# DATABASE MANAGEMENT OF SWIGGY

## MySQL CASE STUDY

*By Onkar Sawant*

# INTRODUCTION

Story: Imagine you're creating a system like Swiggy, where people can order food from restaurants and get it delivered to their doorstep.

## Tables:

1. Users: These are the people using the app. They have names, email addresses, and passwords to log in.
2. Restaurants: These are the places where people can order food from. Each restaurant has a name and serves a particular type of cuisine.
3. Orders: Whenever someone places an order, it's recorded here. It keeps track of who ordered, from which restaurant, what was ordered, how much it cost, and when it was ordered.
4. Delivery Partners: These are the folks who pick up the food from the restaurant and deliver it to the customer. Each delivery partner has a name and an ID.
5. Order Details: This keeps track of the items in each order. For example, if someone orders a pizza and a drink, this table notes down those details.
6. Food Items: This is like a menu of all the food available. It has the names of items like "Non-veg Pizza" or "Choco Lava Cake" and specifies if they're vegetarian or non-vegetarian.
7. Menu: Each restaurant has a menu that lists all the food items they offer along with their prices.

## What Happens:

- Users log in and place orders from Restaurants.
- Each Order is assigned a Delivery Partner who delivers the food.
- The Order Details table records what exactly was ordered in each order.
- The Menu table links restaurants with their respective food items and prices.

## Why It Matters:

- This system helps manage orders efficiently, ensuring that the right food gets to the right people at the right time.
- It keeps track of what's available on the menu, making it easy for users to browse and order their favorite dishes.
- By recording feedback and ratings, it helps improve the overall experience for users, restaurants, and delivery partners alike.

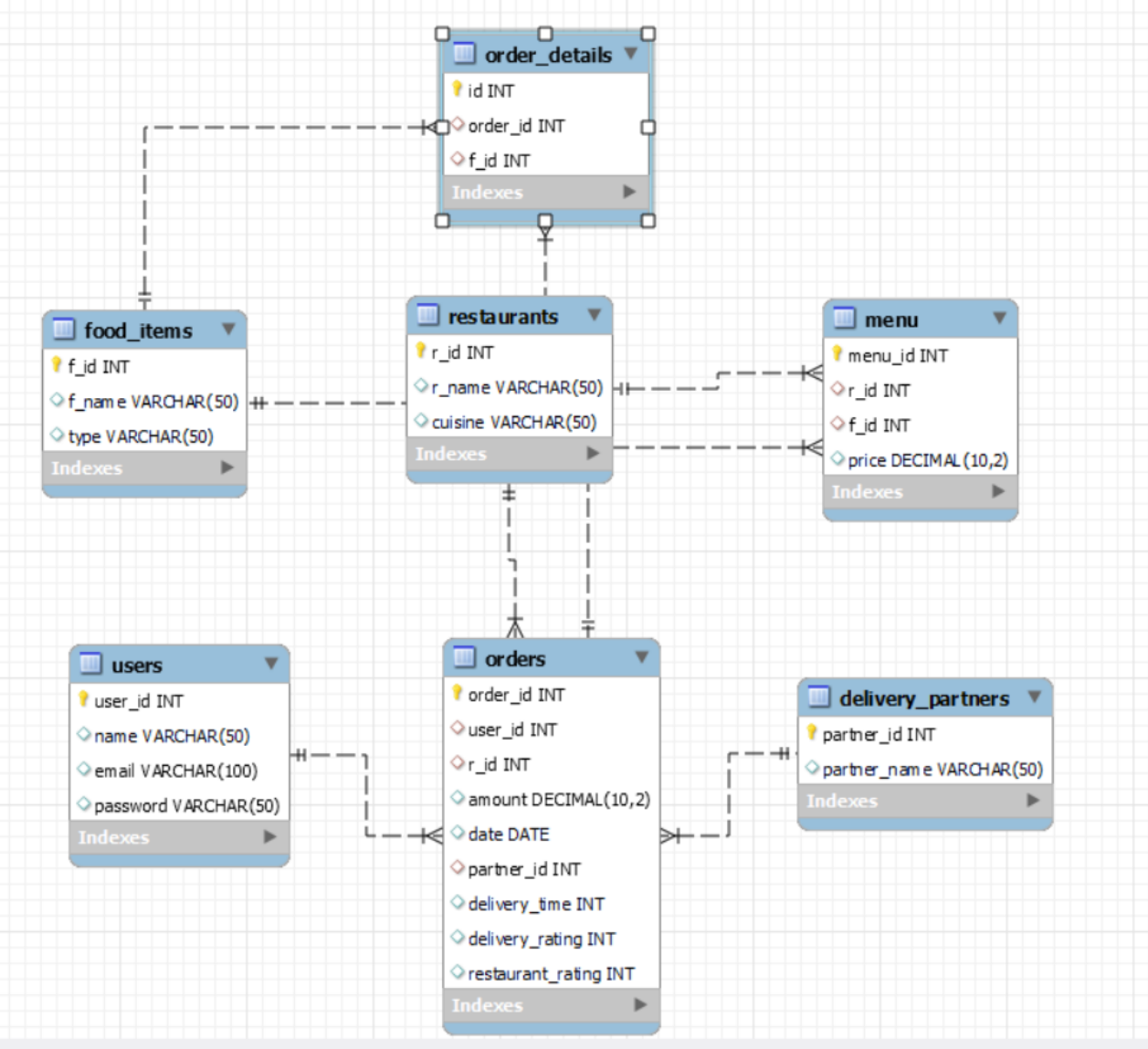
## Entities:

1. Users: Attributes include user_id (PK), name, email, and password.
2. Restaurants: Attributes include r_id (PK), r_name, and cuisine.
3. Orders: Attributes include order_id (PK), user_id (FK), r_id (FK), amount, date, partner_id (FK), delivery_time, delivery_rating, and restaurant_rating.
4. Delivery_Partners: Attributes include partner_id (PK) and partner_name.
5. Order_Details: Attributes include id (PK), order_id (FK), and f_id (FK).
6. Food_Items: Attributes include f_id (PK), f_name, and type.
7. Menu: Attributes include menu_id (PK), r_id (FK), f_id (FK), and price.

## Relationships:

1. User-Order: One-to-many relationship between Users and Orders based on user_id.
2. Restaurant-Order: One-to-many relationship between Restaurants and Orders based on r_id.
3. Order-Delivery_Partner: One-to-many relationship between Orders and Delivery_Partners based on partner_id.
4. Order-Order_Details: One-to-many relationship between Orders and Order_Details based on order_id.
5. Order_Detail-Food_Item: One-to-one relationship between Order_Details and Food_Items based on f_id.
6. Restaurant-Menu: One-to-many relationship between Restaurants and Menu based on r_id.
7. Food_Item-Menu: One-to-many relationship between Food_Items and Menu based on f_id

# E-R DIAGRAM FOR CASE STUDY



# COMMANDS

- create database swiggy;
- use swiggy;

CREATE TABLE users (

- user_id INT PRIMARY KEY,
- name VARCHAR(50),
- email VARCHAR(100),
- password VARCHAR(50)
- );

INSERT INTO users (user_id, name, email, password) VALUES

- (1, 'Nitish', 'nitish@gmail.com', 'p252h'),
- (2, 'Khushboo', 'khushboo@gmail.com', 'hxn9b'),
- (3, 'Vartika', 'vartika@gmail.com', '9hu7j'),
- (4, 'Ankit', 'ankit@gmail.com', 'lkko3'),
- (5, 'Neha', 'neha@gmail.com', '3i7qm'),
- (6, 'Anupama', 'anupama@gmail.com', '46rdw2'),
- (7, 'Rishabh', 'rishabh@gmail.com', '4sw123');

CREATE TABLE restaurants (

- r_id INT PRIMARY KEY,
- r_name VARCHAR(50),
- cuisine VARCHAR(50)
- );

INSERT INTO restaurants (r_id, r_name, cuisine) VALUES

- (1, 'dominos', 'Italian'),
- (2, 'kfc', 'American'),
- (3, 'box8', 'North Indian'),
- (4, 'Dosa Plaza', 'South Indian'),
- (5, 'China Town', 'Chinese');

