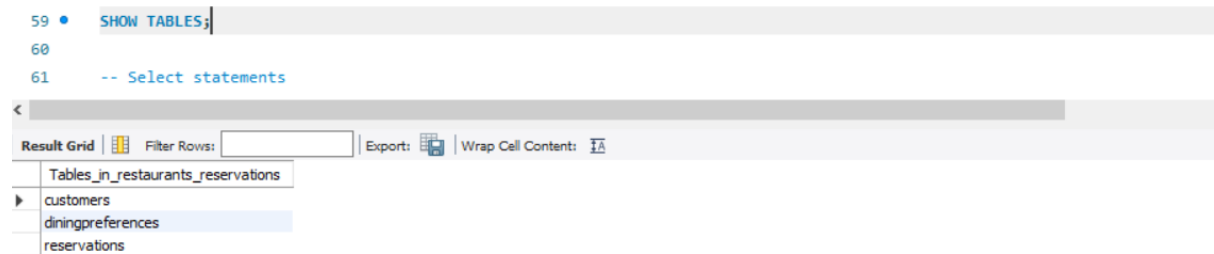


Name: Ibrahim Iddrisu

Date: 23 May 2024

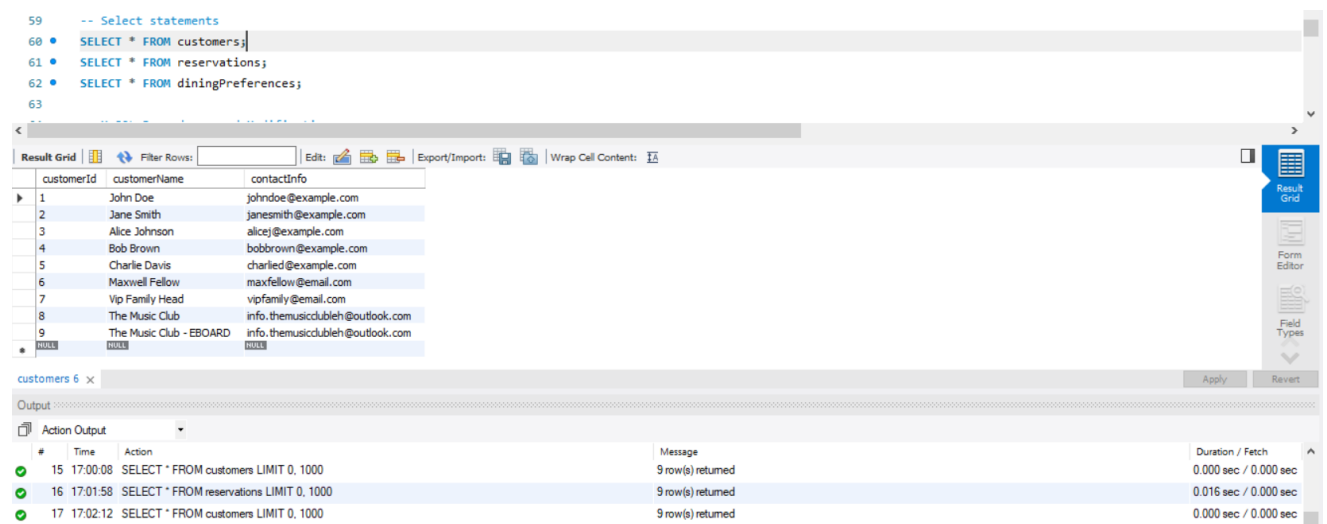
CIS 344 Final Project Report: Restaurant Reservation

MySQL



Beginning with the MySQL component of the project, I established the 'restaurant_reservations' database and constructed three tables named 'customers', 'diningpreferences', and 'reservations'. Each table was populated with essential attributes and corresponding foreign keys. Subsequently, I inserted test entries into the all the tables to verify its initialization and ensure its visibility, as demonstrated here.

