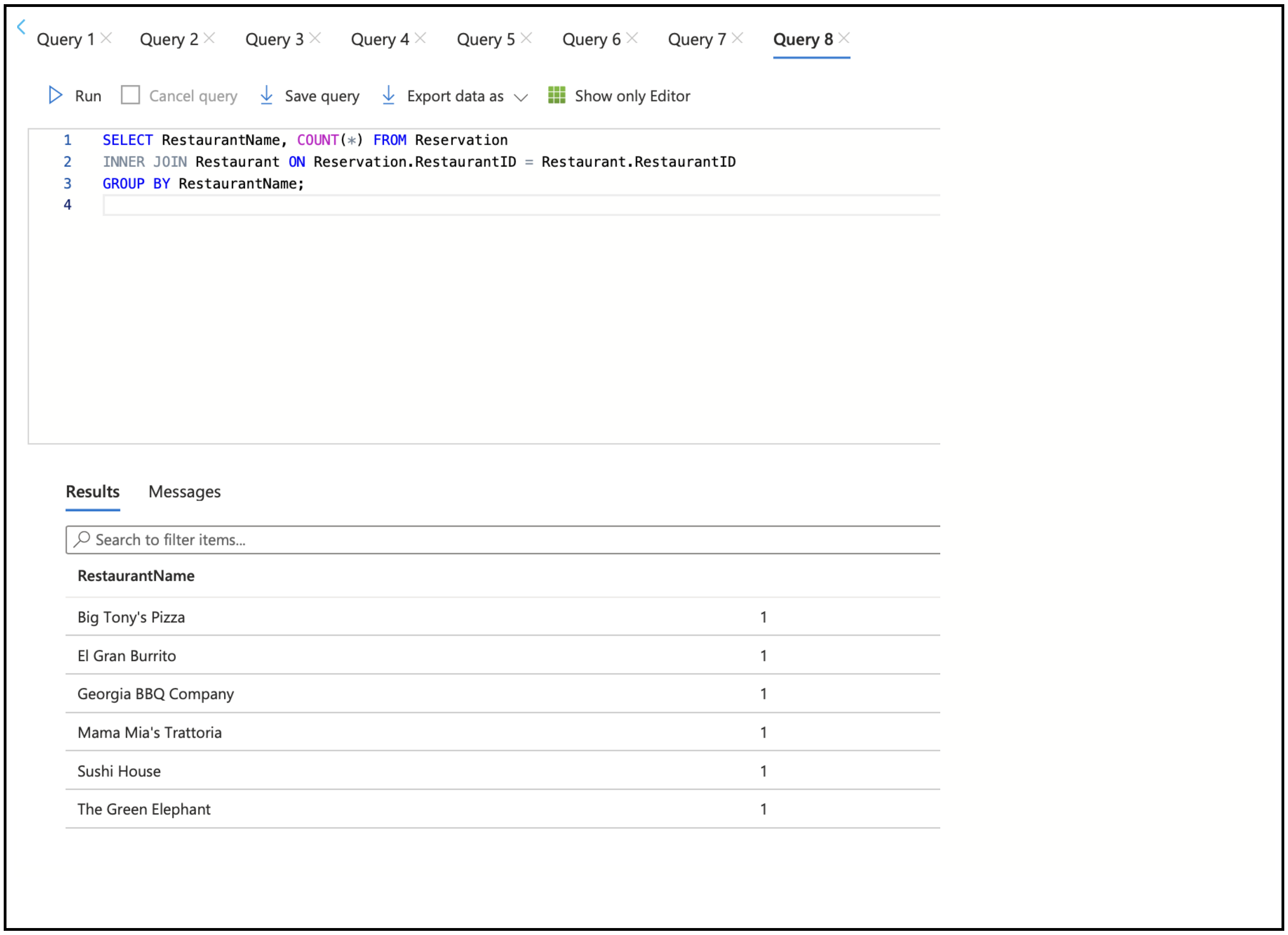
RestaurantName	PlatformName	
El Gran Burrito	Facebook	1
Big Tony's Pizza	Google Reviews	1
Georgia BBQ Company	OpenTable	1
Mama Mia's Trattoria	Trip Advisor	1
The Green Elephant	Yelp	1
Sushi House	Zomato	1

Query 8:

Business Justification:

