SQL - Project

zomato

Facts and Figures of Zomato

- **Introduction**

- Zomato, a global restaurant discovery and food delivery platform, connects millions of users with a diverse array of restaurants, offering a seamless experience for exploring and enjoying culinary delights.

- I used to order food from Zomato almost every weekend when I was residing in my college hostel and am a big fan of the company's service since then. I still choose Zomato over others.

- My liking for the company pushed me to do an SQL project on Zomato.

Objective

- Solve day-to-day complex business problems of online food delivery business using MySQL.
- Highlight skills in advanced SQL.

Data Description

- Zomato has over 1.4 million listed restaurants and 12,000 restaurant partners.
- It has over 300,000 active delivery drivers and 3.5 million active restaurant listings on its platform.
- It is the 11th most visited website and 2nd most visited in India by traffic in the world for Restaurants.
- It has over 80 million monthly active users.
- Its market share in the food-serving business is 55%.
- It delivers over 12 lakh orders daily.
- It is the only Indian food delivery app that operates in 24 countries.

Tables Schema's

Column Name	Column Description
Table 1 : users	
user_id	Unique identifier for the user.
name	Name of the user.
email	Email address of the user.
password	Account password of the user

Column Name	Column Description
Table 2: restaurants	
r_id	Unique identifier for the restaurant.
r_name	Name of the restaurant.
cuisine	Cuisine of the restaurant.

Column Name	Column Description
Table 3: orders	
order_id	Unique identifier for the order.
user_id	Foreign key to the Users table.
r_id	Foreign key to the Restaurants table.
amount	Total amount of the order.
date	Date and time the order was placed.
partner_id	Foreign key to the Delivery Partners table.
delivery_time	Time it took for the order to be delivered.
delivery_rating	Rating given to the delivery partner by the user.
restraunt_rating	Rating given to the restaurant by the user.

Column Name	Column Description
Table 4: delivery_partner	
partner_id	Unique identifier for the delivery partner.
partner_name	Name of the delivery partner.

Column Name	Column Description
Table 5: order_details	
id	Unique identifier for the order detail.
order_id	Foreign key to the Orders table.
food	Name of the food item.

4. 4. Restaurants with monthly sales greater than 1000 for July.

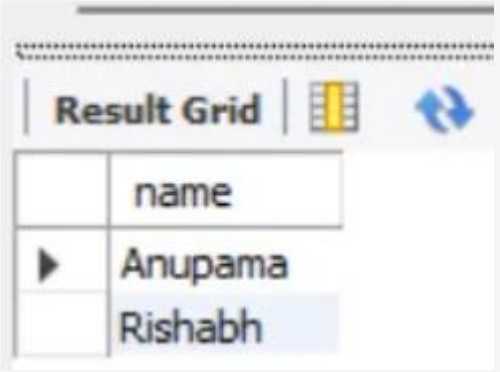
Business problems and Queries for solution

1. Find Customers who have never ordered.
2. Average Price/dish
3. Find the top restaurant in terms of the number of orders in June
4. Restaurants with monthly sales greater than 1000 for July..
5. Show all orders with order details of Nitish (user_id = 1) from 10th June'22 to 10th July'22.
6. Find restaurants with maximum repeat customers.
7. Month-over-month revenue growth of Zomato.
8. Customer — favorite food.
9. Find the most loyal customers for all restaurants.
10. Month-over-month revenue growth of each restaurant.

Question 1.

```
-------------------------------------------------------------------------
-- 1. Find customers who have never ordered.
-------------------------------------------------------------------------
SELECT
 name
FROM
 users
WHERE
 user_id NOT IN
(
SELECT user_id
FROM orders
);
```

Output : ->



Question 2.

```
-- ------------------------------------------------------------
-- 2. Average price/dish
-- ------------------------------------------------------------
SELECT
 f.f_name AS food,
 AVG(price) AS avgPrice
FROM
 menu m
JOIN
 food f ON f.f_id = m.f_id
GROUP BY
 f.f_name;
```

Output : ->

	food	avgPrice
▶	Non-veg Pizza	450.0000
	Veg Pizza	400.0000
	Choco Lava cake	98.3333
	Chicken Wings	230.0000
	Chicken Popcorn	300.0000
	Rice Meal	213.3333
	Roti meal	140.0000
	Masala Dosa	180.0000
	Rava Idli	120.0000
	Schezwan Noodles	220.0000
	Veg Manchurian	180.0000

