In [ ]:
```
!pip install mysql-connector-python
```

In [ ]:
```
import mysql.connector as sql

cnx = sql.connect(host='127.0.0.1', user='root', password='11-Mar-04', database='Railway_reservation',auth_plugin='mysql_native_p
if cnx.is_connected() == False:
    print('Not connected')
else:
    print('Database Connected....')
```

In [ ]:
```
import mysql.connector
from mysql.connector import Error

class TicketBooker:
    def __init__(self):
        try:
            self.conn = mysql.connector.connect(
                host="127.0.0.1",
                user="root",
                password="11-Mar-04",
                database="train"
            )
            self.cursor = self.conn.cursor()
            self.available_lower_berths = 2
            self.available_middle_berths = 2
            self.available_upper_berths = 2
            self.available_rac_tickets = 2
            self.available_waiting_list = 2
            self.lower_berths_positions = list(range(1, 3))
            self.middle_berths_positions = list(range(1, 3))
            self.upper_berths_positions = list(range(1, 3))
            self.rac_positions = list(range(1, 3))
            self.waiting_list_positions = list(range(1, 3))
        except Error as e:
            print(f"Error: {e}")
            self.conn = None

    def add_to_waiting_list(self, passenger, position, status):
        self.book_ticket(passenger, position, status)

    def cancel_ticket(self, passenger_id):
        if self.conn:
            self.cursor.execute('DELETE FROM passengers WHERE id = %s', (passenger_id,))
            self.conn.commit()
            print(f"Ticket cancelled for Passenger ID {passenger_id}")

    def print_available(self):
        print("Available Lower Berths:", self.available_lower_berths)
        print("Available Middle Berths:", self.available_middle_berths)
        print("Available Upper Berths:", self.available_upper_berths)
        print("Available RAC Tickets:", self.available_rac_tickets)
        print("Available Waiting List:", self.available_waiting_list)

    def print_passengers(self):
        if self.conn:
            self.cursor.execute('SELECT * FROM passengers')
            rows = self.cursor.fetchall()
            for row in rows:
                print(row)

    def book_ticket(self, passenger, position, status):
        self.cursor.execute(
            "INSERT INTO passengers (name, age, berth_preference, position, status) VALUES (%s, %s, %s, %s, %s)",
            (passenger.name, passenger.age, passenger.berth_preference, position, status)
        )
        self.conn.commit()

class Passenger:
    def __init__(self, name, age, berth_preference):
        self.name = name
        self.age = age
        self.berth_preference = berth_preference

def book_ticket(booker, passenger):
    if (passenger.berth_preference == "L" and booker.available_lower_berths > 0):
        print("Preferred Berth Available")
        booker.book_ticket(passenger, booker.lower_berths_positions.pop(0), "L")
        booker.available_lower_berths -= 1

    elif (passenger.berth_preference == "M" and booker.available_middle_berths > 0):
        print("Preferred Berth Available")
        booker.book_ticket(passenger, booker.middle_berths_positions.pop(0), "M")
        booker.available_middle_berths -= 1

    elif (passenger.berth_preference == "U" and booker.available_upper_berths > 0):
            print("Preferred Berth Available")
            booker.book_ticket(passenger, booker.upper_berths_positions.pop(0), "U")
            booker.available_upper_berths -= 1

    elif booker.available_lower_berths > 0:
        print("Lower Berth Given")
        booker.book_ticket(passenger, booker.lower_berths_positions.pop(0), "L")
        booker.available_lower_berths -= 1

    elif booker.available_middle_berths > 0:
        print("Middle Berth Given")
        booker.book_ticket(passenger, booker.middle_berths_positions.pop(0), "M")
        booker.available_middle_berths -= 1

    elif booker.available_upper_berths > 0:
        print("Upper Berth Given")
        booker.book_ticket(passenger, booker.upper_berths_positions.pop(0), "U")
        booker.available_upper_berths -= 1

    elif booker.available_rac_tickets > 0:
        print("RAC Available")
        booker.add_to_rac(passenger, booker.rac_positions.pop(0), "RAC")

    elif booker.available_waiting_list > 0:
        print("Added to Waiting List")
        booker.add_to_waiting_list(passenger, booker.waiting_list_positions.pop(0), "WL")

def main():
    global booker
    booker = TicketBooker()
    while True:
        print("1. Book Ticket\n2. Cancel Ticket\n3. Available Tickets\n4. Booked Tickets\n5. Exit")
        choice = int(input("Enter your choice: "))
        if choice == 1:
            name = input("Enter your name: ")
            age = int(input("Enter your age: "))
            berth_preference = input("Enter berth preference (L for Lower, M for Middle, U for Upper): ")
            passenger = Passenger(name, age, berth_preference)
            book_ticket(booker, passenger)
        elif choice == 2:
            passenger_id = int(input("Enter passenger ID to cancel: "))
            booker.cancel_ticket(passenger_id)
        elif choice == 3:
            booker.print_available()
        elif choice == 4:
            booker.print_passengers()
        elif choice == 5:
            print("Exiting...")
