1. User input the passenger’s last name and first name and retrieve all trains they are booked on.

Query 1: SELECT t.train_name FROM train t

JOIN booked b ON b.train_number = t.train_number

JOIN passenger p ON p.ssn = b.passanger_ssn WHERE

p.first_name = '{first_name}' and p.last_name = '{last_name}';

This SQL query retrieves the names of trains that a passenger with a given first and last name has booked. It achieves this by joining the `train`, `booked`, and `passenger` tables and filtering on the given first and last name.

Here is an explanation of the query:

- `SELECT t.train_name`: This specifies the columns to be retrieved, in this case just the `train_name` column from the `train` table.
- `FROM train t JOIN booked b ON b.train_number = t.train_number JOIN passenger p ON p.ssn = b.passanger_ssn`: This specifies the tables to be joined, and the join conditions. The `train` table is joined with the `booked` table on their `train_number` columns, and the `booked` table is joined with the `passenger` table on their `passanger_ssn` and `ssn` columns, respectively.
- `WHERE p.first_name = '{first_name}' and p.last_name = '{last_name}'`: This specifies the filter condition, where only rows that satisfy the given first and last name are retrieved.

2. User input the Date and list of passengers travelling on entered day with confirmed tickets displays on UI.

Query 2: SELECT p.Last_Name, p.First_Name FROM Passenger p

JOIN Booked b ON p.ssn = b.passanger_ssn

JOIN train t ON t.train_number = b.train_number

JOIN train_status ts ON ts.train_name = t.train_name

WHERE ts.train_Date = '{travel_date}' AND b.Status = 'booked';

This query retrieves the last name and first name of the passengers who have booked a train for a specific travel date. The query joins four tables: "Passenger," "Booked," "Train," and "Train_Status."

The first join is between the "Passenger" table and "Booked" table on the passenger's SSN. The second join is between the "Train" table and the "Booked" table on the train number. The third join is between the "Train_Status" table and the "Train" table on the train name.

The query filters the results using the WHERE clause with two conditions. The first condition checks if the "train_date" in the "Train_Status" table matches the input "travel_date." The second condition checks if the status of the booked ticket is 'booked'.

The SELECT statement retrieves the passenger's first and last name from the "Passenger" table. The results are ordered by the passenger's last name.

3. User input the age of the passenger (50 to 60) and UI display the train information (Train Number, Train Name, Source and Destination) and passenger information (Name, Address, Category, ticket status) of passengers who are between the ages of 50 to 60.

Query 3: SELECT t.train_number, t.train_name,

```
 t.source_station, t.destination_station, p.first_name,

 p.last_name, p.address,b.ticket_type, b.status

 FROM Passenger p

JOIN Booked b ON p.ssn = b.passanger_ssn

JOIN train t ON t.train_number = b.train_number

WHERE TIMESTAMPDIFF(YEAR, p.bdate, CURDATE()) = {age};
```

This SQL query retrieves data from multiple tables using the JOIN clause and filtering results based on the age of passengers.

Here's a breakdown of the query:

- The SELECT statement is used to specify the columns that will be included in the query result set. The columns being selected are:
  - t.train_number

    - t.train_name
    - t.source_station
    - t.destination_station
    - p.first_name
    - p.last_name
    - p.address
    - b.ticket_type
    - b.status
- The FROM clause is used to specify the tables that will be used in the query. The tables being used are:
    - Passenger p
    - Booked b
    - train t
- The JOIN clause is used to combine the rows from the Passenger, Booked, and Train tables based on the following conditions:
    - p.ssn = b.passanger_ssn (join between Passenger and Booked tables)
    - t.train_number = b.train_number (join between Train and Booked tables)
- The WHERE clause is used to filter the results based on the age of passengers. The condition is:
    - TIMESTAMPDIFF(YEAR, p.bdate, CURDATE()) = {age}
    - The TIMESTAMPDIFF function calculates the difference in years between the birth date (p.bdate) and the current date (CURDATE()). The result is compared to the given age parameter to filter the results.
- The query returns the selected columns from the joined tables where the age of the passenger matches the given age parameter.

