-- USE zomato

-- Count total order details

SELECT COUNT(\*) FROM order\_details;

-- Random 5 users (SQL Server uses NEWID())

SELECT TOP 5 \* FROM users ORDER BY NEWID();

-- Find NULL values

SELECT \* FROM orders WHERE restaurant\_rating IS NULL;

-- Replace NULL values with 0

UPDATE orders SET restaurant\_rating = 0

WHERE restaurant\_rating IS NULL;

-- Q5: Number of orders per user

SELECT t2.name, COUNT(\*) AS [#orders]

FROM orders t1

JOIN users t2 ON t1.user\_id = t2.user\_id

GROUP BY t2.user\_id, t2.name;

-- Q6: Number of menu items per restaurant

SELECT r\_name, COUNT(\*) AS menu\_items

FROM resturants t1

JOIN menu t2 ON t1.r\_id = t2.r\_id

GROUP BY t1.r\_id, r\_name;

-- Q7: Votes & Average rating per restaurant

SELECT r\_name,

COUNT(\*) AS num\_votes,

ROUND(AVG(restaurant\_rating), 2) AS rating

FROM orders t1

JOIN restaurants t2 ON t1.r\_id = t2.r\_id

WHERE restaurant\_rating IS NOT NULL

GROUP BY t1.r\_id, r\_name;

-- Q8: Most frequently ordered food

SELECT TOP 1 f\_name, COUNT(\*) AS cnt

FROM menu t1

JOIN food t2 ON t1.f\_id = t2.f\_id

GROUP BY t1.f\_id, f\_name

ORDER BY COUNT(\*) DESC;

-- Q9: Highest revenue restaurant in July

SELECT TOP 1 r\_name, SUM(amount) AS revenue

FROM orders t1

JOIN restaurants t2 ON t1.r\_id = t2.r\_id

WHERE DATENAME(MONTH, date) = 'July'

GROUP BY t1.r\_id, r\_name

ORDER BY revenue DESC;

-- Month by month revenue for a restaurant (example: box8)

SELECT DATENAME(MONTH, date) AS [Month],

SUM(amount) AS revenue

FROM orders t1

JOIN restaurants t2 ON t1.r\_id = t2.r\_id

WHERE r\_name = 'box8'

GROUP BY DATENAME(MONTH, date), MONTH(date)

ORDER BY MONTH(date);

-- Q10: Restaurants with revenue > 1500

SELECT r\_name, SUM(amount) AS revenue

FROM orders t1

JOIN restaurants t2 ON t1.r\_id = t2.r\_id

GROUP BY t1.r\_id, r\_name

HAVING SUM(amount) > 1500;

-- Q11: Users who never ordered

SELECT user\_id, name FROM users

EXCEPT

SELECT t1.user\_id, t2.name

FROM orders t1

JOIN users t2 ON t1.user\_id = t2.user\_id;

-- Q12: Orders by user 5 in date range

SELECT t1.order\_id, f\_name, date

FROM orders t1

JOIN order\_details t2 ON t1.order\_id = t2.order\_id

JOIN food t3 ON t2.f\_id = t3.f\_id

WHERE user\_id = 5 AND date BETWEEN '2022-05-15' AND '2022-07-15';

-- Q13: Most frequently ordered food by each user

SELECT t1.user\_id, t3.f\_id, COUNT(\*) AS cnt

FROM users t1

JOIN orders t2 ON t1.user\_id = t2.user\_id

JOIN order\_details t3 ON t2.order\_id = t3.order\_id

GROUP BY t1.user\_id, t3.f\_id

ORDER BY cnt DESC;

-- Q14: Restaurant with lowest avg price

SELECT TOP 1 r\_name,

SUM(price)\*1.0 / COUNT(\*) AS Avg\_price

FROM menu t1

JOIN restaurants t2 ON t1.r\_id = t2.r\_id

GROUP BY t1.r\_id, r\_name

ORDER BY Avg\_price ASC;

-- Q15: Partner salary calculation

SELECT partner\_name,

COUNT(\*) \* 100 + AVG(delivery\_rating) \* 1000 AS salary

FROM orders t1

JOIN delivery\_partner t2 ON t1.partner\_id = t2.partner\_id

GROUP BY t1.partner\_id, partner\_name

ORDER BY salary DESC;

-- Q17: Correlation (not directly supported in SQL Server)

-- Need to compute manually with formula (if needed)

-- Q19: Restaurants with only Veg items

SELECT r\_name

FROM menu t1

JOIN food t2 ON t1.f\_id = t2.f\_id

JOIN restaurants t3 ON t1.r\_id = t3.r\_id

GROUP BY t1.r\_id, r\_name

HAVING MIN(type) = 'Veg' AND MAX(type) = 'Veg';

-- Q20: Min, Max, Avg order amount per user

SELECT t2.name,

MIN(amount) AS MinAmount,

MAX(amount) AS MaxAmount,

AVG(amount) AS AvgAmount

FROM orders t1

JOIN users t2 ON t1.user\_id = t2.user\_id

GROUP BY t1.user\_id, t2.name;