Question 3.

```
-----------------------------------------------------------------------
-- 3. Find the top restaurant in terms of the number of orders in june.
-----------------------------------------------------------------------
SELECT
 DISTINCT r.r_name AS restraunt,
 MONTHNAME(o.date) AS month,
 COUNT(o.order_id) AS orderCount
FROM
 orders o
JOIN
 restraunts r ON r.r_id = o.r_id
WHERE
 MONTH(o.date) = 6
GROUP BY
 r.r_name
ORDER BY
 orderCount DESC
LIMIT
 1;
```

Output : ->

	restraunt	month	orderCount
▶	kfc	June	3

Question 4.

```sql
-------------------------------------------------------------------------
-- 4. Restaurants with monthly sales greater than 1000 for July.
-------------------------------------------------------------------------
SELECT
 r.r_name AS restraunt,
 SUM(o.amount) AS revenue
FROM
 orders o
JOIN
 restraunts r ON r.r_id = o.r_id
WHERE
 MONTHNAME(o.date) = 'July'
GROUP BY
 r.r_name
HAVING
 revenue > 1000 ;
```

Output : ->

	restraunt	revenue
▶	China Town	1050
	dominos	1100
	kfc	1935

Question 5.

```
-------------------------------------------------------------------------------
-- 5. Show all orders with order details of Nitish (user_id = 1) from 10th June
-------------------------------------------------------------------------------
SELECT
 u.name AS users,
 o.date AS orderDate,
 r.r_name AS restraunt,
 f.f_name AS food
FROM
 orders o
JOIN
 restraunts r ON r.r_id = o.r_id
JOIN
 users u ON u.user_id = o.user_id
JOIN
 order_details od ON od.order_id = o.order_id
JOIN
 food f ON f.f_id = od.f_id
WHERE
 u.user_id = 1
AND
 o.date BETWEEN '2022-06-10' AND '2022-07-10'
ORDER BY
 orderDate;
```

Output : ->

orderDate	users	restraunt	food
2022-06-15	Nitish	box8	Choco Lava cake
2022-06-15	Nitish	box8	Rice Meal
2022-06-29	Nitish	box8	Choco Lava cake
2022-06-29	Nitish	box8	Rice Meal
2022-07-10	Nitish	box8	Choco Lava cake
2022-07-10	Nitish	box8	Roti meal

Question 6.

```sql
-----------------------------------------------------------------------------
-- 6. Find restaurants with maximum repeat customers.
-----------------------------------------------------------------------------
SELECT
 r.r_name AS restraunt,
 COUNT(*) AS loyal_customers,
 SUM(orderCount) AS total_order_count
FROM
(
SELECT
 o.r_id, o.user_id,
 COUNT( DISTINCT o.order_id) AS orderCount
FROM
 orders o
JOIN
 restraunts r ON r.r_id = o.r_id
JOIN
 users u ON u.user_id = o.user_id
GROUP BY
 o.r_id, r.r_name, o.user_id
HAVING
 orderCount > 1
)t

JOIN
 restraunts r on t.r_id = r.r_id
GROUP BY
 restraunt
HAVING
 loyal_customers > 1;
```

Output : ->

	restraunt	loyal_customers	total_order_count
▶	kfc	2	6

Question 7.

```
-------------------------------------------------------------------------
-- 7. Month-over-month revenue growth of Zomato.
-------------------------------------------------------------------------
WITH T AS
(
SELECT
 MONTHNAME (date) AS month,
 SUM(amount) AS revenue,
 LAG (SUM(amount)) OVER (ORDER BY date) AS prevRevenue
FROM
 orders
GROUP BY
 month
ORDER BY
 date
)
SELECT
 month,
 revenue,
 ((revenue-prevRevenue)/ prevRevenue) * 100 AS 'MOM revenue growth (%)'
FROM
T;
```

Output : ->

	month	revenue	MoM revenue growth(%)
▶	May	2425	NULL
	June	3220	32.7835
	July	4845	50.4658

Question 8.

```
-------------------------------------------------------------------------
-- 8. Customer - favourite food
-------------------------------------------------------------------------
WITH T AS
(
SELECT
 o.user_id, u.name, od.order_id,
 od.f_id, f.f_name as favouriteFood,
 COUNT(od.f_id) AS orderCount,
 RANK() OVER (PARTITION BY u.name ORDER BY COUNT(od.f_id) DESC) AS orderRank
FROM
 orders o
JOIN
 order_details od ON od.order_id = o.order_id
JOIN
 food f ON f.f_id = od.f_id
JOIN
 users u ON u.user_id = o.user_id
GROUP BY
 u.user_id,
    f.f_name
ORDER BY
 u.name,
    orderCount DESC
)
SELECT
 name,
 favouriteFood,
    orderCount
FROM
 T
WHERE
 orderRank = 1;
```