Overall, this query retrieves information about passengers who have booked train tickets, including the train number, train name, source and destination stations, passenger's first and last name, address, ticket type, and booking status. The results are filtered based on the age of the passenger.

4. List all the train names along with the count of passengers it is carrying.

Query 4: SELECT t.train_name, count(p.ssn) FROM train t

```
            JOIN booked b ON t.train_number = b.train_number

            JOIN passenger p ON b.Passanger_ssn = p.ssn

            WHERE b.status = 'booked' GROUP BY t.train_number;
```

This SQL query is selecting the train name and the count of passengers for each train that has at least one booked passenger. The query is joining three tables:

"train", "booked", and "passenger", and using the "train_number" and "Passenger_ssn" fields to link the tables.

The "JOIN" keyword is used to combine rows from two or more tables based on a related column between them. In this case, the "train" table is joined with the "booked" table using the "train_number" column, and then the "booked" table is joined with the "passenger" table using the "Passenger_ssn" column.

The "WHERE" clause is used to filter the rows based on a condition, in this case selecting only the rows where the "status" field in the "booked" table is equal to 'booked'. This ensures that only passengers who have booked a train seat are included in the count.

Finally, the "GROUP BY" clause is used to group the result set by the "train_number" field in the "booked" table, which allows the query to count the number of passengers for each train separately.

Overall, this query will return a result set with two columns: "train_name" and "count(p.ssn)". The "train_name" column will contain the name of each train that has at least one booked passenger, and the "count(p.ssn)" column will contain the number of passengers that have booked a seat on each train.

5. Enter a train name and retrieve all the passengers with confirmed status travelling in that train.

Query 5: SELECT t.train_name,count(*) FROM passenger p

```
JOIN booked b ON p.ssn = b.passanger_ssn

JOIN train t ON b.train_number = t.train_number

WHERE t.train_name = '{train_name}' AND b.status = 'booked';
```

This SQL query is selecting the count of passengers for a specific train that has at least one booked passenger. The query is joining three tables: "passenger", "booked", and "train", and using the "ssn", "train_number", and "train_name" fields to link the tables.

The "JOIN" keyword is used to combine rows from two or more tables based on a related column between them. In this case, the "passenger" table is joined with the "booked" table using the "ssn" column, and then the "booked" table is joined with the "train" table using the "train_number" column.

The "WHERE" clause is used to filter the rows based on two conditions. First, it selects only the rows where the "train_name" field in the "train" table is equal to the specified train name (which is passed as a parameter). This ensures that only passengers who have booked a seat on the specified train are included in the count. Second, it selects only the rows where the "status" field in the "booked" table is equal to 'booked'. This ensures that only passengers who have actually booked a seat (as opposed to canceled or waitlisted passengers) are included in the count.

Finally, the query uses the "COUNT(*)" function to count the number of rows in the result set. Since we are only selecting the count of passengers, the query does not include a GROUP BY clause, which means that all the passengers for the specified train will be counted together.

Overall, this query will return a result set with two columns: "train_name" and "count(*)". The "train_name" column will contain the specified train name, and the "count(*)" column will contain the number of passengers who have booked a seat on that train.*

Query 6: DELETE b FROM Booked b

```
        JOIN Passenger p ON p.ssn = b.Passanger_ssn

        JOIN Train t ON b.train_number = t.train_number

        WHERE t.train_name = '{train_name}' AND p.ssn = '{passengers_ssn}' AND
        b.status = 'booked';

         UPDATE Booked b

        SET b.status = 'Booked'

        where b.passanger_ssn = (

        select p.ssn from passenger p, train t

        where b.passanger_ssn = p.ssn and t.train_number = b.train_number and
        t.train_name = '{train_name}' and b.Ticket_Type = '{ticket_type}' and b.status
        = 'WaitL'

        ORDER BY b.passanger_ssn)

        LIMIT 1;
```

The first SQL statement is deleting records from the `Booked` table where the `train_name` matches the specified value and the `Passenger_ssn` matches the specified passenger's SSN, and the `status` is 'booked'.

