Zomato Case Study:

This case study outlines a simplified version of Zomato's database schema. The database consists of seven tables:

1. **users**
   Columns: user_id, name, email
   Summary: This table stores basic information about each user on the platform. It uniquely identifies every user by user_id.

2. **restaurants**
   Columns: r_id, r_name, cuisine
   Summary: This table provides details about restaurants, including a unique identifier (r_id), the restaurant's name, and its primary cuisine type.

3. **food**
   Columns: f_id, f_name, type (e.g., Veg or Non-Veg)
   Summary: This table lists all food items available on the platform. Each item has a unique ID (f_id) and is classified by type.

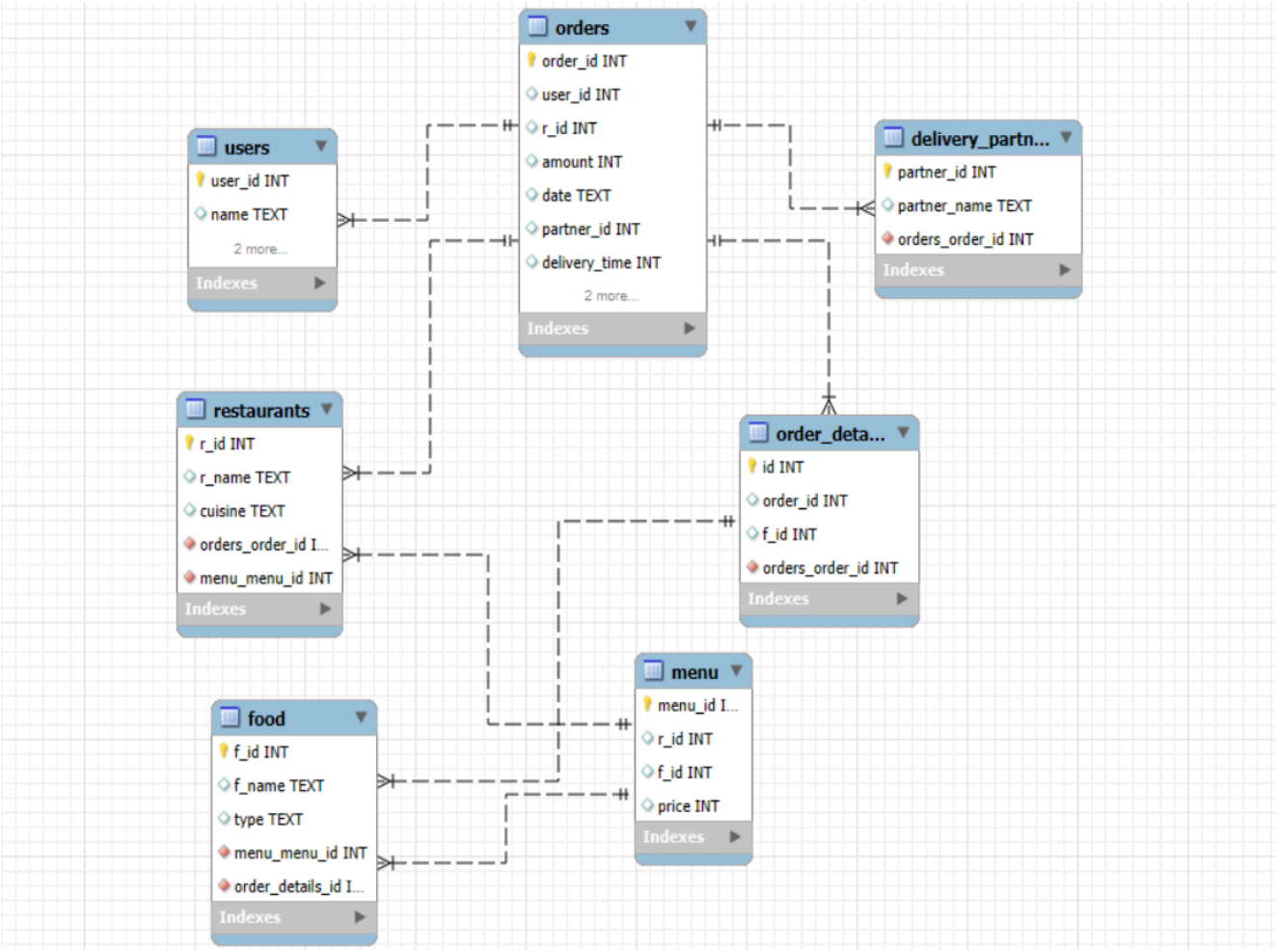
4. **menu**
   Columns: menu_id, r_id, f_id, price
   Summary: This table links restaurants to the food items they offer and specifies the price for each item at a given restaurant.

