CSC461 Project 1

MileStone 2

Fangzhou Liu(#15), Subo Zhuang(#68), Ning Gu(#54), Nan Huang(#48)

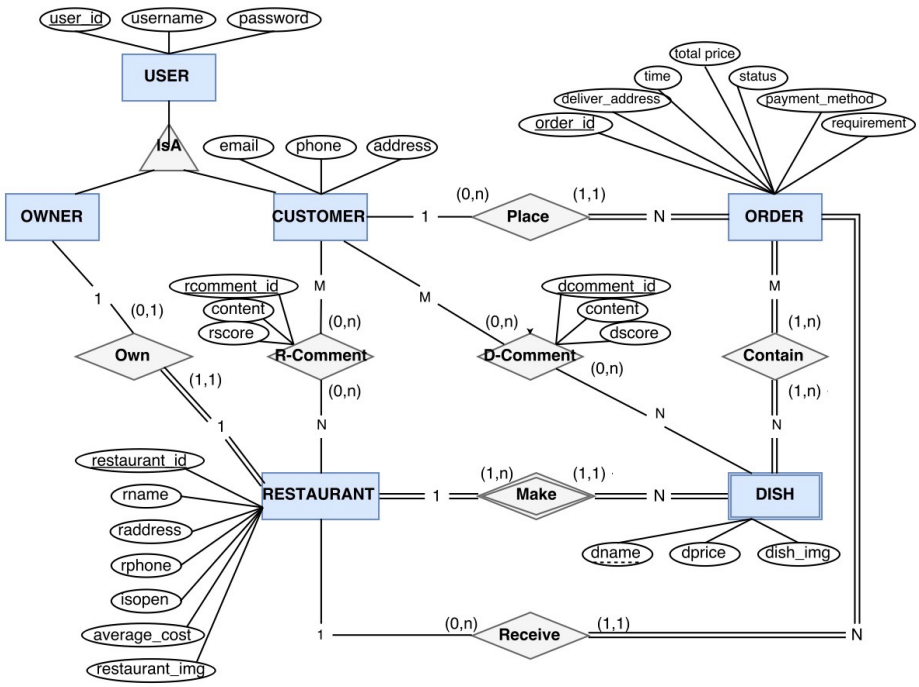
Project Detail

Project name: URObento
Team name: D3C
Team ID: 8
Team member:

Fangzhou Liu	(#15)
Subo Zhuang	(#68)
Ning Gu	(#54)
Nan Huang	(#48)

ER Diagram

ER Diagram



Assumption

- Each owner has one and only one restaurant, each restaurant has no branches.
- Different restaurant may share same dish names, but the price and image can be different.
- For each comment towards restaurant or dish, user can either give a detailed comment or a score. But user cannot post an empty comment(no comment and score)

List of Tables

In our database, we totally have 8 tables.

CUSTOMER

Customer(<u>user_id</u>, username, password, email, phone, address)
Foreign key: None

This table is derived from the entity Customer in the ER diagram with its attributes and its super class's (entity User in the ER diagram) attributes.
For the specialization customer and owner are user (the Is-A relationships in the ER diagram), we choose the option to build relations for the subclasses (customer and owner) only. This works because the subclasses are total i.e. every user is a customer or owner.

ATTRIBUTE	DATA TYPE	DEFAULT VALUE	IS NULL
<u>user_id</u>	INT	N/A	NO
username	VARCHAR(50)	N/A	NO
password	VARCHAR(20)	N/A	NO
email	VARCHAR(50)	NULL	YES
phone	VARCHAR(20)	N/A	NO
address	VARCHAR(50)	N/A	NO

- USER_ID: the id for each user. This attribute will be automatically assigned when user sign up in our system. Each user_id can uniquely and simply identifies an user. So, it work as the primary key.
- USERNAME: the username set by each user. This attribute, like all the system with login functions, used to authorize the users.
- PASSWORD: the password set by each user. This attribute, together with the password, used to authorize the users.

- EMAIL: the email for each customer. It used to receive some information, like coupons or discount notification given by both our system or restaurant owners.
- PHONE: the phone number for each customer. Owners use this information to contact the customer.
- ADDRESS: the permanent address for each user. It can be regarded as the default delivering address when customer ordered.

OWNER

Customer(<u>user_id</u>, username, password, email, restaurant_id)
Foreign Key: restaurant_id refer to Restaurant. Delete action: SET NULL.