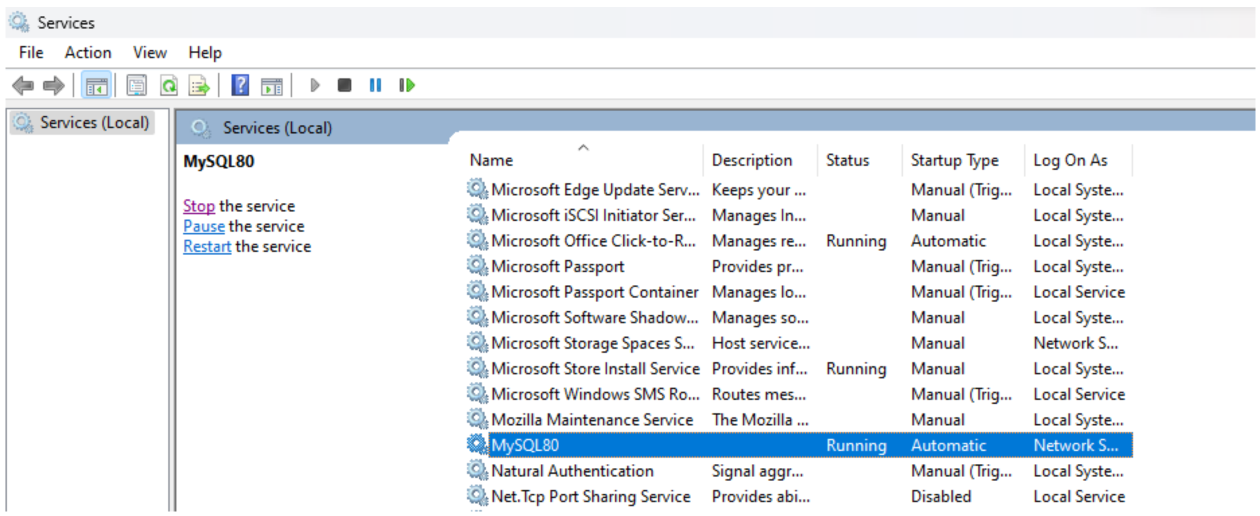
The second SQL statement is updating records in the `Booked` table where the `passenger_ssn` matches the SSN of the passenger who is waiting for a booking on the specified train with the specified `Ticket_Type` and the `status` is 'WaitL'. The `SET` clause updates the status to 'Booked'. The `SELECT` subquery is used to select the SSN of the passenger who is first in line for the waitlist, based on the specified criteria, and the `LIMIT` clause ensures that only one record is updated.

HOW TO RUN OUR PROGRAM:

Here are step-by-step instructions for implementing the project.

1. Download the zip file provided and extract its contents to a folder on your local computer.

2. Make sure SQL Server is installed and running on your local computer. If not, download and install it before proceeding.



