Railway database management

```
CREATE DATABASE railway_management;

USE railway_management;

CREATE TABLE Trains (
    TrainID INT PRIMARY KEY AUTO_INCREMENT,
    TrainName VARCHAR(255),
    DepartureStation VARCHAR(255),
    ArrivalStation VARCHAR(255),
    DepartureTime TIME,
    ArrivalTime TIME
);

CREATE TABLE Stations (
    StationID INT PRIMARY KEY AUTO_INCREMENT,
    StationName VARCHAR(255)
);

CREATE TABLE Passengers (
    PassengerID INT PRIMARY KEY AUTO_INCREMENT,
    PassengerName VARCHAR(255),
    Age INT
);

CREATE TABLE Tickets (
    TicketID INT PRIMARY KEY AUTO_INCREMENT,
    TrainID INT,
    PassengerID INT,
```

```
    SeatNumber INT,
    BookingDate DATE,
    FOREIGN KEY (TrainID) REFERENCES Trains(TrainID),
    FOREIGN KEY (PassengerID) REFERENCES Passengers(PassengerID)
);

CREATE TABLE Routes (
    RouteID INT PRIMARY KEY AUTO_INCREMENT,
    TrainID INT,
    StationOrder INT,
    StationID INT,
    FOREIGN KEY (TrainID) REFERENCES Trains(TrainID),
    FOREIGN KEY (StationID) REFERENCES Stations(StationID)
);

-- Insert into Stations
INSERT INTO Stations (StationName) VALUES
    ('Station A'),
    ('Station B'),
    ('Station C'),
    ('Station D'),
    ('Station E');

-- Insert into Trains
INSERT INTO Trains (TrainName, DepartureStation, ArrivalStation,
DepartureTime, ArrivalTime) VALUES
    ('Train 1', 'Station A', 'Station E', '08:00:00', '12:00:00'),
    ('Train 2', 'Station B', 'Station D', '09:00:00', '11:30:00'),
    ('Train 3', 'Station C', 'Station A', '11:30:00', '14:45:00'),
    ('Train 4', 'Station D', 'Station B', '14:00:00', '16:30:00'),
    ('Train 5', 'Station E', 'Station C', '15:30:00', '18:45:00');
```

```
-- Insert into Passengers
INSERT INTO Passengers (PassengerName, Age) VALUES
    ('Alice', 25),
    ('Bob', 32),
    ('Charlie', 45),
    ('David', 28),
    ('Eve', 19);

-- Insert into Tickets
INSERT INTO Tickets (TrainID, PassengerID, SeatNumber, BookingDate)
VALUES
    (1, 1, 101, '2023-09-15'),
    (1, 2, 102, '2023-09-15'),
    (2, 3, 103, '2023-09-16'),
    (2, 4, 104, '2023-09-16'),
    (3, 5, 105, '2023-09-17');

-- Insert into Routes
INSERT INTO Routes (TrainID, StationOrder, StationID) VALUES
    (1, 1, 1),
    (1, 2, 2),
    (1, 3, 3),
    (1, 4, 4),
    (1, 5, 5);
```

1) Let's say you want to find all passengers who have booked a ticket for "Train 1" departing from "Station A" to "Station E"

```
SELECT Passengers.PassengerName
FROM Passengers
JOIN Tickets ON Passengers.PassengerID = Tickets.PassengerID
JOIN Trains ON Tickets.TrainID = Trains.TrainID
```

```
WHERE Trains.TrainName = 'Train 1'
AND Trains.DepartureStation = 'Station A'
AND Trains.ArrivalStation = 'Station E';
```

2) **List all train names that operate between two specific stations (e.g., Station A and Station B):**

```
SELECT DISTINCT TrainName
FROM Trains
WHERE DepartureStation = 'Station A' AND ArrivalStation = 'Station B';
```

3) **Find the average age of passengers who have booked tickets:**

```
SELECT AVG(Age) AS AverageAge
FROM Passengers;
```

4) **Retrieve a list of all passengers who booked a ticket for a specific train (e.g., Train 1):**

```
SELECT PassengerName
FROM Passengers
WHERE PassengerID IN (
    SELECT PassengerID
    FROM Tickets
    WHERE TrainID = 1 -- Replace with the desired TrainID
);
```

5) **Count the number of available seats for a specific train (e.g., Train 2):**

```
SELECT (SELECT COUNT(*) FROM Trains WHERE TrainID = 2) -
COUNT(*) AS AvailableSeats
FROM Tickets
WHERE TrainID = 2;
```

6) **Find the station names and their order for a specific route (e.g., RouteID 1):**

```
SELECT Stations.StationName, Routes.StationOrder
FROM Stations
JOIN Routes ON Stations.StationID = Routes.StationID
WHERE RouteID = 1; -- Replace with the desired RouteID
```

7) **List all passengers who have not booked any tickets:**

```
SELECT PassengerName
FROM Passengers
WHERE PassengerID NOT IN (SELECT DISTINCT PassengerID FROM Tickets);
```