Similarly to the table Customer, This table is derived from the entity Owner in the ER diagram with its attributes and its super class's (entity User in the ER diagram) attributes. The foreign key restaurant_id is derived from the relationship Own between owner and restaurant in the ER diagram.(foreign key approach)

ATTRIBUTE	DATA TYPE	DEFAULT VALUE	IS NULL
<u>user_id</u>	INT	N/A	NO
username	VARCHAR(50)	N/A	NO
password	VARCHAR(20)	N/A	NO
restaurant_id	INT	NULL	YES

- USER_ID: Same as USER_ID in the Customer table.
- USERNAME: Same as USERNAME in the Customer table.
- PASSWORD: Same as PASSWORD in the Customer table
- This attribute will be automatically assigned when user sign up in our system. Each owner_id can uniquely and simply identifies an user. So, it work as the primary key. Also, it referred to the USER_ID attribute in the USER table, representing its IS-A relationship. When delete a entry in this table, all restaurant belong to this owner will be deleted. Delete Cascade.
- RESTAURANT_ID: the id of the restaurant that this owner owns. Here we assume that each restaurant has an unique owner, even it is an branch restaurant. It's the foreign key referred to the restaurant_id in the restaurant table. When the primary key is deleted in the restaurant table, this value should be set to null i.e the owner no longer own any restaurant.

RESTAURANT

Restaurant(restaurant_id, rname, raddress, rphone, is_open, average_cost, restaurant_img)
Foreign key: None

This table is derived from the entity Restaurant in the ER diagram with its attributes.

ATTRIBUTE	DATA TYPE	DEFAULT VALUE	IS NULL
restaurant_id	INT	N/A	NO
rname	VARCHAR(30)	N/A	NO
raddress	VARCHAR(50)	N/A	NO
rphone	VARCHAR(20)	N/A	NO
is_open	BOOLEAN	N/A	NO
average_cost	VARCHAR(5)	N/A	YES
restaurant_img	VARCHAR(100)	DEFAULT IMAGE URL	NO

- RESTAURANT_ID: the id for each restaurant. This attribute will be automatically assigned when owner sign up in our system. Each restaurant_id can uniquely and simply identifies a restaurant. So, it work as the primary key. Also, it works as the foreign key, referred to the RESTAURANT_ID attribute in the DISH, ORDER, ORDER_DETAIL and COMMENT table. When delete an entry in this table, all dishes, orders and comments related to this restaurant will be deleted. Delete Cascade.
- RESTAURANT_NAME: the name of the restaurant.
- RADDRESS: the location of the restaurant.
- RPHONE: the phone number of the restaurant.
- IS_OPEN: the open status. User knows whether the restaurant they'd like to order is open or close through this information.
- AVE_COST: average cost for this restaurant. We use level number to indicate the average cost, higher score means higher cost.
- RIMG: an image url for the appearance or the decoration of the image. User can see the appearance or the inner decoration of the restaurant. If no image, we will set a default image to this restaurant. A nice image may attract users to order the dishes.

DISH

Dish(restaurant_id, dname, dprice, dish_img)
Foreign Key: restaurant_id refer to Restaurant. Delete action: CASCADE.

This table is derived from the weak entity DISH in the ER diagram with its attributes and the attribute restaurant_id (primary key) from is owner entity restaurant.

ATTRIBUTE	DATA TYPE	DEFAULT VALUE	IS NULL
restaurant_id	INT	N/A	NO
dname	VARCHAR(30)	N/A	NO
dprice	FLOAT	N/A	NO
dish_img	VARCHAR(100)	DEFAULT IMAGE URL	NO

- RESTAURANT_ID: From restaurant table sever as part of the primary key in this table. It is also a foreign key. When the restaurant_id have been deleted from the restaurant table, it should also be deleted form this table (on delete cascade) i.e none of the dishes in this table should have restaurant_id anymore.
- DNAME: the name of the dish. Thus the dishes can have the same name in different restaurant, we cannot use this attribute alone as the primary key. So, we combine the restaurant_id and the dname together to make the primary key for this table.
- DPRICE: the price of the dish.
- DISH_IMG: the image url of this dish. If no image, we will set a default image to this dish.

DISH COMMENT and RESTAURANT COMMENT

Dish_comment(comment_id, user_id, comment, score, restaurant_id, dname)
Foreign Key: user_id refer to Custom Delete action: NO ACTION; {restaurant_id, dname} refer to Dish. Delete action: NO ACTION.