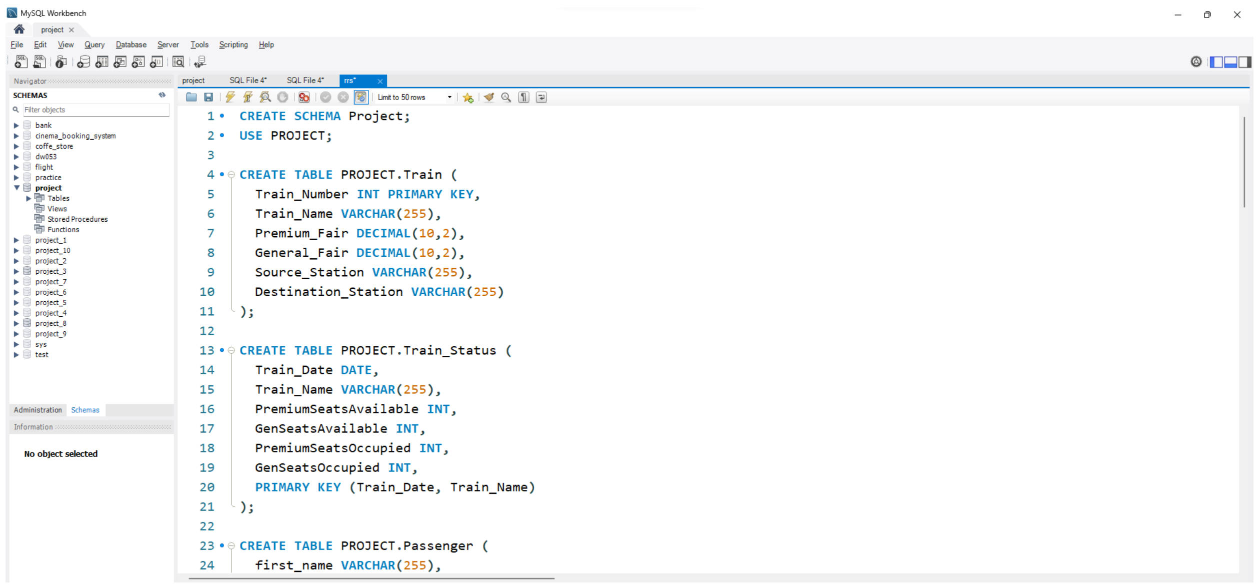
MySQL80

Stop the service
Pause the service
Restart the service

Name	Description	Status	Startup Type	Log On As
Microsoft Edge Update Serv...	Keeps your ...		Manual (Trig...	Local Syste...
Microsoft iSCSI Initiator Ser...	Manages In...		Manual	Local Syste...
Microsoft Office Click-to-R...	Manages re...	Running	Automatic	Local Syste...
Microsoft Passport	Provides pr...		Manual (Trig...	Local Syste...
Microsoft Passport Container	Manages lo...		Manual (Trig...	Local Service
Microsoft Software Shadow...	Manages so...		Manual	Local Syste...
Microsoft Storage Spaces S...	Host service...		Manual	Network S...
Microsoft Store Install Service	Provides inf...	Running	Manual	Local Syste...
Microsoft Windows SMS Ro...	Routes mes...		Manual (Trig...	Local Service
Mozilla Maintenance Service	The Mozilla ...		Manual	Local Syste...
MySQL80		Running	Automatic	Network S...
Natural Authentication	Signal aggr...		Manual (Trig...	Local Syste...
Net.Tcp Port Sharing Service	Provides abi...		Disabled	Local Service

You can check the status of the server in “services” in windows and “System Preferences” in MAC.

