Database Team Project Report

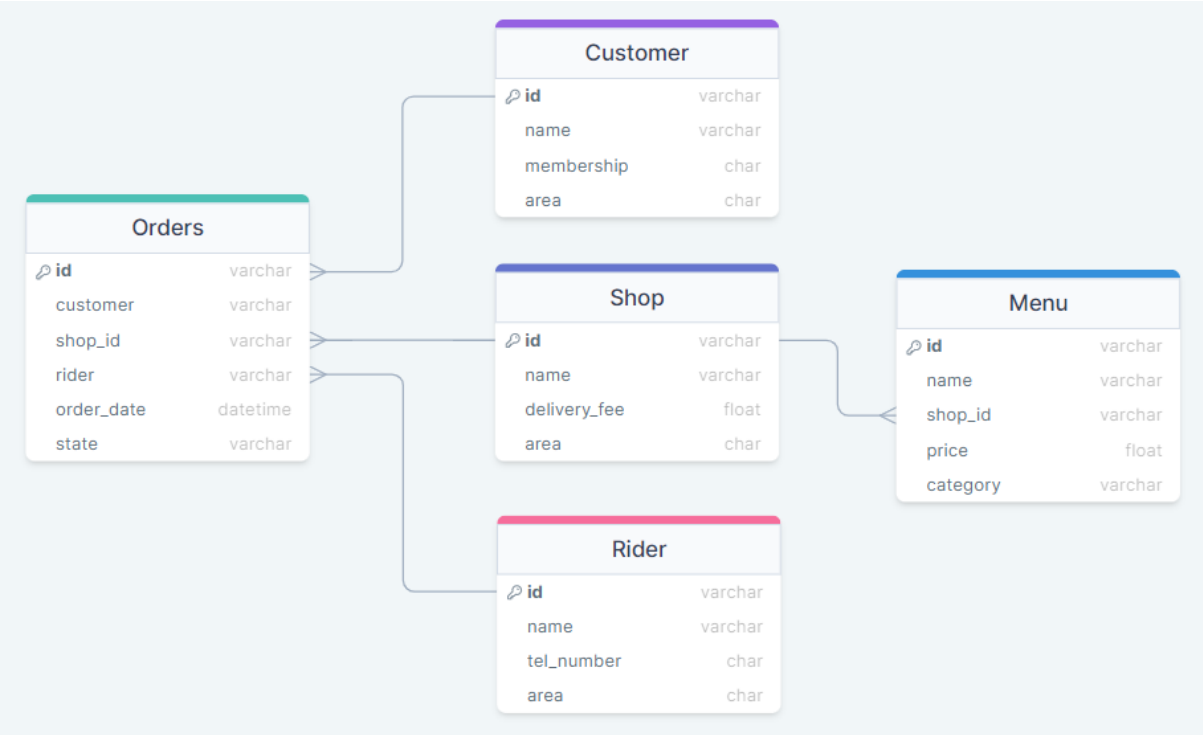
Team Shin (1976211 Shin Hyejoon, 2071074 Shin Bokyung)

Delivery Service

This JAVA application is an application for shop owners with MySQL DB using 5 tables. As a restaurant(shop) owner, you can manage your menus and delivery orders with 6 application menus.

Basically, orders are made by combining Shop, Customer, and Rider within the same area. The process from ordering to completion of delivery is as follows. A customer inserts an order into Orders table. After that, the shop accepts the order by changing the state to ‘Cooking’. There are two ways to determine which rider will deliver the order. The first way is to enter rider id to ‘Cooking’ state Order by himself / herself. Otherwise, shop owner has to find and contact a rider who can deliver the order within the same area. After the rider completes the delivery, he / she will update the order's state to ‘Complete’.

Database Schema



With the DB Schema we made, we can create applications for riders or applications for customers. But the 6 menus that our team made are for shop owners.

There are 5 tables. There are Shop, Rider, Customer, Menu, and Orders table. Each table has id as primary key. Among the five tables, Menu and Orders table has foreign key.

Customer table store customer id, customer's name, membership rank and residence area. Shop table store shop id, shop's name, delivery fee separate from the shop's menu price, and area of the store. Menu table store Menu id, name, id of the shop that sells the menu, price and food's category. Rider table store Rider's id, name, contact number, and delivery area where the rider can deliver.

We made 2 indices on Ordere table to find specific tuples faster. They are on shop_id and state attribute.

In addition, there are two views. One is Order_info which contains order_ id, shop_id and customer_id from Orders table to show which shop the customer ordered from. Another one is Not_in_delivery which contains the information of riders who are available(not in delivery).