CREATE TABLE orders (

- order_id INT PRIMARY KEY,
- user_id INT,
- r_id INT,
- foreign key(user_id) references users(user_id),
- foreign key(r_id) references restaurants(r_id),
- amount DECIMAL(10, 2),
- date DATE,
- partner_id INT,
- foreign key(partner_id) references delivery_partners(partner_id),
- delivery_time INT,
- delivery_rating INT,
- restaurant_rating INT
- );

INSERT INTO orders (order_id, user_id, r_id, amount, date, partner_id, delivery_time, delivery_rating, restaurant_rating)

- VALUES
- (1001, 1, 1, 550.00, '2022-05-10', 1, 25, 5, 3),
- (1002, 1, 2, 415.00, '2022-05-26', 1, 19, 5, 2),
- (1003, 1, 3, 240.00, '2022-06-15', 5, 29, 4, NULL),
- (1004, 1, 3, 240.00, '2022-06-29', 4, 42, 3, 5),
- (1005, 1, 3, 220.00, '2022-07-10', 1, 58, 1, 4),
- (1006, 2, 1, 950.00, '2022-06-10', 2, 16, 5, NULL),
- (1007, 2, 2, 530.00, '2022-06-23', 3, 60, 1, 5),
- (1008, 2, 3, 240.00, '2022-07-07', 5, 33, 4, 5),
- (1009, 2, 4, 300.00, '2022-07-17', 4, 41, 1, NULL),
- (1010, 2, 5, 650.00, '2022-07-31', 1, 67, 1, 4),
- (1011, 3, 1, 450.00, '2022-05-10', 2, 25, 3, 1),
- (1012, 3, 4, 180.00, '2022-05-20', 5, 33, 4, 1),
- (1013, 3, 2, 230.00, '2022-05-30', 4, 45, 3, NULL),
- (1014, 3, 2, 230.00, '2022-06-11', 2, 55, 1, 2),
- (1015, 3, 2, 230.00, '2022-06-22', 3, 21, 5, NULL),
- (1016, 4, 4, 300.00, '2022-05-15', 3, 31, 5, 5),
- (1017, 4, 4, 300.00, '2022-05-30', 1, 50, 1, NULL),
- (1018, 4, 4, 400.00, '2022-06-15', 2, 40, 3, 5),
- (1019, 4, 5, 400.00, '2022-06-30', 1, 70, 2, 4),
- (1020, 4, 5, 400.00, '2022-07-15', 3, 26, 5, 3),
- (1021, 5, 1, 550.00, '2022-07-01', 5, 22, 2, NULL),

- (1022, 5, 1, 550.00, '2022-07-08', 1, 34, 5, 1),
- (1023, 5, 2, 645.00, '2022-07-15', 4, 38, 5, 1),
- (1024, 5, 2, 645.00, '2022-07-21', 2, 58, 2, 1),
- (1025, 5, 2, 645.00, '2022-07-28', 2, 44, 4, NULL);

CREATE TABLE delivery_partners (

- partner_id INT PRIMARY KEY,
- partner_name VARCHAR(50)
- );

INSERT INTO delivery_partners (partner_id, partner_name)

VALUES

- (1, 'Suresh'),
- (2, 'Amit'),
- (3, 'Lokesh'),
- (4, 'Kartik'),
- (5, 'Gyandeep');

CREATE TABLE order_details (

- id INT PRIMARY KEY,
- order_id INT,
- f_id INT,
- FOREIGN KEY (order_id) REFERENCES orders(order_id),
- FOREIGN KEY (f_id) REFERENCES food_items(f_id)
- );

INSERT INTO order_details (id, order_id, f_id) VALUES

- (1, 1001, 1),
- (2, 1001, 3),
- (3, 1002, 4),
- (4, 1002, 3),
- (5, 1003, 6),
- (6, 1003, 3),
- (7, 1004, 6),
- (8, 1004, 3),
- (9, 1005, 7),
- (10, 1005, 3),
- (11, 1006, 1),