8) **Retrieve the train name and duration of the longest journey:**

```
SELECT TrainName, TIMEDIFF(ArrivalTime, DepartureTime) AS Duration
FROM Trains
ORDER BY Duration DESC
LIMIT 1;
```

9) **Calculate the total number of tickets booked for a specific train (e.g., Train 3):**

```
SELECT COUNT(*) AS TotalTickets
FROM Tickets
WHERE TrainID = 3; -- Replace with the desired TrainID
```

10) **Find the most popular departure station (station with the most departures):**

```
SELECT DepartureStation, COUNT(*) AS DepartureCount
FROM Trains
```

```
GROUP BY DepartureStation
ORDER BY DepartureCount DESC
LIMIT 1;
```

11) **etrieve the passenger name, train name, and booking date for all tickets booked by passengers aged 25 or younger:**

```
SELECT Passengers.PassengerName, Trains.TrainName,
Tickets.BookingDate
FROM Passengers
JOIN Tickets ON Passengers.PassengerID = Tickets.PassengerID
JOIN Trains ON Tickets.TrainID = Trains.TrainID
WHERE Passengers.Age <= 25;
```

12) **List all train names and their respective departure and arrival stations:**

```
SELECT TrainName, DepartureStation, ArrivalStation

FROM Trains;
```

13) **Find the passengers who have booked a ticket with a specific seat number (e.g., SeatNumber 101):**

```
SELECT DISTINCT Passengers.PassengerName

FROM Passengers

JOIN Tickets ON Passengers.PassengerID = Tickets.PassengerID

WHERE Tickets.SeatNumber = 101; -- Replace with the desired
SeatNumber
```

14) Calculate the total number of stations in the database:

```
SELECT COUNT(*) AS TotalStations
FROM Stations;
```

15) List the passengers who have booked multiple tickets (more than one ticket):

```
SELECT Passengers.PassengerName, COUNT(*) AS TicketCount
FROM Passengers
JOIN Tickets ON Passengers.PassengerID = Tickets.PassengerID
GROUP BY Passengers.PassengerName
HAVING TicketCount > 1;
```

16) Find the train with the earliest departure time:

```
SELECT TrainName, DepartureTime
FROM Trains
ORDER BY DepartureTime
LIMIT 1;
```

17) Retrieve the number of tickets booked for each train:

```
SELECT Trains.TrainName, COUNT(*) AS TicketCount
FROM Trains
LEFT JOIN Tickets ON Trains.TrainID = Tickets.TrainID
GROUP BY Trains.TrainName;
```

18) List all passengers who have booked tickets for a specific station (e.g., Station C):

```
SELECT DISTINCT Passengers.PassengerName
FROM Passengers
JOIN Tickets ON Passengers.PassengerID = Tickets.PassengerID
JOIN Routes ON Tickets.TrainID = Routes.TrainID
WHERE Routes.StationID = (
    SELECT StationID
    FROM Stations
    WHERE StationName = 'Station C'
);
```

19) Calculate the total number of routes for each train:

```
SELECT Trains.TrainName, COUNT(*) AS RouteCount
FROM Trains
LEFT JOIN Routes ON Trains.TrainID = Routes.TrainID
```

```
GROUP BY Trains.TrainName;
```

20) Retrieve a list of all train names and their respective departure and arrival times sorted by departure time:

```
SELECT TrainName, DepartureTime, ArrivalTime

FROM Trains

ORDER BY DepartureTime;
```

21) Find the passengers who have booked tickets for more than one train:

```
SELECT Passengers.PassengerName, COUNT(DISTINCT
Tickets.TrainID) AS TrainCount
FROM Passengers
JOIN Tickets ON Passengers.PassengerID = Tickets.PassengerID
GROUP BY Passengers.PassengerName
HAVING TrainCount > 1;
```

22) Retrieve all train information:

```
SELECT * FROM trains;
```

23) Retrieve all station information:

```
SELECT * FROM stations;
```

24) Retrieve all passenger information

```
SELECT * FROM passengers;
```

25) Retrieve all ticket information:

```
SELECT * FROM tickets;
```

26)Retrieve all route information:

```
SELECT * FROM routes;
```

27) Find all passengers on a specific train (e.g., train_id = 123):

```
SELECT * FROM routes;
```

28)Find all tickets for a specific passenger (e.g., passenger_id = 456):

```
SELECT * FROM tickets WHERE passenger_id = 456;
```

29)List all routes that start from a specific station (e.g., station_id = 789):

```
SELECT * FROM routes WHERE start_station_id = 789;
```

30)Find the total number of passengers on a specific train (e.g., train_id = 123):

```
SELECT COUNT(*) FROM passengers WHERE train_id = 123;
```

31) Find the total number of tickets booked by a specific passenger (e.g., passenger_id = 456):

```
SELECT COUNT(*) FROM tickets WHERE passenger_id = 456;
```

32)Find the most popular route (route with the highest ticket bookings):

```
SELECT route_id, COUNT(*) AS ticket_count FROM tickets
GROUP BY route_id
ORDER BY ticket_count DESC
LIMIT 1;
```