SELECT * From 'Customers'



SELECT * From 'diningPreferences'

```

59 -- Select statements
60 • SELECT * FROM customers;
61 • SELECT * FROM reservations;
62 • SELECT * FROM diningPreferences;
63

```

preferenceId	customerId	favoriteTable	dietaryRestrictions
1	1	Table 5	None
2	2	Table 9	Gluten-free
3	3	Table 20	Vegetarian
4	4	Table 15	No dairy
5	5	Table 1	Vegan
NULL	NULL	NULL	NULL

diningPreferences 7 x

Output

#	Time	Action	Message	Duration / Fetch
16	17:01:58	SELECT * FROM reservations LIMIT 0, 1000	9 row(s) returned	0.016 sec / 0.000 sec
17	17:02:12	SELECT * FROM customers LIMIT 0, 1000	9 row(s) returned	0.000 sec / 0.000 sec
18	17:04:19	SELECT * FROM diningPreferences LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec

SELECT * From 'reservations'

```

59 -- Select statements
60 • SELECT * FROM customers;
61 • SELECT * FROM reservations;
62 • SELECT * FROM diningPreferences;
63

```

reservationId	customerId	reservationTime	numberOfGuests	specialRequests
1	1	2024-05-20 19:00:00	2	Window seat preferred
2	2	2024-05-21 20:00:00	4	Allergic to peanuts
3	3	2024-05-22 18:30:00	3	Celebrating anniversary
4	4	2024-05-23 19:45:00	5	Need high chair for a toddler
5	5	2024-05-24 20:30:00	2	Vegetarian meal options
6	6	2024-05-24 20:30:00	5	Vip table for 5 please.
7	7	2024-05-24 20:30:00	4	Vip table for 4 please. Alcohol for only 2.
8	8	2024-05-24 15:30:00	12	Drink refills every 30 mins.
9	9	2024-05-24 20:30:00	7	Drink refills every 30 mins. Vip Table Please.
NULL	NULL	NULL	NULL	NULL

reservations 8 x

Output

#	Time	Action	Message	Duration / Fetch
17	17:02:12	SELECT * FROM customers LIMIT 0, 1000	9 row(s) returned	0.000 sec / 0.000 sec
18	17:04:19	SELECT * FROM diningPreferences LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
19	17:06:58	SELECT * FROM reservations LIMIT 0, 1000	9 row(s) returned	0.000 sec / 0.000 sec