Implementation

To classify our JAVA code in a large framework, it can be divided into database class and main class. In database class, we connect to jdbc and define six menus. In main class, we ask user to enter shop id before using 6 menus according to the application's intentions. After that, show 6 menus using `switch-case-default` . For each case, the method defined in the database class is used to execute the menu. Besides the 6 menus, we made "(0) Exit" to terminate the program. All user inputs are received by preparedstatement.

```
INSERT INTO Menu VALUES (?, ?, ?, ?, ?)
```

(1) Add Menu: This menu adds a new menu to the shop that matches the shop id initially entered. The list of menu currently registered in the shop will be shown. You can enter your input for the new menu one by one. If the registration of a new menu is completed without any problem, <Insertion Complete> will be printed. Else if an error like duplicate value for primary key occurs, insertion will not be inserted.

```
DELETE FROM Menu WHERE shop_id=? and id=?
```

(2) Delete Menu: It is for deletion of the menu that exists in your shop. The list of menu currently registered in the shop will be shown. You get a user input of menu id to delete. If the menu deletion is completed without any problem, <Deletion Complete> will be printed. Else if an error like deleting a value that does not exist, it cannot be deleted.

```
UPDATE Orders SET state=? WHERE id=?
```

(3) Change Order State: To update the state of an order, select menu 3. The list of the order in your shop will be printed out. Enter the id of the order to be updated. You can enter the desirable state among 'Cooking','In delivery' and 'Complete'. If the update is successfully done, <Order [order_number] is set to [state] > will be printed.

```
SELECT area, avg(delivery_fee) as average_fee FROM shop GROUP BY area
```

(4) Show Average delivery fee: Every shop has its own delivery fee. This menu shows the average delivery fee for each area. No user input required.

```
// this view is defined in 'createschema.sql'
CREATE VIEW Not_in_delivery as
SELECT DISTINCT rider.id as r_id,
  rider.name as r_name,
  rider.tel_number as r_tel,
  rider.area as r_area
FROM Rider, Orders
WHERE (Rider.id <> ALL(SELECT rider FROM Orders))
  OR (Rider.id = Orders.rider AND Orders.state="Complete");

SELECT DISTINCT r_id, r_name, r_tel
FROM Not_in_delivery INNER JOIN Shop ON Not_in_delivery.r_area = Shop.area
WHERE Shop.id = ?
```

(5) Find Rider: Among the riders in the same area, this menu will show you the id, name, and phone number of the riders who are not currently delivering. This menu shows riders in the same area who do not exist in the Order table, or riders who exist in the Orders table but the state is ‘Complete’. If the rider does not exist even after searching both cases, <No Rider Available> is printed.

```
// this view is defined in 'createschema.sql'
CREATE VIEW order_info as
SELECT Orders.id as O_id,shop_id as S_id, customer as C_id
FROM Orders;

SELECT customer.id,customer.name
FROM order_info JOIN customer ON customer.id= order_info.C_id
WHERE order_info.S_id=? and customer.membership=?
```

(6) Find Client By Membership Rank: Every customer has membership rank : S/A/B/C. The shop owner can find out the customers who have specific rank. Enter the rank you want to find. This menu finds within the clients who have ever ordered a menu in the shop at least once. The id and the name of the customers will be printed. If there is no customer, then <No Customer with membership [rank]> will be printed.

(0) Exit: This menu terminates the application.

Basic Setting

Initial tuples for each tables.

Rider

id	name	tel_number	area
R0808	S.Coups	010-1995-0808	S
R0610	Jun	010-1996-0610	S
R1107	The8	010-1997-1107	F
R0218	DK	010-1997-0218	W
R0116	S.k.	010-1996-0116	F

id	name	tel_number	area
R0211	Dino	010-1999-0211	W

Customer

id	name	rank	area
C1004	J.Han	S	F
C1230	Joshua	S	W
C0615	Hoshi	A	S
C0717	Wonwoo	A	S
C1122	Woozi	A	F
C0406	MinGyu	B	S
C0218	Vernon	C	W

Shop

id	name	delivery_fee	area
S0424	FML	3800	S
S0527	Face the Sun	4500	W
S0415	Darl+ing	2000	F
S1022	Attacca	2500	F
S0618	Your Choice	4000	S

Menu

id	name	shop_id	price	category
M1019	Sushi	S0425	8000	Japanese
M0622	Bulgogi	S0618	9000	Korean
M0916	Spaghetti	S0527	12000	Western
M0805	Jajangmyeon	S1022	5000	Chinese
M0121	Steak	S0415	19000	Western
M0716	Corn Soup	S0415	2000	Western
M0205	Udon	S0425	7500	Japanese
M1106	Bibimbap	S0527	9500	Korean
M2091	Risotto	S0415	11000	Western
M3226	Champon	S1022	6500	Chinese
M4961	Salad	S0527	3500	Western
M5004	Kimchi pancake	S0618	6500	Korean