33)Find the average ticket price for a specific train (e.g., train_id = 123):

```
SELECT AVG(ticket_price) FROM tickets WHERE train_id = 123;
```

34) Find all available trains from one station to another (e.g., from station_id = 789 to station_id = 456):

```
SELECT DISTINCT t.* FROM trains t
INNER JOIN routes r1 ON t.route_id = r1.route_id
INNER JOIN routes r2 ON t.route_id = r2.route_id
WHERE r1.end_station_id = 789 AND r2.start_station_id = 456;
```

35) Find the passenger with the most tickets booked:

```
SELECT passenger_id, COUNT(*) AS ticket_count FROM tickets
GROUP BY passenger_id
ORDER BY ticket_count DESC
LIMIT 1;
```

36)List all stations served by a specific train (e.g., train_id = 123):

```
SELECT s.* FROM stations s
INNER JOIN routes r ON s.station_id = r.start_station_id OR
s.station_id = r.end_station_id
WHERE r.train_id = 123;
```

37)Find the total revenue generated by a specific train (e.g., train_id = 123):

```
SELECT SUM(ticket_price) FROM tickets WHERE train_id = 123;
```

38)Find the route with the highest ticket revenue:

```
SELECT r.route_id, SUM(t.ticket_price) AS revenue FROM routes r
INNER JOIN tickets t ON r.route_id = t.route_id
GROUP BY r.route_id
ORDER BY revenue DESC
LIMIT 1;
```

39)List all passengers who booked a ticket for a specific route (e.g., route_id = 789):

```
SELECT DISTINCT p.* FROM passengers p
INNER JOIN tickets t ON p.passenger_id = t.passenger_id
WHERE t.route_id = 789;
```

40) Find the number of routes serving a specific station (e.g., station_id = 456):

```
SELECT COUNT(*) FROM routes WHERE start_station_id = 456 OR
end_station_id = 456;
```

41) List all tickets booked on a specific date (e.g., date = '2023-09-14'):

```
SELECT * FROM tickets WHERE booking_date = '2023-09-14';
```

42) Find the number of available seats on a specific train (e.g., train_id = 123):

```
SELECT (seating_capacity - COUNT(*)) AS available_seats FROM trains
LEFT JOIN tickets ON trains.train_id = tickets.train_id
WHERE trains.train_id = 123
GROUP BY trains.train_id;
```

43) Find the passenger who traveled the farthest (based on the distance between start and end stations):

```
SELECT p.*, (s1.distance + s2.distance) AS total_distance FROM passengers p
JOIN tickets t ON p.passenger_id = t.passenger_id
JOIN routes r ON t.route_id = r.route_id
JOIN stations s1 ON r.start_station_id = s1.station_id
JOIN stations s2 ON r.end_station_id = s2.station_id
ORDER BY total_distance DESC
LIMIT 1;
```

44) Find the busiest station (station with the most departures and arrivals):

```
SELECT station_id, COUNT(*) AS traffic FROM (
  SELECT start_station_id AS station_id FROM routes
  UNION ALL
  SELECT end_station_id AS station_id FROM routes
) AS combined
GROUP BY station_id
ORDER BY traffic DESC
```

```
LIMIT 1;
```

45) Find the most common departure time for all trains:

```
SELECT departure_time, COUNT(*) AS frequency FROM trains
GROUP BY departure_time
ORDER BY frequency DESC
LIMIT 1;
```

46) Find the passengers who booked a ticket but did not board the train:

```
SELECT p.* FROM passengers p
LEFT JOIN tickets t ON p.passenger_id = t.passenger_id
WHERE t.ticket_id IS NULL;
```

47) Find the number of passengers on each train route:

```
SELECT r.route_id, COUNT(*) AS passenger_count FROM routes r
JOIN tickets t ON r.route_id = t.route_id
GROUP BY r.route_id;
```

48) Find the routes with no booked tickets:

```
SELECT r.route_id FROM routes r
LEFT JOIN tickets t ON r.route_id = t.route_id
WHERE t.ticket_id IS NULL;
```

49) Find the passengers who booked a ticket for a specific route (e.g., route_id = 789) but did not board the train:

```
SELECT p.* FROM passengers p
LEFT JOIN tickets t ON p.passenger_id = t.passenger_id
WHERE t.route_id = 789 AND t.ticket_id IS NULL;
```

50) Find the passengers who traveled the most on a specific route (e.g., route_id = 789):

```
SELECT p.passenger_id, SUM(r.distance) AS total_distance FROM passengers p
JOIN tickets t ON p.passenger_id = t.passenger_id
JOIN routes r ON t.route_id = r.route_id
WHERE r.route_id = 789
GROUP BY p.passenger_id
ORDER BY total_distance DESC
LIMIT 1;
```

51) Find the stations with no train service (neither as start nor end stations):

```
SELECT s.station_id FROM stations s
LEFT JOIN routes r1 ON s.station_id = r1.start_station_id
LEFT JOIN routes r2 ON s.station_id = r2.end_station_id
WHERE r1.route_id IS NULL AND r2.route_id IS NULL;
```