Group Number: 3
Group Member: Mingjun Yang, Xuanqi Li, Qing Hu, Yutong Yan

Database Design Document

Business Problems

Our team plans to design a database for **FOOD DELIVERY** platform, which is used for three kinds of users – customers, restaurant owners, and food delivery carries. Therefore, the database should implement the requirements, including storage, search, track and report varied data.

All Entities

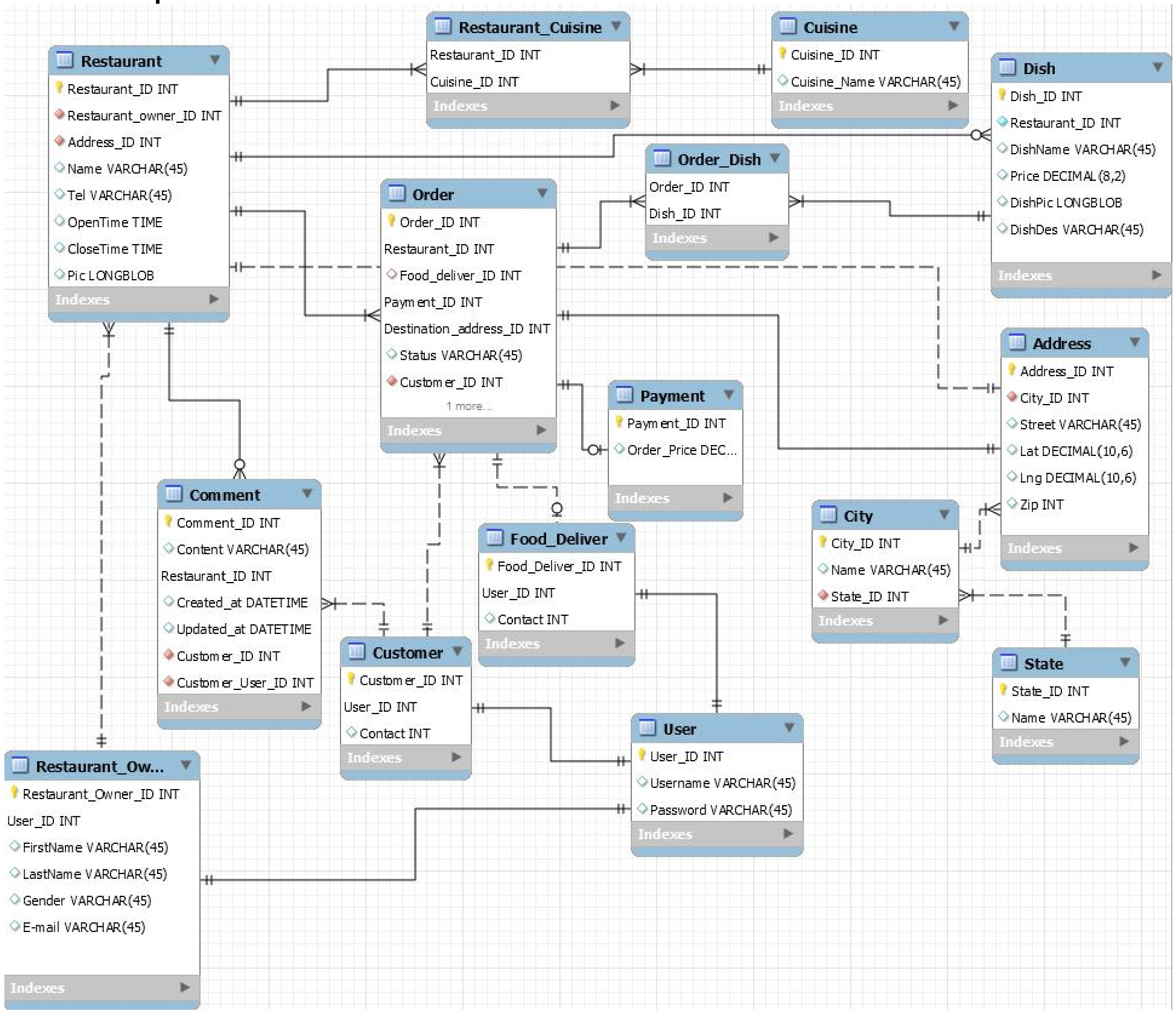
Users
Restaurant_Owner
Customer
Food_deliver

