Scenario Explained

E/15/202

Introduction

In here I am considering a food ordering app, something like Uber Eats. There a customer can order many foods only from one restaurant at a time. They cannot order from different restaurants at the same time. When the customer added all the food items they want from a particular restaurant they check their final bill (with all the discounts), if it is an affordable amount, they choose to order the food items. When the customer submits all those details then the food app should give it a new order number and all the details of the order should be added to the database. These details include the order number, customer name, delivery address, delivery boy's Id, ordered restaurant name, ordered food items. I will not check about how the delivery boy's ID is added to the database. In here I will consider only about the customer details.

When a customer submits all the details about the foods and the restaurant, ordering happens in several steps. During the first step it allows to cancel the order, but after that step the order cannot be cancelled. So in the first step, if a customer chooses to cancel the order, all the details added by the customer should be rollback. But after that first step all the details will be committed to the database.

In this type of app two users cannot access the same customer data, because it considered one customer account as one customer. But there may be instances where a customer gives their account details to another friend. And both of them end up trying to change their customer details at the same time unknowingly. Not only that but also, two or more concurrent orders from different users can be happen. In those situations we need to have a mechanism to handle the newly created order numbers.

Consider there are 3 tables as restaurants, order_details, customer_details

When a customer added the food items with the particular restaurant and submit the details,

1) Transaction starts
2) First the attribute orderNumber of order_details table will assign a unique number to that order (here the orderNumber is auto incremented).
3) Under that unique number order details of that particular customer will be added.
4) If the customer did not cancel the order during the first step, the details will be committed to the order_details table;
5) If the customer cancels the order during the first step, the details will be rollback. And the orderNumebr which was assigned during the transaction will not be used again.
6) When 2 or more concurrent users try to order from the food app, they will be able to add their details and commit smoothly. But the orderNumbers will be uniquely assigned for each and every order according to the time they commit.

#A.1) First concurrent user

```
mysql> set autocommit = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into order_details (CustomerID,restaurantID,foodItemCode,foodItemQuantity,totalPrice) values("A002","R001","F002",1,300);
Query OK, 1 row affected (0.10 sec)

mysql> select * from order_details;
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
| orderNumber | orderDateTime       | customerID | restaurantID | foodItemCode | foodItemQuantity | totalPrice |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
|           1 | 2020-05-20 18:45:18 | A001       | R001         | F001         |                1 |        200 |
|           4 | 2020-05-20 18:52:14 | A002       | R001         | F002         |                1 |        300 |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
2 rows in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.07 sec)

mysql> select * from order_details;
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
| orderNumber | orderDateTime       | customerID | restaurantID | foodItemCode | foodItemQuantity | totalPrice |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
|           1 | 2020-05-20 18:45:18 | A001       | R001         | F001         |                1 |        200 |
|           4 | 2020-05-20 18:52:14 | A002       | R001         | F002         |                1 |        300 |
|           5 | 2020-05-20 18:52:41 | A001       | R001         | F002         |                1 |        300 |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
```

#A.2) Second concurrent user

```
mysql> set autocommit = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into order_details (CustomerID,restaurantID,foodItemCode,foodItemQuantity,totalPrice) values("A001","R001","F002",1,300);
Query OK, 1 row affected (0.02 sec)

mysql> select * from order_details;
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
| orderNumber | orderDateTime       | customerID | restaurantID | foodItemCode | foodItemQuantity | totalPrice |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
|           1 | 2020-05-20 18:45:18 | A001       | R001         | F001         |                1 |        200 |
|           5 | 2020-05-20 18:52:41 | A001       | R001         | F002         |                1 |        300 |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
2 rows in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.06 sec)

mysql> select * from order_details;
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
| orderNumber | orderDateTime       | customerID | restaurantID | foodItemCode | foodItemQuantity | totalPrice |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
|           1 | 2020-05-20 18:45:18 | A001       | R001         | F001         |                1 |        200 |
|           4 | 2020-05-20 18:52:14 | A002       | R001         | F002         |                1 |        300 |
|           5 | 2020-05-20 18:52:41 | A001       | R001         | F002         |                1 |        300 |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
```