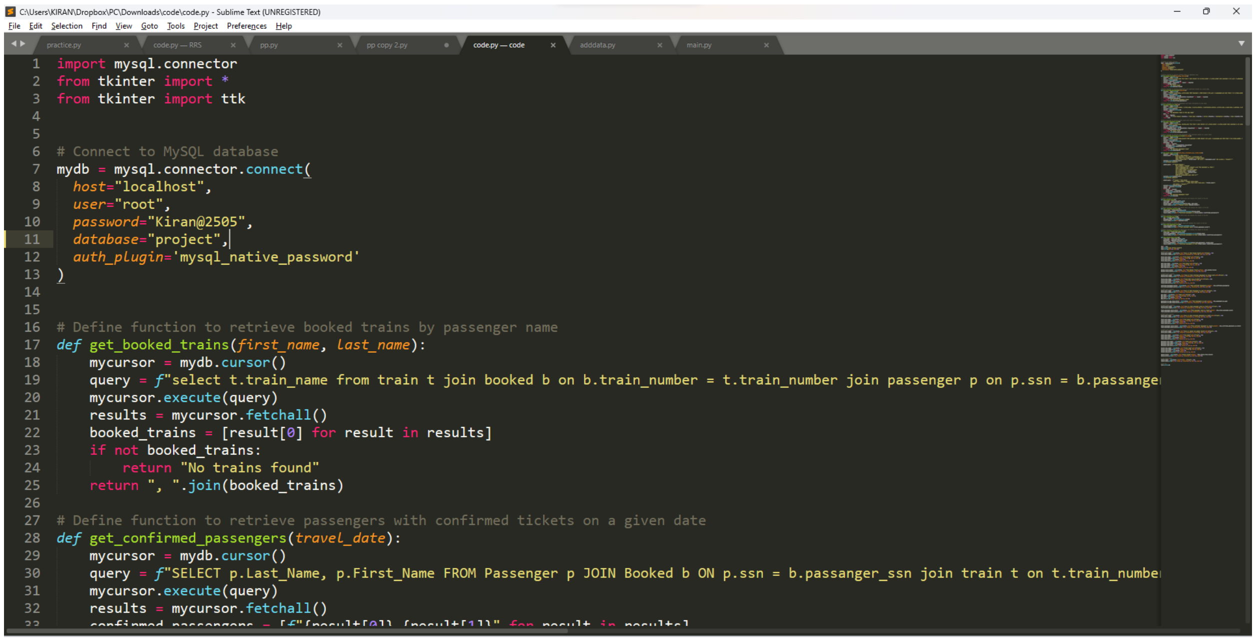
3. Open SQL environment and connect to the local SQL Server. Open the rrs.sql file from the extracted folder and run the code to create the database and tables required for the project.



```
CREATE SCHEMA Project;
USE PROJECT;

CREATE TABLE PROJECT.Train (
  Train_Number INT PRIMARY KEY,
  Train_Name VARCHAR(255),
  Premium_Fair DECIMAL(10,2),
  General_Fair DECIMAL(10,2),
  Source_Station VARCHAR(255),
  Destination_Station VARCHAR(255)
);

CREATE TABLE PROJECT.Train_Status (
  Train_Date DATE,
  Train_Name VARCHAR(255),
  PremiumSeatsAvailable INT,
  GenSeatsAvailable INT,
  PremiumSeatsOccupied INT,
  GenSeatsOccupied INT,
  PRIMARY KEY (Train_Date, Train_Name)
);

CREATE TABLE PROJECT.Passenger (
  first_name VARCHAR(255),
```

This is a MySQL window.

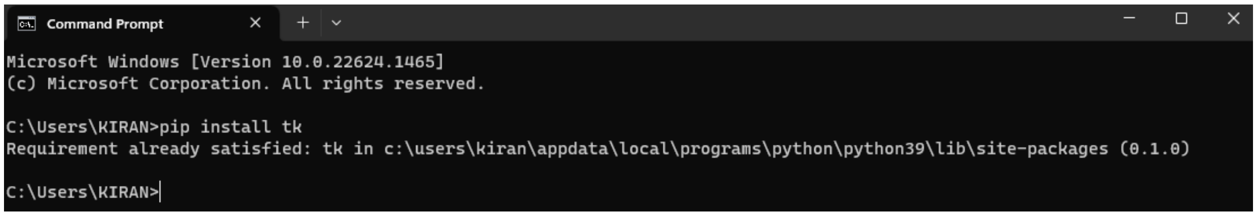
4. Open the code.py file in any text editor or integrated development environment (IDE) of your choice, such as Sublime Text, PyCharm, Visual Studio Code, or IDLE.