Restaurant
Cuisine
Restaurant_Cuisine
Dish
Comment

Order
Order_Dish
Payment

Address
City
State

Relationships:

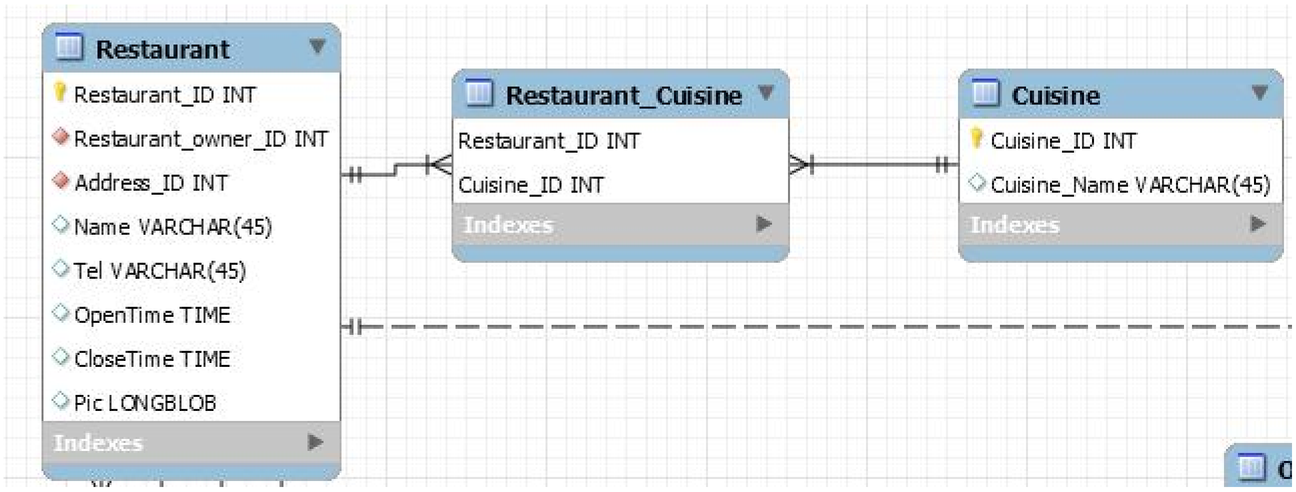


Key Design Decisions:
Our design decisions will address the following questions

- ✧ What entities and why these entities?
- ✧ What the PKs and FKs of each entity?
- ✧ What attributes contain?
- ✧ How the entity is related to others? And why?
- ✧ 1:1 or 1:n ?

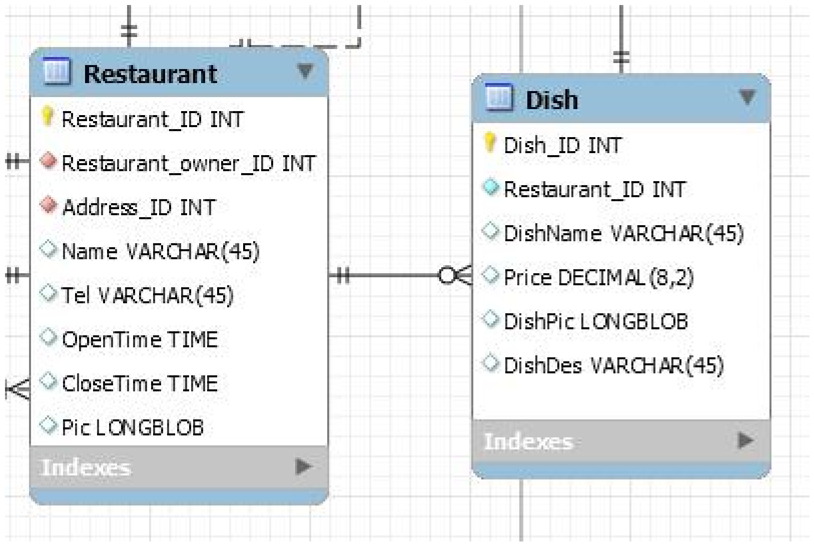
Decision Explanation:

1. **Restaurant & Restaurant-Cuisine & Cuisine**



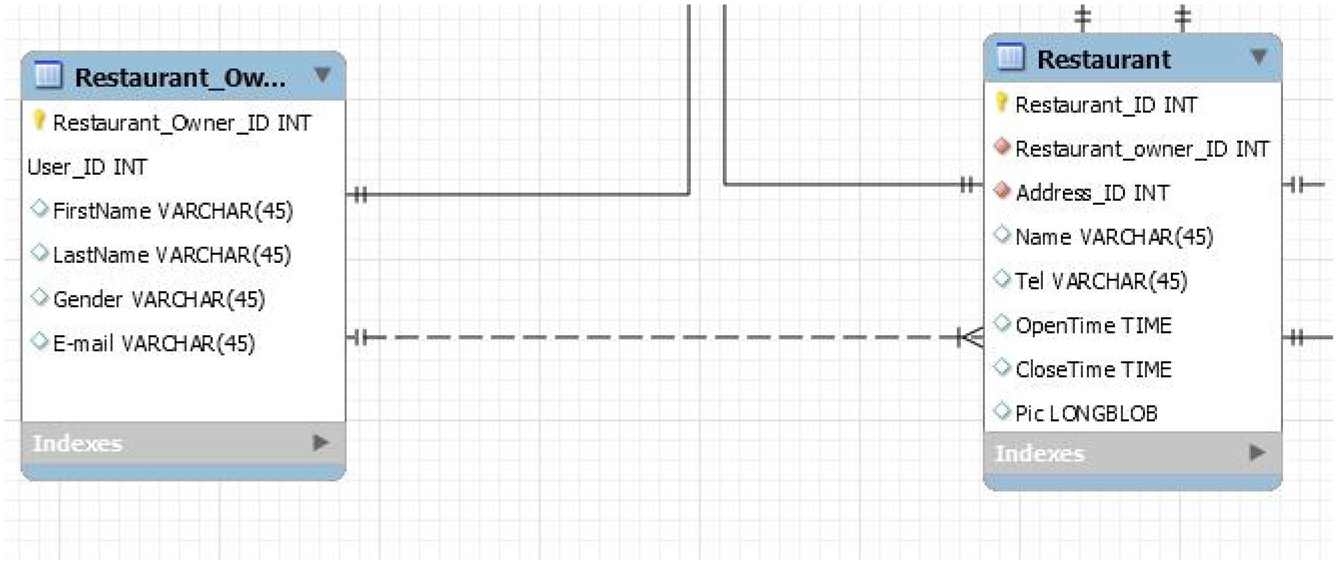
Explanation: One restaurant can be classified into several cuisines, such as Hot-pot is a kind of Asian Food, as well as Chinese Food. Additional, one cuisine contains several restaurants. In order to search a Restaurant depending on Cuisine, we design a table named Restaurant-Cuisine to present this many-to-many relationship. Moreover, Restaurant_ID and Cuisine_ID are both primary keys, therefore the three entities are in an identifying relationship.