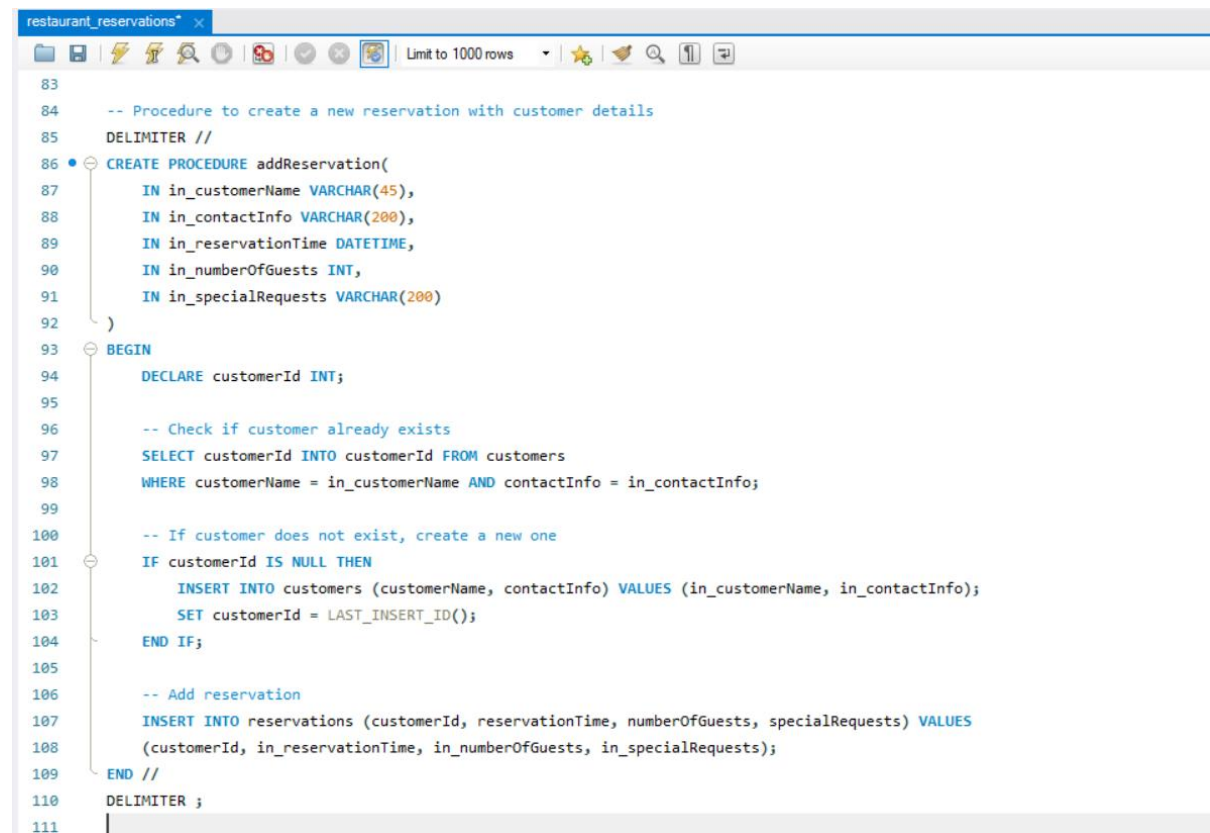
Next, I have created two stored procedures to 'find the reservation' and 'add a special customer request. findReservations procedure uses the existing customer id to find the reservation and addSpecialRequest procedure uses the existing reservation for the same.

```

67
68 -- Procedure to find all reservations for a customer using their ID
69 DELIMITER //
70 • CREATE PROCEDURE findReservations(IN in_customerId INT)
71 BEGIN
72     SELECT * FROM reservations WHERE customerId = in_customerId;
73 END //
74 DELIMITER ;
75
76 -- Procedure to update the specialRequests field in the reservations table
77 DELIMITER //
78 • CREATE PROCEDURE addSpecialRequest(IN in_reservationId INT, IN in_requests VARCHAR(200))
79 BEGIN
80     UPDATE reservations SET specialRequests = in_requests WHERE reservationId = in_reservationId;
81 END //
82 DELIMITER ;
83

```

The next stored procedure is addReservation procedure. This procedure checks if the customer id exists or not. If the customer id is not present, the new customer id created and then the reservation is made.

A screenshot of a SQL IDE window titled 'restaurant_reservations'. The window has a toolbar at the top with icons for file operations, execution, and search. Below the toolbar, the SQL code for the 'addReservation' stored procedure is displayed. The code is as follows:

```
83
84  -- Procedure to create a new reservation with customer details
85  DELIMITER //
86  CREATE PROCEDURE addReservation(
87      IN in_customerName VARCHAR(45),
88      IN in_contactInfo VARCHAR(200),
89      IN in_reservationTime DATETIME,
90      IN in_numberOfGuests INT,
91      IN in_specialRequests VARCHAR(200)
92  )
93  BEGIN
94      DECLARE customerId INT;
95
96      -- Check if customer already exists
97      SELECT customerId INTO customerId FROM customers
98      WHERE customerName = in_customerName AND contactInfo = in_contactInfo;
99
100     -- If customer does not exist, create a new one
101     IF customerId IS NULL THEN
102         INSERT INTO customers (customerName, contactInfo) VALUES (in_customerName, in_contactInfo);
103         SET customerId = LAST_INSERT_ID();
104     END IF;
105
106     -- Add reservation
107     INSERT INTO reservations (customerId, reservationTime, numberOfGuests, specialRequests) VALUES
108     (customerId, in_reservationTime, in_numberOfGuests, in_specialRequests);
109 END //
110 DELIMITER ;
111
```

Python: I started off by modifying the restaurantDatabase.py file with the appropriate MySQL Credentials:

```
import mysql.connector
from mysql.connector import Error

class RestaurantDatabase:
    def __init__(self,
                  host="localhost",
                  port="3306",
                  database="restaurants_reservations",
                  user='root',
                  password=''):
```

To first communicate with MySQL, I used the same port number database that's in MySQL so python can retrieve the information.