This table is derived from the relationship comment between Customer and Dish in the ER diagram with its attributes and the attributes user_id (primary key Customer table) restaurant_id dname (primary key Dish table) as foreign keys.

Restaurant_comment(comment_id, user_id, comment, score, restaurant_id)
Foreign Key: user_id refer to Custom. Delete action: NO ACTION; restaurant_id refer to Restaurant. Delete action: NO ACTION.

This table is derived from the relationship comment between Customer and Restaurant in the ER diagram with its attributes and the attributes user_id (primary key Customer table) restaurant_id(primary key Restaurant table) as foreign keys.

ATTRIBUTE	DATA TYPE	DEFAULT VALUE	IS NULL
comment_id	INT	N/A	NO
user_id	INT	N/A	NO
comment	VARCHAR(100)	NULL	YES
score	INT	NULL	YES
restaurant_id/dname	INT/VARCHAR(30)	N/A	NO

- COMMENT_ID: the comment id. Each id indicates a unique comment for a dish or a restaurant. It work as the primary key in this table. When deleted an entry in this table, there should be no effect on table user, restaurant or dish. No Action.
- USER_ID: foreign key indicates the user given this comment. On delete, no action. The comment is kept.
- COMMENT: the content of the comment.
- SCORE: the overall score for this restaurant/dish. Customer can still give scores rather than word comments to each restaurant/dish. It also reflect the quality and popularity of the product.
- RESTAURANT_ID/DNAME: foreign key indicates the target of this comment. On delete, no action. The comment is kept.

ORDER

Order(order_id, user_id, deliver_address, time, total_price, status, paying_method)
Foreign Key: user_id refer to Customer. Delete action: NO ACTION; restaurant_id refer to Restaurant. Delete action: NO ACTION.

This table is derived from the entity Order in the ER diagram with its attributes.
The foreign key user_id is derived from the relationship Set between Customer and Oder in the ER diagram.
The foreign key restaurant_id is derived from the relationship Receive between Restaurant and Oder in the ER diagram. (Foreign key approach)

ATTRIBUTE	DATA TYPE	DEFAULT VALUE	IS NULL
order_id	INT	N/A	NO
user_id	INT	N/A	NO
restaurant_id	INT	N/A	NO
deliver_address	VARCHAR(50)	N/A	NO
time	TIMESTAMP	N/A	NO

total_price	FLOAT	N/A	NO
status	INT	N/A	NO
paying_method	INT	N/A	NO
requirement	VARCHAR(100)	NULL	YES

- ORDER_ID: a number uniquely represents each order. It works as the primary key in this table and the foreign key in the ORDER_DETAIL table. When delete the entry in this table, the entry with the same order_id in ORDER_DETAIL table should be deleted. Delete Cascade.
- USER_ID: foreign key refer to the CUSTOMER table. It represents the user who have this order. On delete, no action. History of his orders is kept when a customer is deleted.
- RESTAURANT_ID: foreign key refer to Restaurant. On delete, no action. History of all its orders is kept when a restaurant is deleted.
- DELIVER_ADDRESS: the address that the ordered dish to be delivered. The default address that user set when signing up can be used here.
- TIME: the time that this ordered generated.
- TOTAL_PRICE: the total price of this order.
- STATUS: a number indicates the current status of this order. Being processed, being delivered or finished.
- PAYING_METHODS: a number represents the payment of this order. It can be paid online, or paid with cash. Also, user can choose pay their bill before or after delivery.
- REQUIREMENT: special requirement for the order.

ORDER DISHES

Oder_dishes(order_id, restaurant_id, dname)
Foreign Key: order_id refer to Oder. Delete action: CASCADE; {restaurant_id, dname} refer to Dish. Delete action: NO ACTION.

This table is derived from the relationship Contain between Oder and Dish. The relationship is a M-N relationship, we need a new table. Primary key of the Oder and Dish are referred as foreign keys in this table.

ATTRIBUTE	DATA TYPE	DEFAULT VALUE	IS NULL
order_id	INT	N/A	NO
restaurant_id	INT	N/A	NO
dname	VARCHAR(30)	N/A	NO

- ORDER_ID, RESTAURANT_ID, DNAME: the composite primary key in this table. Thus a user can order multiple dishes within the same restaurant in one order, ORDER_ID, RESTAURANT_ID and DNAME cannot uniquely identify a entry. They are also foreign keys.

Relational Mapping Diagram