2. **Restaurant & Dish**



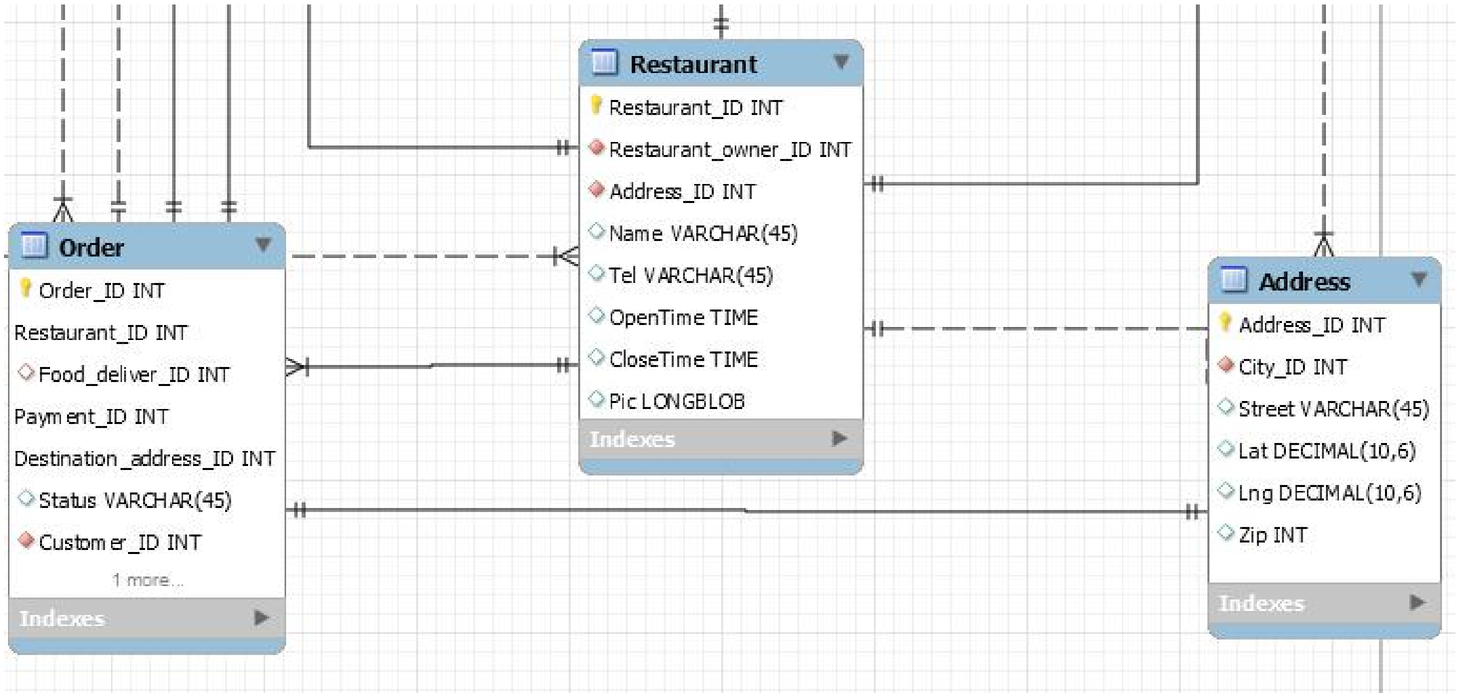
Explanation: This is a one-to-many relationship between Restaurant and Dish, because one restaurant could serve varied dishes.

3. Restaurant & Restaurant_owner



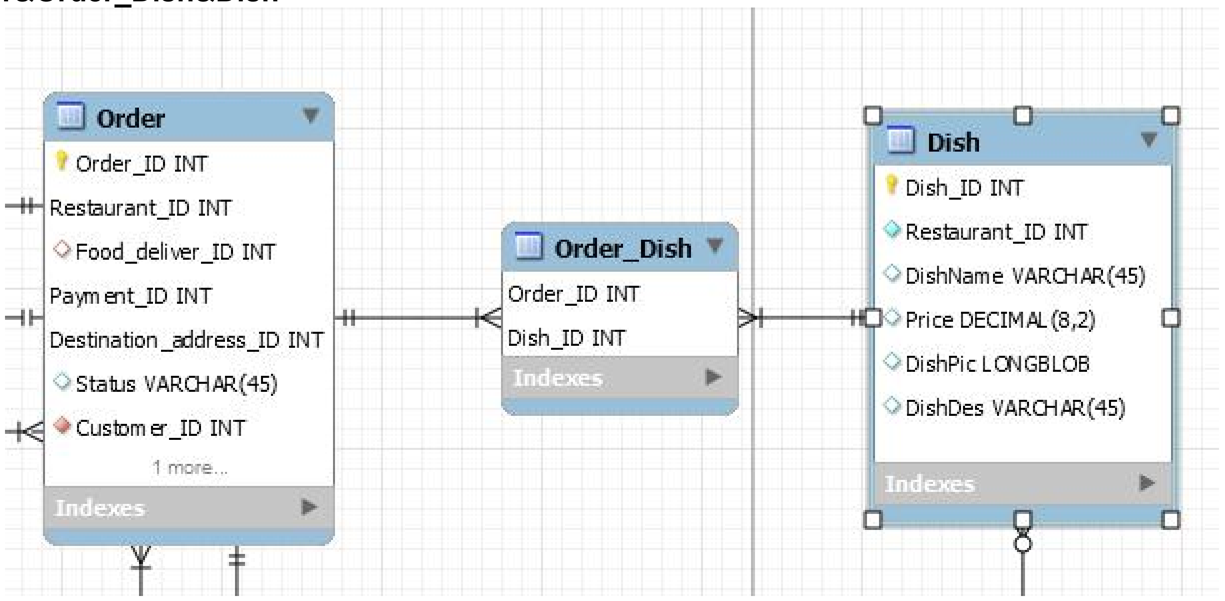
Explanation: This is a many-to-one relationship between Restaurant and Restaurant_owner, because one restaurant has one owner, however one owner may have several restaurants. In addition, there is no common primary key, so these two entities are non-identifying. In Restaurant entity, Restaurant_owner_ID is seemed as a foreign key.