Next, I added methods in python to do various tasks such as adding and viewing reservations, Adding customer. Each method in this portion of the python code is similar except for changing the “query” portion to each method's respective statements that correspond to the statements in MySQL.

```
def addReservation(self, customer_name, contact_info, reservation_time, number_of_guests, special_requests):
    ''' Method to insert a new reservation into the reservations table '''
    if self.connection.is_connected():
        self.cursor = self.connection.cursor()

        # Check if customer already exists
        self.cursor.execute("SELECT customerId FROM customers WHERE customerName = %s AND contactInfo = %s", (customer_name, contact_info))
        result = self.cursor.fetchone()

        if result:
            customer_id = result[0]
        else:
            # Add new customer
            self.cursor.execute("INSERT INTO customers (customerName, contactInfo) VALUES (%s, %s)", (customer_name, contact_info))
            self.connection.commit()
            customer_id = self.cursor.lastrowid

        # Add reservation
        query = "INSERT INTO reservations (customerId, reservationTime, numberOfGuests, specialRequests) VALUES (%s, %s, %s, %s)"
        self.cursor.execute(query, (customer_id, reservation_time, number_of_guests, special_requests))
        self.connection.commit()
        print("Reservation added successfully")

def getAllReservations(self):
    ''' Method to get all reservations from the reservations table '''
    if self.connection.is_connected():
        self.cursor = self.connection.cursor()
        query = """
        SELECT r.reservationId, c.customerName, c.contactInfo, r.reservationTime, r.numberOfGuests, r.specialRequests
        FROM reservations r
        JOIN customers c ON r.customerId = c.customerId
        """
        self.cursor.execute(query)
        records = self.cursor.fetchall()
        return records

def deleteReservation(self, reservation_id):
    ''' Method to delete a reservation from the reservations table '''
    if self.connection.is_connected():
        self.cursor = self.connection.cursor()
        query = "DELETE FROM reservations WHERE reservationId = %s"
        self.cursor.execute(query, (reservation_id,))
        self.connection.commit()
        print("Reservation deleted successfully")

if __name__ == "__main__":
    db = RestaurantDatabase()
    # Calls the reservations in the database to view in the View Reservations page.
    reservations = db.getAllReservations()
    for reservation in reservations:
        print(reservation)
```

Some of the additional functions I implemented are getCustomerPreference, addSpecialRequest, findReservation, and deleteReservation.

Now in the restuarantServer.py file

Mainly I called the methods from the restaurantDatabase.py and made sure it was visible on the local host web page. For adding reservation, in do_POST I initialize the variables from the reservation table and then call the method to add a new reservation from the database file.

```
def do_POST(self):
    try:
        if self.path == '/addReservation':
            form = cgi.FieldStorage(
                fp=self.rfile,
                headers=self.headers,
                environ={'REQUEST_METHOD': 'POST'})

            customer_name = form.getvalue("customer_name")
            contact_info = form.getvalue("contact_info")
            reservation_time = form.getvalue("reservation_time")
            number_of_guests = form.getvalue("number_of_guests")
            special_requests = form.getvalue("special_requests")

            # Validate input
            if not customer_name or not contact_info or not reservation_time or not number_of_guests.isdigit():
                self.send_error(400, "Invalid input data")
                return

            number_of_guests = int(number_of_guests)

            # Call the Database Method to add a new reservation
            self.database.addReservation(customer_name, contact_info, reservation_time, number_of_guests, special_requests)
            print("Reservation added for customer:", customer_name)

            # Respond with a confirmation message
            self.send_response(200)
            self.send_header('Content-type', 'text/html')
            self.end_headers()

            self.wfile.write(b"<html><head><title>Add Reservation</title></head>")
            self.wfile.write(b"<body>")
            self.wfile.write(b"<center><h2>Reservation Added</h2>")
            self.wfile.write(b"<p><p>Reservation has been successfully added.</p>")
            self.wfile.write(b"<a href='/addReservation'>Add Another Reservation</a><br>")
            self.wfile.write(b"<a href='/viewReservations'>View Reservations</a></center>")
            self.wfile.write(b"</body></html>")
```