5. **orders**
   Columns: order_id, user_id, r_id, amount, date, partner_id, delivery_time, delivery_rating, restaurant_rating
   Summary: This table captures high-level information about each order, including which user placed the order, which restaurant it came from, the total amount, and ratings for delivery and the restaurant.

6. **delivery_partner**
   Columns: partner_id, partner_name
   Summary: This table stores information about delivery partners, including a unique ID (partner_id) and the partner's name.

7. **order_details**
   Columns: id, order_id, f_id
   Summary: This table provides a breakdown of the items in each order, linking an order_id to specific food items (f_id).

ER-Diagram:



QUERIES:

Q1) Find number of orders placed by each customer

➢ SELECT t2.name,COUNT(*) AS '#orders' FROM orders t1
JOIN users t2
ON t1.user_id = t2.user_id
GROUP BY t2.user_id

Q2) Find restaurants with most number of menu items

➢ SELECT r_name,COUNT(*) AS 'menu_items' FROM restaurants t1
JOIN menu t2
ON t1.r_id = t2.r_id
GROUP BY t2.r_id

Q3) Find number of votes and average rating for all the restaurants

- 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;

Q4) Find the food item that is being sold at most number of restaurants

- SELECT f_name,COUNT(*) FROM menu t1
  JOIN food t2
  ON t1.f_id = t2.f_id
  GROUP BY t1.f_id
  ORDER BY COUNT(*) DESC LIMIT 1;

Q5) Find restaurant with max revenue in a given month(ex:may)

- SELECT MONTHNAME(DATE(date)) AS month,date FROM orders
  SELECT r_name,SUM(amount) AS 'revenue' FROM orders t1
  JOIN restaurants t2
  ON t1.r_id = t2.r_id
  WHERE MONTHNAME(DATE(date)) = 'July'
  GROUP BY t1.r_id
  ORDER BY revenue DESC LIMIT 1;

Q6) Find month by month revenue for a particular restaurant (ex:kfc)

- SELECT MONTHNAME(DATE(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 MONTHNAME(DATE(date))
  ORDER BY MONTH(DATE(date));

Q7) Find restaurants with sales>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
  HAVING revenue > 1500;

Q8) Find customers who never ordered

```
SELECT user_id,name FROM users
EXCEPT
SELECT t1.user_id,name FROM orders t1
JOIN users t2
ON t1.user_id=t2.user_id;
```

Q9) Show order details of a particular customer in a given 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';
```

Q10) Find customer favourite food

```
WITH user_food_counts AS (
 SELECT
        o.user_id,
        od.f_id,
        COUNT(*) AS item_count
 FROM orders o
 JOIN order_details od
        ON o.order_id = od.order_id
 GROUP BY o.user_id, od.f_id
)
 SELECT
        user_id,f_id,
        MAX(item_count) AS max_count
 FROM user_food_counts
 GROUP BY user_id
```

Q11) Find most costly restaurants (i.e. avg price per dish)

```
SELECT r_name,SUM(price)/COUNT(*) AS 'Avg_price' FROM menu t1
JOIN restaurants t2
ON t1.r_id = t2.r_id
GROUP BY t1.r_id
ORDER BY Avg_price ASC LIMIT 1;
```

Q12) Find the delivery partner compensation using the formula(no.of deliveries*100+1000*avg_price)

➢
```
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
ORDER BY salary DESC;
```

Q13) Find all the Veg restaurants

```
SELECT r.r_name
FROM restaurants r
WHERE r.r_id NOT IN (
        SELECT m.r_id
        FROM menu m
        JOIN food f ON m.f_id = f.f_id
        WHERE f.type = 'Non-Veg'
);
```

Q14) Find the min and max amount for all the customer

➢
```
SELECT name,MIN(amount),MAX(amount),AVG(amount) FROM orders t1
JOIN users t2
ON t1.user_id = t2.user_id
GROUP BY t1.user_id
```

Q15) Identify the users who have spent the most money on orders.

➢
```
SELECT u.name, SUM(o.amount) AS total_spent
FROM orders o
JOIN users u ON o.user_id = u.user_id
GROUP BY u.user_id
ORDER BY total_spent DESC
LIMIT 5;
```