4. Order&Restaurant&Address



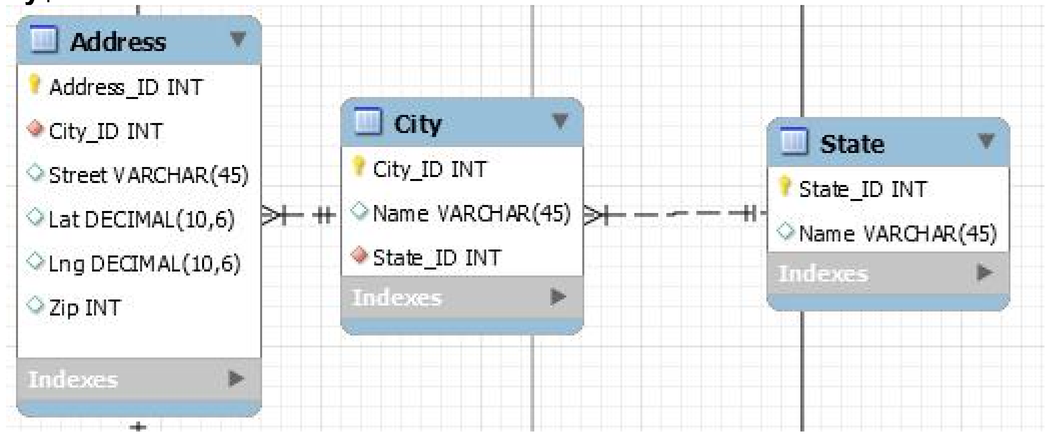
Explanation: One restaurant can have several orders. Every order will have only one address of a restaurant. When Restaurant_ID, Payment_ID, DestinationAddress_ID are primary keys.Restaurant&Order, Order&Address are identifying relationship. Restaurant&Address are non identifying relationships. Order: Restaurant_ID is FP.

5. Order&Order_Dish&Dish



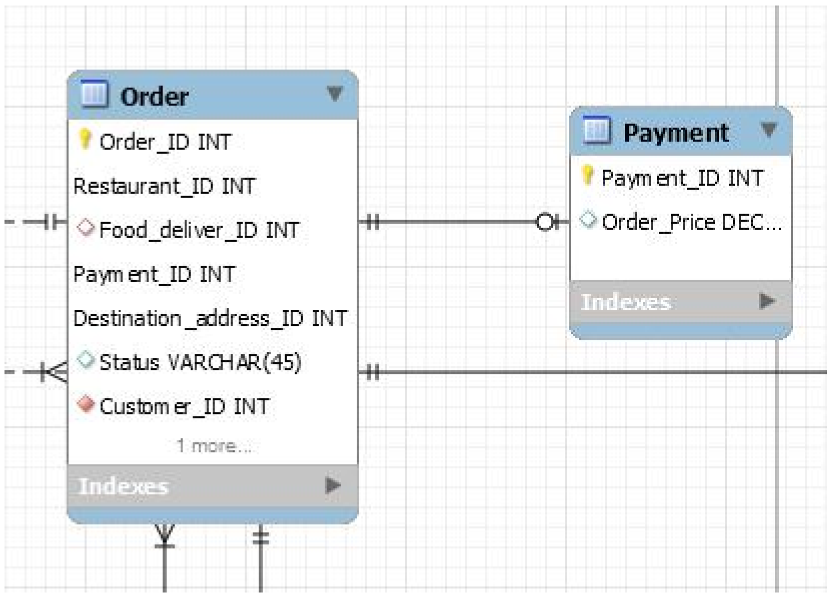
Explanation: This is a many-many relationship between Order and Dish. We define Dish as a specific type of food like noodles, burgers, etc. So one order can have many dishes and one dish can be ordered in many orders. Restaurant_ID, Payment_ID, DestinationAddress_ID are primary keys. There are identifying.