C:\Users\KIRAN\Dropbox\PC\Downloads\code\code.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
practice.py
code.py — RRS
pp.py
pp copy 2.py
code.py — code
adddata.py
main.py

```python
import mysql.connector
from tkinter import *
from tkinter import ttk


# Connect to MySQL database
mydb = mysql.connector.connect(
  host="localhost",
  user="root",
  password="Kiran@2505",
  database="project",
  auth_plugin='mysql_native_password'
)


# Define function to retrieve booked trains by passenger name
def get_booked_trains(first_name, last_name):
    mycursor = mydb.cursor()
    query = f"select t.train_name from train t join booked b on b.train_number = t.train_number join passenger p on p.ssn = b.passange
    mycursor.execute(query)
    results = mycursor.fetchall()
    booked_trains = [result[0] for result in results]
    if not booked_trains:
        return "No trains found"
    return ", ".join(booked_trains)

# Define function to retrieve passengers with confirmed tickets on a given date
def get_confirmed_passengers(travel_date):
    mycursor = mydb.cursor()
    query = f"SELECT p.Last_Name, p.First_Name FROM Passenger p JOIN Booked b ON p.ssn = b.passanger_ssn join train t on t.train_numbe
    mycursor.execute(query)
    results = mycursor.fetchall()
    confirmed_passengers = [f"{result[0]} {result[1]}" for result in results]
```

I used Sublime Text to run my python code.

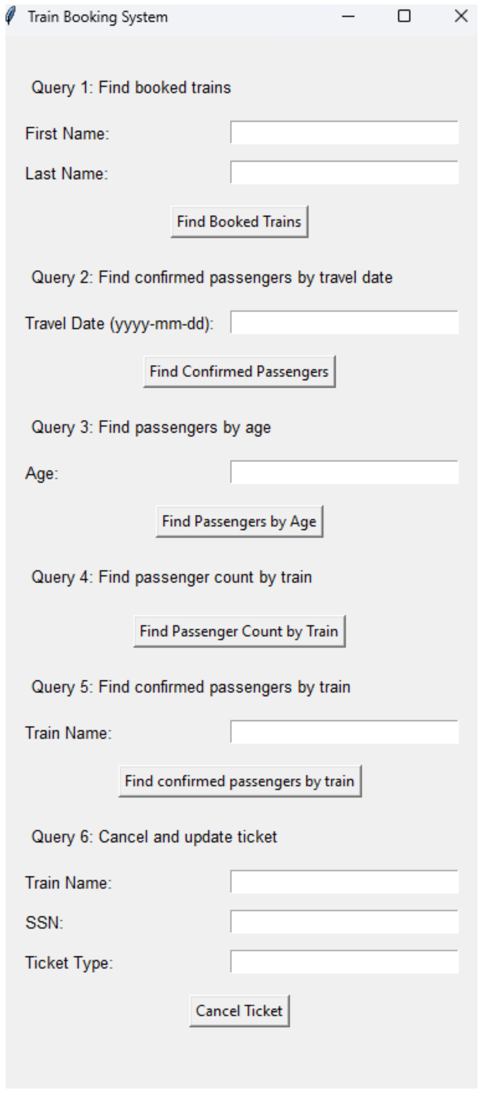
5. Install any required Python modules by running the command "pip install [module name]" in the terminal of your IDE or command prompt. For this project Tkinter is the required library. So, to install it use the pip install tkinter prompt.



```
Microsoft Windows [Version 10.0.22624.1465]
(c) Microsoft Corporation. All rights reserved.

C:\Users\KIRAN>pip install tk
Requirement already satisfied: tk in c:\users\kiran\appdata\local\programs\python\python39\lib\site-packages (0.1.0)

C:\Users\KIRAN>
```

As the library is already installed in my computer it is showing Requirement already satisfied.

6. As I used MySQL, to connect the python code to the database please make required changes in the code. i.e; replace user and password with yours in mydb.
7. Run the Python code “code.py” in your local computer.
8. Follow the prompts in the console to input the required user inputs.


Train Booking System
Query 1: Find booked trains
First Name:
Last Name:
Find Booked Trains
Query 2: Find confirmed passengers by travel date
Travel Date (yyyy-mm-dd):
Find Confirmed Passengers
Query 3: Find passengers by age
Age:
Find Passengers by Age
Query 4: Find passenger count by train
Find Passenger Count by Train
Query 5: Find confirmed passengers by train
Train Name:
Find confirmed passengers by train
Query 6: Cancel and update ticket
Train Name:
SSN:
Ticket Type:
Cancel Ticket

This is how the output window looks like.

9. Review the output generated by the code to verify that it is working correctly and producing the desired results.