This query is useful to get the number of reservations for each restaurant. This information can be used to understand the popularity of each restaurant and can help in staffing and resource allocation decisions for the restaurant or help customers understand which restaurants tend to get the most busy.

```sql
SELECT RestaurantName, COUNT(*) FROM Reservation
INNER JOIN Restaurant ON Reservation.RestaurantID = Restaurant.RestaurantID
GROUP BY RestaurantName;
```



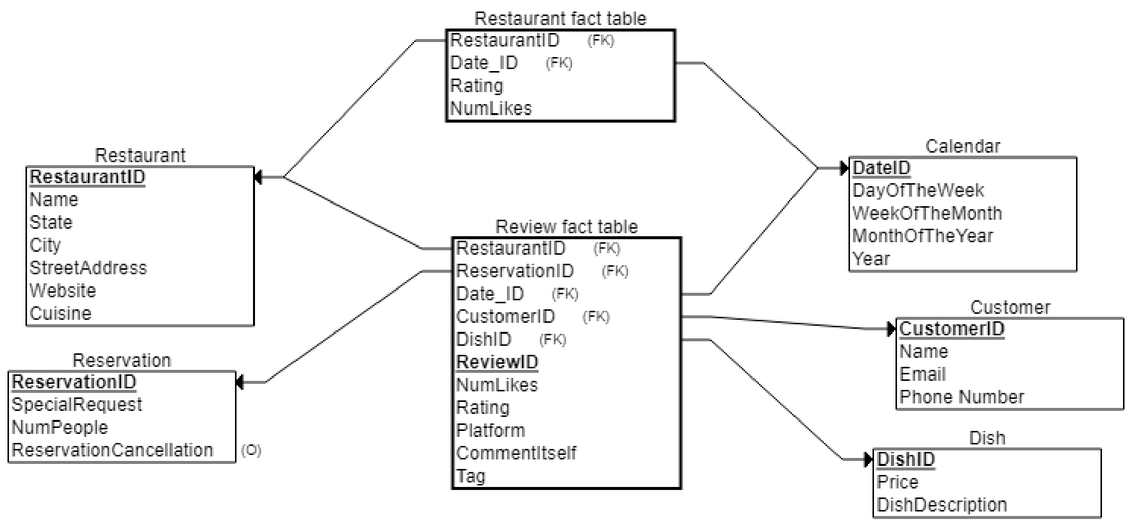
```sql
SELECT RestaurantName, COUNT(*) FROM Reservation
INNER JOIN Restaurant ON Reservation.RestaurantID = Restaurant.RestaurantID
GROUP BY RestaurantName;
```

RestaurantName	
Big Tony's Pizza	1
El Gran Burrito	1
Georgia BBQ Company	1
Mama Mia's Trattoria	1
Sushi House	1
The Green Elephant	1

Part 2)

2. There are two fact tables. The subject of analysis for the review fact table are the reviews themselves. The subject of analysis for the other fact table is each restaurant.

3.



4.

The Review fact table includes very detailed data. The restaurant, the reservation, the date, the customer who made the reservation and the dish associated with the review will be correlated with each review. Also, the number of likes, the platform, the comment (textual data), the rating, and the tag will be correlated with each review. So, it is a very granular fact table that will allow for aggregations for further analysis. The restaurant fact table is aggregated on the restaurant level. So, the rating for each restaurant will be very clear along with the amount of likes associated with each restaurant. For example, if a restaurant has a rating of four out of five stars, the data will show that the reviews have 896 likes. On the contrary, a restaurant could get five out of five stars, but have 5 likes on those reviews leading to a less credible review. Also, the date of the aggregation is noted, so an investigation could be made if a restaurant's rating goes significantly down or up. This is to prevent fake negative reviews or fake positive reviews.

5.

- For an example of Extract, Transform, and Load, we will look at constructing the Restaurant Fact Table

- For extraction from the operational database, the reviews in a certain time period need to be grouped by RestaurantID. The number of likes and rating will be other fields that will be aggregated on the restaurant level. The date of the aggregation will be the DateID.
- For Transformation, any records with missing values of number of likes should be dropped. The DateID needs to be transformed to the appropriate format for analysis. There should be quality checks making sure all the data was transferred properly. All of the data should be in the correct format to fit the data warehouse.
- For Loading, the restaurant dimension should have an initial load of the data from the operational database. After this, the data should be updated every day. This is to highlight a sudden drop in a restaurant's ratings for any particular reason including false reviews.