6. Address&City$State



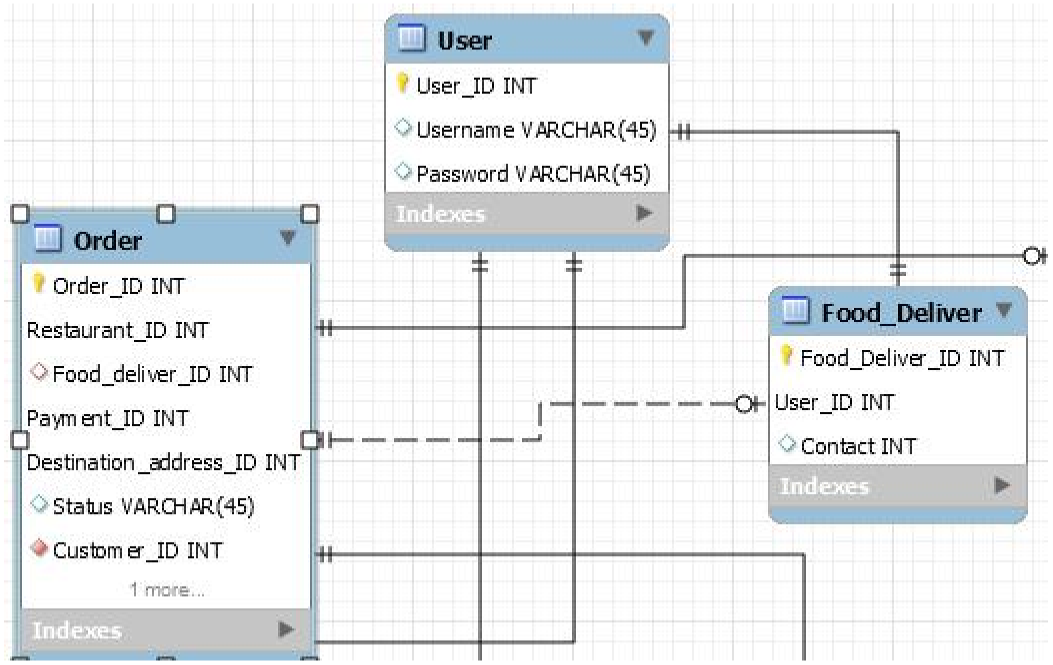
Explanation: In the US, there are many cities in one state and thousands of address in a city. So there is a many-one relationship between city and state and there is a many-one relationship between address and city. City: State_Id is FP. Address: City_ID is FP. no PK.

7. Order&Payment



Explanation: One order has only one payment so there is a one-one relationship. Payment: no FP.