#B.1) First concurrent user

```
mysql> set autocommit = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into order_details (CustomerID,restaurantID,foodItemCode,foodItemQuantity,totalPrice) values("A002","R001","F002",1,300);
Query OK, 1 row affected (0.02 sec)

mysql> commit;
Query OK, 0 rows affected (0.06 sec)

mysql> select * from order_details;
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
| orderNumber | orderDateTime       | customerID | restaurantID | foodItemCode | foodItemQuantity | totalPrice |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
|           1 | 2020-05-20 18:45:18 | A001       | R001         | F001         |                1 |        200 |
|           4 | 2020-05-20 18:52:14 | A002       | R001         | F002         |                1 |        300 |
|           5 | 2020-05-20 18:52:41 | A001       | R001         | F002         |                1 |        300 |
|           6 | 2020-05-20 19:16:45 | A002       | R001         | F002         |                1 |        300 |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
4 rows in set (0.00 sec)

mysql> select * from order_details;
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
| orderNumber | orderDateTime       | customerID | restaurantID | foodItemCode | foodItemQuantity | totalPrice |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
|           1 | 2020-05-20 18:45:18 | A001       | R001         | F001         |                1 |        200 |
|           4 | 2020-05-20 18:52:14 | A002       | R001         | F002         |                1 |        300 |
|           5 | 2020-05-20 18:52:41 | A001       | R001         | F002         |                1 |        300 |
|           6 | 2020-05-20 19:16:45 | A002       | R001         | F002         |                1 |        300 |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
```

#B.2) second concurrent user

```
mysql> set autocommit = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into order_details (CustomerID,restaurantID,foodItemCode,foodItemQuantity,totalPrice) values("A001","R001","F002",1,300);
Query OK, 1 row affected (0.03 sec)

mysql> select * from order_details;
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
| orderNumber | orderDateTime       | customerID | restaurantID | foodItemCode | foodItemQuantity | totalPrice |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
|           1 | 2020-05-20 18:45:18 | A001       | R001         | F001         |                1 |        200 |
|           4 | 2020-05-20 18:52:14 | A002       | R001         | F002         |                1 |        300 |
|           5 | 2020-05-20 18:52:41 | A001       | R001         | F002         |                1 |        300 |
|           6 | 2020-05-20 19:16:45 | A002       | R001         | F002         |                1 |        300 |
|           7 | 2020-05-20 19:17:24 | A001       | R001         | F002         |                1 |        300 |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
5 rows in set (0.00 sec)

mysql> rollback;
Query OK, 0 rows affected (0.07 sec)

mysql> select * from order_details;
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
| orderNumber | orderDateTime       | customerID | restaurantID | foodItemCode | foodItemQuantity | totalPrice |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
|           1 | 2020-05-20 18:45:18 | A001       | R001         | F001         |                1 |        200 |
|           4 | 2020-05-20 18:52:14 | A002       | R001         | F002         |                1 |        300 |
|           5 | 2020-05-20 18:52:41 | A001       | R001         | F002         |                1 |        300 |
|           6 | 2020-05-20 19:16:45 | A002       | R001         | F002         |                1 |        300 |
+-------------+---------------------+------------+--------------+--------------+------------------+------------+
```

7) Assume a customer gives his/her account details for a friend. They are in 2 different places. Now both of them are trying to change the address at the same time.

#C.1) First concurrent session – same customer (A001)

```
mysql> set autocommit = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> update customer_details set customerAddress="Colombo" where customerID="A001";
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> commit;
Query OK, 0 rows affected (0.06 sec)

mysql> select customerAddress from customer_details where customerID="A001";
+-----------------+
| customerAddress |
+-----------------+
| Galle           |
+-----------------+
1 row in set (0.00 sec)
```

#C.2) Second concurrent session – same customer (A001)

```
mysql> set autocommit = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> update customer_details set customerAddress="Galle" where customerID="A001";
Query OK, 1 row affected (7.38 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> commit;
Query OK, 0 rows affected (0.07 sec)

mysql> select customerAddress from customer_details where customerID="A001";
+-----------------+
| customerAddress |
+-----------------+
| Galle           |
+-----------------+
1 row in set (0.00 sec)
```

As you can see only last change will be committed to the database. When both of them try to update the second session has to wait until the first session ended. If the second session has to wait too long it will end the waiting giving a timeout error.