            break
        else:
            print("Invalid choice, please try again.")

if __name__ == "__main__":
    main()
```

```
1. Book Ticket
2. Cancel Ticket
3. Available Tickets
4. Booked Tickets
5. Exit
Enter your choice: 1
Enter your name: renu
Enter your age: 39
Enter berth preference (L for Lower, M for Middle, U for Upper): L
Preferred Berth Available
1. Book Ticket
2. Cancel Ticket
3. Available Tickets
4. Booked Tickets
5. Exit
Enter your choice: 1
Enter your name: vinu
Enter your age: 25
Enter berth preference (L for Lower, M for Middle, U for Upper): L
Preferred Berth Available
1. Book Ticket
2. Cancel Ticket
3. Available Tickets
4. Booked Tickets
5. Exit
Enter your choice: 1
Enter your name: preethi
Enter your age: 26
Enter berth preference (L for Lower, M for Middle, U for Upper): L
Middle Berth Given
1. Book Ticket
2. Cancel Ticket
3. Available Tickets
4. Booked Tickets
5. Exit
Enter your choice: 4
(2, 'dhoni', 40, 'u', 1, 'L')
(3, 'kohli', 35, 'l', 1, 'L')
(4, 'harshini', 18, 'm', 2, 'L')
(5, 'harini', 20, 'u', 3, 'L')
(6, 'divya', 19, 'U', 1, 'U')
(7, 'meenu', 30, 'U', 1, 'U')
(8, 'riya', 50, 'L', 1, 'L')
(9, 'aaa', 45, 'U', 1, 'U')
```

```
(10, 'anu', 21, 'L', 1, 'L')
(11, 'renu', 39, 'L', 1, 'L')
(12, 'vinu', 25, 'L', 2, 'L')
(13, 'preethi', 26, 'L', 1, 'M')
1. Book Ticket
2. Cancel Ticket
3. Available Tickets
4. Booked Tickets
5. Exit
Enter your choice: 2
Enter passenger ID to cancel: 9
Ticket cancelled for Passenger ID 9
1. Book Ticket
2. Cancel Ticket
3. Available Tickets
4. Booked Tickets
5. Exit
Enter your choice: 4
(2, 'dhoni', 40, 'u', 1, 'L')
(3, 'kohli', 35, 'l', 1, 'L')
(4, 'harshini', 18, 'm', 2, 'L')
(5, 'harini', 20, 'u', 3, 'L')
(6, 'divya', 19, 'U', 1, 'U')
(7, 'meenu', 30, 'U', 1, 'U')
(8, 'riya', 50, 'L', 1, 'L')
(10, 'anu', 21, 'L', 1, 'L')
(11, 'renu', 39, 'L', 1, 'L')
(12, 'vinu', 25, 'L', 2, 'L')
(13, 'preethi', 26, 'L', 1, 'M')
1. Book Ticket
2. Cancel Ticket
3. Available Tickets
4. Booked Tickets
5. Exit
```

```
In [ ]:
```

```
In [ ]:
```