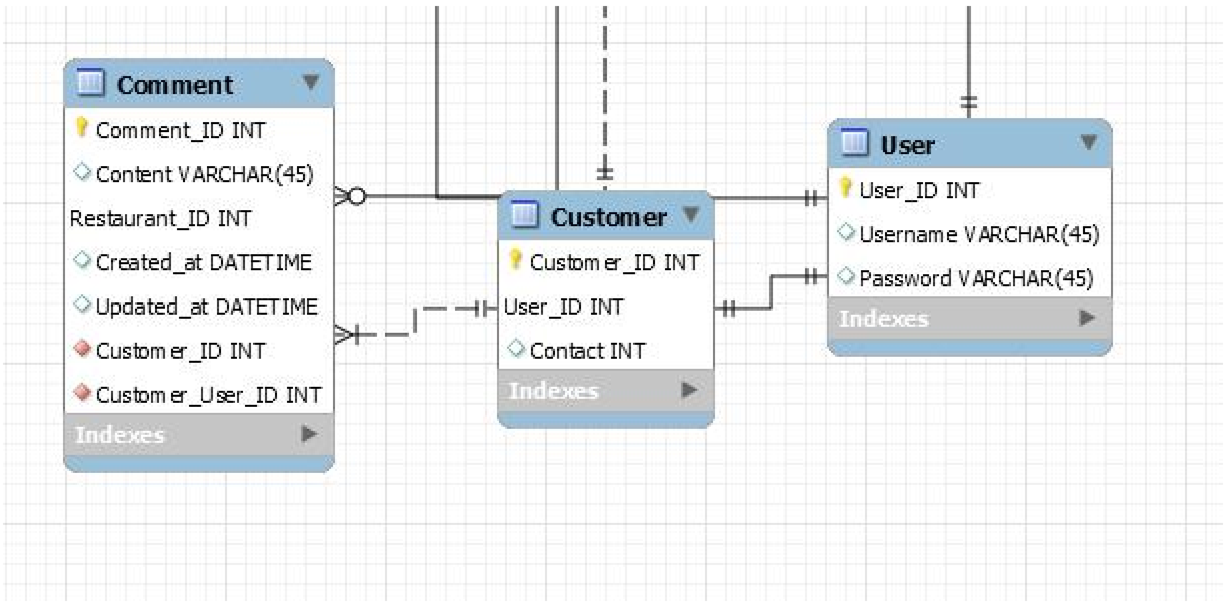
8. Order&Food_Deliever&User



Explanation: One man delivers one order, but when an order starts, there will be no food deliver so it is optional for Food_Deliver, and it is non-identifying.

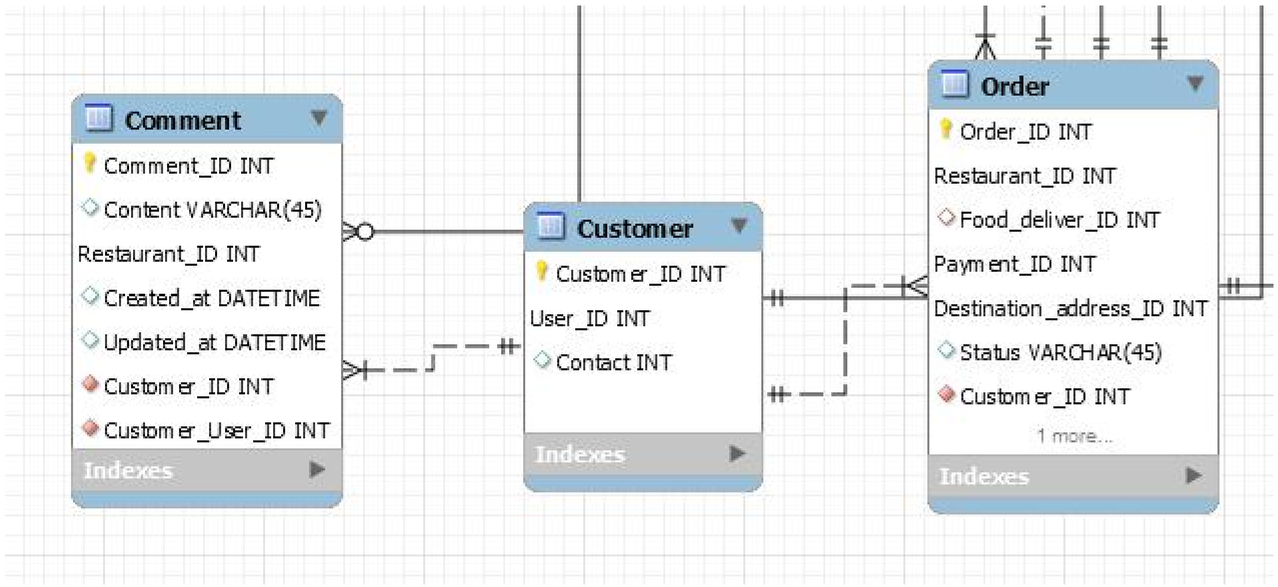
Food_Deliver: User_ID is FP. When deliveryman is confirmed, there is a one-one and identifying relationship between User and Food_Deliver.

9. Comment&Cusotomer&User



Explanation: A customer must be a User, so the relationship between User and Customer should be 1-to-1. A customer can make many comments, so there is a one-to-many relationship between comment and customer.
Customer: User_ID is FK.
Comment: Restaurant_ID is FK. The relationship between comment and customer is non identifying and the relationship between customer and user is identifying.

10. Comment&Cusotomer&Order



Explanation: A customer can have many order, and can have many comment, so there are one-to-many relationships between Comment and Customer, Customer and Order.