- (12, 1006, 2),
- (13, 1006, 3),
- (14, 1007, 4),
- (15, 1007, 3),
- (16, 1008, 6),
- (17, 1008, 3),
- (18, 1009, 8),
- (19, 1009, 9),
- (20, 1010, 10),
- (21, 1010, 11),
- (22, 1010, 6),
- (23, 1011, 1),
- (24, 1012, 8),
- (25, 1013, 4),
- (26, 1014, 4),
- (27, 1015, 4),
- (28, 1016, 8),
- (29, 1016, 9),
- (30, 1017, 8),
- (31, 1017, 9),
- (32, 1018, 10),
- (33, 1018, 11),
- (34, 1019, 10),
- (35, 1019, 11),
- (36, 1020, 10),
- (37, 1020, 11),
- (38, 1021, 1),
- (39, 1021, 3),
- (40, 1022, 1),
- (41, 1022, 3),
- (42, 1023, 3),
- (43, 1023, 4),
- (44, 1023, 5),
- (45, 1024, 3),
- (46, 1024, 4),
- (47, 1024, 5),
- (48, 1025, 3),

- (49, 1025, 4),
- (50, 1025, 5);

CREATE TABLE food_items (

- f_id INT PRIMARY KEY,
- f_name VARCHAR(50),
- type VARCHAR(50));

INSERT INTO food_items (f_id, f_name, type)

VALUES

- (1, 'Non-veg Pizza', 'Non-veg'),
- (2, 'Veg Pizza', 'Veg'),
- (3, 'Choco Lava cake', 'Veg'),
- (4, 'Chicken Wings', 'Non-veg'),
- (5, 'Chicken Popcorn', 'Non-veg'),
- (6, 'Rice Meal', 'Veg'),
- (7, 'Roti meal', 'Veg'),
- (8, 'Masala Dosa', 'Veg'),
- (9, 'Rava Idli', 'Veg'),
- (10, 'Schezwan Noodles', 'Veg'),
- (11, 'Veg Manchurian', 'Veg');

CREATE TABLE menu (

- menu_id INT PRIMARY KEY,
- r_id INT,
- f_id INT,
- price DECIMAL(10, 2),
- FOREIGN KEY (r_id) REFERENCES restaurants(r_id),
- FOREIGN KEY (f_id) REFERENCES food_items(f_id));

INSERT INTO menu (menu_id, r_id, f_id, price)

VALUES

- (1, 1, 1, 450.00),
- (2, 1, 2, 400.00),
- (3, 1, 3, 100.00),
- (4, 2, 3, 115.00),
- (5, 2, 4, 230.00),
- (6, 2, 5, 300.00),

- (7, 3, 3, 80.00),
- (8, 3, 6, 160.00),
- (9, 3, 7, 140.00),
- (10, 4, 6, 230.00),
- (11, 4, 8, 180.00),
- (12, 4, 9, 120.00),
- (13, 5, 6, 250.00),
- (14, 5, 10, 220.00),
- (15, 5, 11, 180.00);

select * from users;

select * from restaurants;

select * from orders;

select * from order_details;

select * from food_items;

select * from delivery_partners;

select * from menu;

-- 1) find customers that have never ordered?

- select user_id, name from users where user_id not in (select user_id from orders);

-- 2) average price of all the dishes?

- select food_items.f_id, food_items.f_name, avg(price) from menu inner join food_items on food_items.f_id = menu.f_id group by menu.f_id ;
- select f.f_id, f.f_name, avg(price) from menu as m inner join food_items as f on f.f_id = m.f_id group by m.f_id ;

-- 3) Find top restaurant in terms of no of orders for a given month?

- select r.r_name, r.r_id, count(r.r_id) as total_orders from orders as o inner join restaurants as r on r.r_id = o.r_id where date>='2022-07-01' and date<='2022-07-31' group by r_id order by total_orders desc limit 1;

-- 4) find restaurants with monthly sales > 500 rs

- select r.r_name, r.r_id, sum(amount) as monthly_income from orders as o inner join restaurants as r on r.r_id = o.r_id  where date>='2022-06-01' and date<='2022-06-30' group by r.r_id having sum(amount)>500 order by monthly_income desc;

-- 5) show all orders with order details for a particular customer in a particular date range

- select u.name, o.user_id, o.date, f.f_name, r.r_name, o.amount from orders as o inner join users as u on o.user_id = u.user_id  inner join order_details as od on o.order_id = od.order_id inner join food_items as f on od.f_id = f.f_id inner join restaurants as r on r.r_id = o.r_id where o.user_id <= 1 and date>='2022-05-01' and date<='2022-07-3