To create the web page itself, I added the necessary fields to input the information shown here.

```
self.wfile.write(b"<html><head><title>Restaurant Portal</title></head>")
self.wfile.write(b"<body>")
self.wfile.write(b"<center><h1>Restaurant Portal</h1>")
self.wfile.write(b"<hr>")
self.wfile.write(b"<div> <a href='/'>Home</a>| \
    <a href='/addReservation'>Add Reservation</a>|\
    <a href='/deleteReservation'>Delete Reservation</a>|\
    <a href='/viewReservations'>View Reservations</a></div>")
self.wfile.write(b"<hr><h2>All Reservations</h2>")
self.wfile.write(b"<table border=2> \
    <tr><th> Reservation ID </th>\
    <th> Customer Name </th>\
    <th> Contact Info </th>\
    <th> Reservation Time </th>\
    <th> Number of Guests </th>\
    <th> Special Requests </th></tr>")
records = self.database.getAllReservations()
for row in records:
    self.wfile.write(b' <tr> <td>')
    self.wfile.write(str(row[0]).encode())
    self.wfile.write(b'</td><td>')
    self.wfile.write(str(row[1]).encode())
    self.wfile.write(b'</td><td>')
    self.wfile.write(str(row[2]).encode())
    self.wfile.write(b'</td><td>')
    self.wfile.write(str(row[3]).encode())
    self.wfile.write(b'</td><td>')
    self.wfile.write(str(row[4]).encode())
    self.wfile.write(b'</td><td>')
    self.wfile.write(str(row[5]).encode())
    self.wfile.write(b'</td></tr>')

self.wfile.write(b"</table></center>")
self.wfile.write(b"</body></html>")
return
```

Similarly, I have called other functions as well in the server file as per the requirements of the project.

Other things I added were some validation inputs to make sure that a user actually inserts data into the forms when adding a reservation and deleting a reservation.

If they do not enter information they are prompted to do it.

I also made sure to place the menu navigation links on all the pages throughout the website to make the view uniform and pleasant.

I also added some comments to guide the user on how to type in their reservation.

See Screenshots of the website in full function below.

View Reservations

[Home](#) [Add Reservation](#) [Delete Reservation](#) [View Reservations](#)

All Reservations

Reservation ID	Customer Name	Contact Info	Reservation Time	Number of Guests	Special Requests
1	John Doe	johndoe@example.com	2024-05-20 19:00:00	2	Window seat preferred
2	Jane Smith	jan smith@example.com	2024-05-21 20:00:00	4	Allergic to peanuts
3	Alice Johnson	alicej@example.com	2024-05-22 18:30:00	3	Celebrating anniversary
4	Bob Brown	bobbrown@example.com	2024-05-23 19:45:00	5	Need high chair for a toddler
5	Charlie Davis	charlied@example.com	2024-05-24 20:30:00	2	Vegetarian meal options
6	Maxwell Fellow	maxfellow@email.com	2024-05-24 20:30:00	5	Vip table for 5 please.
7	Vip Family Head	vipfamily@email.com	2024-05-24 20:30:00	4	Vip table for 4 please. Alcohol for only 2.
8	The Music Club	info.themusicclubleh@outlook.com	2024-05-24 15:30:00	12	Drink refills every 30 mins.
9	The Music Club - EBOARD	info.themusicclubleh@outlook.com	2024-05-24 20:30:00	7	Drink refills every 30 mins. Vip Table Please.
10	The Music Club - EBOARD	info.themusicclubleh@outlook.com	2024-05-24 20:30:00	7	Drink refills every 30 mins. Vip Table Please.

ADD RESERVATIONS

Add Reservations

[Home](#) [Add Reservation](#) [Delete Reservation](#) [View Reservations](#)

Add Reservation

Customer Name: (Name of person booking the reservation?)

The Music Club - EBOARD

Contact Info: (An email or mobile number is fine.)

info.themusicclubleh@outlo

Reservation Time: (Time format: YYYY-MM-DD HH:MM:SS)

2024-05-24 20:30:00

Number of Guests: (How many people are coming?)