Output : ->

	Name	FavoriteFood	OrderCount
▶	Ankit	Schezwan Noodles	3
	Ankit	Veg Manchurian	3
	Khushboo	Choco Lava cake	3
	Neha	Choco Lava cake	5
	Nitish	Choco Lava cake	5
	Vartika	Chicken Wings	3

Question 9.

```
-------------------------------------------------------------------------
-- 9. Find the most loyal customers for all restaurant.
-------------------------------------------------------------------------
WITH X AS
(
SELECT
 u.name, r.r_name,
 COUNT(o.r_id) AS OrderCount,
 DENSE_RANK() OVER (PARTITION BY r.r_name ORDER BY COUNT(o.r_id) DESC) AS row_ra
FROM
 orders o
JOIN
 restraunts r ON o.r_id = r.r_id
JOIN
 users u ON o.user_id = u.user_id
GROUP BY
  o.user_id,
  o.r_id
)
SELECT
 *
FROM
 X
WHERE
 row_rank = 1;
```

Output : ->

	name	restraunt	order_count	row_rank
▶	Nitish	box8	3	1
	Ankit	China Town	2	1
	Neha	dominos	2	1
	Ankit	Dosa Plaza	3	1
	Vartika	kfc	3	1
	Neha	kfc	3	1

Question 10.

```
-----------------------------------------------------------------------------
-- 10. Month-over-month revenue growth of a restaurant.
-----------------------------------------------------------------------------
WITH X AS
(
SELECT
 o.r_id, r.r_name As restraunt,
 MONTHNAME(o.date) AS monthName,
 SUM(o.amount) AS revenue,
 LAG(SUM(o.amount)) OVER (PARTITION BY r.r_name ORDER BY MONTH(o.date)) AS prevR
FROM
 orders o
JOIN
 restraunts r ON o.r_id = r.r_id
GROUP BY
 r.r_name,
 MONTHNAME(o.date)
)
SELECT
 restraunt,
 monthName,
 ((revenue - prevRevenue)/ NULLIF (prevRevenue, 0))*100 AS 'MoM revenue growth (
FROM
 X
ORDER BY
 r_id;
```

Output : ->

restraunt	monthName	MoM revenue growth(%)
dominos	May	NULL
dominos	June	-5.0000
dominos	July	15.7895
kfc	May	NULL
kfc	June	53.4884
kfc	July	95.4545
box8	June	NULL
box8	July	-4.1667
Dosa Plaza	May	NULL
Dosa Plaza	June	-48.7179
Dosa Plaza	July	-25.0000
China Town	June	NULL
China Town	July	162.5000

Question 11.

```sql
-- ------------------------------------------------------------------
-- 11. Top 3 most paired products.
-- ------------------------------------------------------------------
SELECT
    f1.f_name AS product1,
    f2.f_name AS product2,
    COUNT(o.order_id) AS pair_count
FROM
    orders o
JOIN
    order_details od1 ON o.order_id = od1.order_id
JOIN
    order_details od2 ON o.order_id = od2.order_id
JOIN
    food f1 ON f1.f_id = od1.f_id
JOIN
    food f2 ON f2.f_id = od2.f_id
WHERE
    od1.f_id < od2.f_id
GROUP BY
    f1.f_name, f2.f_name
ORDER BY
    pair_count DESC
LIMIT
    3;
```

Output : ->

product1	product2	pair_count
Choco Lava cake	Chicken Wings	5
Non-veg Pizza	Choco Lava cake	4
Schezwan Noodles	Veg Manchurian	4

Side Notes And Assumptions

- The process of analyzing data with SQL was time-consuming (though not more so than the time invested in documenting it). Good documentation, comments, and notes really helped when rechecking the results, especially for understanding the context and the thought process applied.
- Another insight: a query running successfully does not guarantee correct results. While identifying the most loyal customers for all restaurants (Q9), the user 'Ankit' was inadvertently left out of the result set due to a query mistake.
- Finally, Chicken wings and Choco lava cake are the most ordered food items together. And I vouch for that!