Orders

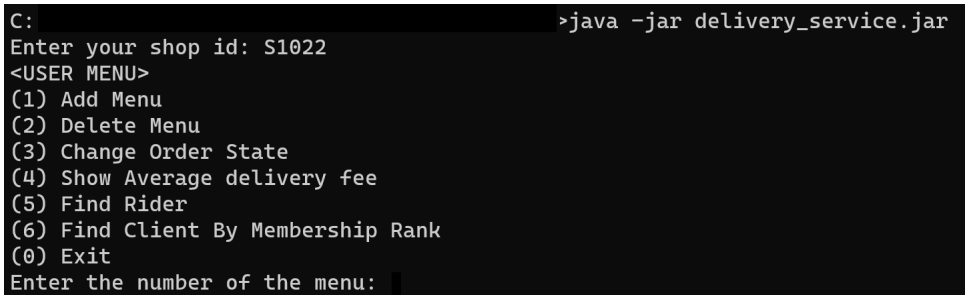
id	customer	shop_id	rider	order_date	state
O0522	C1004	S1022	R1107	2023-05-26 15:45:21	In delivery
O0704	C0717	S1022	R0610	2023-05-26 15:58:11	Cooking
O0425	C1230	S0527	R0218	2023-05-26 13:00:22	Cooking
O0910	C1122	S0415	R0116	2023-05-25 12:15:34	Complete
O0529	C0218	S0527	R0211	2023-05-26 12:43:06	In delivery
O1205	C0406	S0618	R0610	2023-04-20 11:12:02	Complete

- index1: “shop_id_index”
- index2: “state_index”

```
CREATE INDEX shop_id_index on Orders(shop_id);
CREATE INDEX state_index on Orders(state);
```

Result

Enter shop id



Menu 1: Add menu

Create new menu Malatang with the ID M1234.

```
mysql> select * from menu where shop_id ='S1022';
+-------+-------------+---------+-------+----------+
| id    | name        | shop_id | price | category |
+-------+-------------+---------+-------+----------+
| M0805 | Jajangmyeon | S1022   |  5000 | Chinese  |
| M1234 | Malatang    | S1022   |  7000 | Chinese  |
| M3226 | Champon     | S1022   |  6500 | Chinese  |
+-------+-------------+---------+-------+----------+
3 rows in set (0.00 sec)
```

The menu has been successfully created on the database.

Menu 2: Delete Menu

```
Enter the number of the menu: 2
M0805    Jajangmyeon     S1022   5000    Chinese
M1234    Malatang        S1022   7000    Chinese
M3226    Champon         S1022   6500    Chinese
Enter deletion target menu id: M1234
<Deletion Complete>
```

Delete Malatang from the menu by its ID M1234.

```
mysql> select * from menu where shop_id ='S1022';
+-------+-------------+---------+-------+----------+
| id    | name        | shop_id | price | category |
+-------+-------------+---------+-------+----------+
| M0805 | Jajangmyeon | S1022   |  5000 | Chinese  |
| M3226 | Champon     | S1022   |  6500 | Chinese  |
+-------+-------------+---------+-------+----------+
2 rows in set (0.00 sec)
```

The menu M1234 was successfully deleted from the database.

Menu 3: Change Order State

```
Enter the number of the menu: 3
<Order List>
O0522   C1004   R1107   2023-05-26 15:45:21     Cooking
O0704   C0717   R0610   2023-05-26 15:58:11     Cooking

Enter Order id: O0522
Enter state(Cooking/In delivery/Complete): In delivery
<Order O0522 is set to In delivery>
```

Change the state of the order O0522 to ‘In delivery’

```
mysql> select * from orders where shop_id ='S1022';
+-------+----------+---------+-------+---------------------+-------------+
| id    | customer | shop_id | rider | order_date          | state       |
+-------+----------+---------+-------+---------------------+-------------+
| O0522 | C1004    | S1022   | R1107 | 2023-05-26 15:45:21 | In delivery |
| O0704 | C0717    | S1022   | R0610 | 2023-05-26 15:58:11 | Cooking     |
+-------+----------+---------+-------+---------------------+-------------+
2 rows in set (0.00 sec)
```

Successfully updated on the database.

Menu 4: Show Average delivery fee

```
Enter the number of the menu: 4
<Average Delivery Fee For Each Area>
Area F: 2250
Area S: 3900
Area W: 4500
```

Print the average delivery fee for each area.

Menu 5: Find Rider

```
Enter the number of the menu: 5
<Current available riders in your area>
R0116    S.k.    010-1998-0116
```

Print the current available riders in the area

Menu 6: Find Client By Membership Rank

```
Enter the number of the menu: 6
Enter membership rank(S/A/B/C): S
<Customer with membership S>
Customer ID: C1004      Name: J.Han
```

Print customer with membership S, who have ever ordered a menu in the shop at least once.

Responsiblity

Shin Bokyung

- Create ER Diagram

- Organize and expand tables and attributes after the theme is decided
- Draft for 6 menus
- Programmed 3 menus on the application and their SQL query: menu 1, 2, and 5
- Write report and README

Shin Hyejoon

- Create the initial idea of the theme, delivery service
- Make all three SQL scripts (createschema.sql, initdata.sql, dropschema.sql)
- Program 3 menus on the application and their SQL query: menu 3, 4, and 6
- Program main java class and complied java code into the jar application
- Write report