7

Special Requests: (Any special request? An allergy maybe?)

Drink refills every 30 mins. \

Submit

Reservation Added

Reservation has been successfully added.

[Add Another Reservation](#)
[View Reservations](#)

VIEW TO SEE IF RESERVATION ADDED

View Reservations

[Home](#) [Add Reservation](#) [Delete Reservation](#) [View Reservations](#)

All Reservations

Reservation ID	Customer Name	Contact Info	Reservation Time	Number of Guests	Special Requests
1	John Doe	johndoe@example.com	2024-05-20 19:00:00	2	Window seat preferred
2	Jane Smith	jan smith@example.com	2024-05-21 20:00:00	4	Allergic to peanuts
3	Alice Johnson	alicej@example.com	2024-05-22 18:30:00	3	Celebrating anniversary
4	Bob Brown	bobbrown@example.com	2024-05-23 19:45:00	5	Need high chair for a toddler
5	Charlie Davis	charlied@example.com	2024-05-24 20:30:00	2	Vegetarian meal options
6	Maxwell Fellow	maxfellow@email.com	2024-05-24 20:30:00	5	Vip table for 5 please.
7	Vip Family Head	vipfamily@email.com	2024-05-24 20:30:00	4	Vip table for 4 please. Alcohol for only 2.
8	The Music Club	info.themusicclubleth@outlook.com	2024-05-24 15:30:00	12	Drink refills every 30 mins.
9	The Music Club - EBOARD	info.themusicclubleth@outlook.com	2024-05-24 20:30:00	7	Drink refills every 30 mins. Vip Table Please.
10	The Music Club - EBOARD	info.themusicclubleth@outlook.com	2024-05-24 20:30:00	7	Drink refills every 30 mins. Vip Table Please.

DELETE RESERVATION BY ID

Delete Reservations

[Home](#) [Add Reservation](#) [Delete Reservation](#) [View Reservations](#)

Delete Reservation

Reservation ID to delete: (Please input the ID you wish to delete)

Delete

Delete Reservations

[Home](#) [Add Reservation](#) [Delete Reservation](#) [View Reservations](#)

Delete Reservation

Reservation ID to delete: (Please input the ID you wish to delete)

Delete

Reservation Deleted

Reservation has been successfully deleted.

[Delete Another Reservation](#)
[View Reservations](#)

VIEW ALL RESERVATIONS TO SEE IF RESERVATION DELETED

It can be s

View Reservations

Home Add Reservation Delete Reservation View Reservations					
All Reservations					
Reservation ID	Customer Name	Contact Info	Reservation Time	Number of Guests	Special Requests
1	John Doe	johndoe@example.com	2024-05-20 19:00:00	2	Window seat preferred
2	Jane Smith	janesmith@example.com	2024-05-21 20:00:00	4	Allergic to peanuts
3	Alice Johnson	alicej@example.com	2024-05-22 18:30:00	3	Celebrating anniversary
4	Bob Brown	bobbrown@example.com	2024-05-23 19:45:00	5	Need high chair for a toddler
5	Charlie Davis	charlied@example.com	2024-05-24 20:30:00	2	Vegetarian meal options
6	Maxwell Fellow	maxfellow@email.com	2024-05-24 20:30:00	5	Vip table for 5 please.
7	Vip Family Head	vipfamily@email.com	2024-05-24 20:30:00	4	Vip table for 4 please. Alcohol for only 2.
8	The Music Club	info.themusicclubleth@outlook.com	2024-05-24 15:30:00	12	Drink refills every 30 mins.
9	The Music Club - EBOARD	info.themusicclubleth@outlook.com	2024-05-24 20:30:00	7	Drink refills every 30 mins. Vip Table Please.

It can be seen that the reservation ID number 10 has been deleted from the reservations list just as we have desired to delete it. This means that the operation was a success.

REPEAT OPERATIONS AS DESIRED.

END OF REPORT.

LINK TO GITHUB FOR PROJECT FILES.

<https://github.com/ibrahimidd20/spring2024cis344final>