Assignment: Database Operations for Railway Management System

Duration: 2 Hours

Scenario:

The Railway Management System is a critical part of the transportation sector that manages trains, schedules, routes, passengers, and ticket bookings. You have been appointed as the database consultant to design and implement a database system that efficiently manages the day-to-day operations of the railway system.

# **Task 1: Database Creation and Table Setup**

1. **Create a database named RailwayManagementDB.**

**Ans :**

mysql> CREATE DATABASE RailwayManagementDB;

Query OK, 1 row affected (0.01 sec)

mysql> use RailwayMANAGEMENTDB;

Database changed.

1. **Create the following tables to track train information, schedules, routes, passengers, and bookings:** 
   * **Trains: To store information about trains (TrainID, TrainName, TrainType, TotalSeats). o Routes: To store information about routes (RouteID, StartStation, EndStation, Distance).**
   * **Schedules: To store train schedules (ScheduleID, TrainID, RouteID, DepartureTime, ArrivalTime).**
   * **Passengers: To store passenger information (PassengerID, FirstName, LastName, Age, Email).**
   * **Bookings: To store booking details (BookingID, PassengerID, ScheduleID, BookingDate, SeatNumber).**

**Answer :-**

1. mysql> CREATE DATABASE RailwayManagementDB;

Query OK, 1 row affected (0.01 sec)

mysql> use RailwayMANAGEMENTDB;

Database changed



* mysql> **create table Trains**(TrainID INT, TrainName VARCHAR(100),TraintType VARCHAR(50),TotalSeats INT);

Query OK, 0 rows affected (0.04 sec)

* **mysql> CREATE TABLE Routes**(

RouteID INT,StartStation VARCHAR(100),EndStation VARCHAR(100),Distance INT);

Query OK, 0 rows affected (0.04 sec)

* mysql> **CREATE TABLE Passengers**( PassengerID INT, FirstName VARCHAR(50),LastName VARCHAR(50), Age INT, Email VARCHAR(100));

Query OK, 0 rows affected (0.03 sec)

* mysql**> CREATE TABLE Bookings**(

BookingID INT, PassengersID INT, ScheduleID INT, BookingDate DATE, SeatNumber INT);

Query OK, 0 rows affected (0.03 sec)

* mysql> CREATE TABLE **Schedules** ( ScheduleID INT, TrainID INT, RouteID INT, DepartureTime DATETIME, ArrivalTime DATETIME );

Query OK, 0 rows affected (0.06 sec)

* **mysql> DESC Trains;**

+------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+------------+--------------+------+-----+---------+-------+

| TrainID | int | YES | | NULL | |

| TrainName | varchar(100) | YES | | NULL | |

| TraintType | varchar(50) | YES | | NULL | |

| TotalSeats | int | YES | | NULL | |

+------------+--------------+------+-----+---------+-------+

4 rows in set (0.01 sec)

* **mysql> DESC Routes;**

+--------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------------+--------------+------+-----+---------+-------+

| RouteID | int | YES | | NULL | |

| StartStation | varchar(100) | YES | | NULL | |

| EndStation | varchar(100) | YES | | NULL | |

| Distance | int | YES | | NULL | |

+--------------+--------------+------+-----+---------+-------+

4 rows in set (0.00 sec)

* mysql> DESC **Passengers;**

+-------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------------+--------------+------+-----+---------+-------+

| PassengerID | int | YES | | NULL | |

| FirstName | varchar(50) | YES | | NULL | |

| LastName | varchar(50) | YES | | NULL | |

| Age | int | YES | | NULL | |

| Email | varchar(100) | YES | | NULL | |

+-------------+--------------+------+-----+---------+-------+

5 rows in set (0.00 sec)

* mysql> DESC **Bookings;**

+--------------+------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------------+------+------+-----+---------+-------+

| BookingID | int | YES | | NULL | |

| PassengersID | int | YES | | NULL | |

| ScheduleID | int | YES | | NULL | |

| BookingDate | date | YES | | NULL | |

| SeatNumber | int | YES | | NULL | |

+--------------+------+------+-----+---------+-------+

5 rows in set (0.00 sec)

* mysql> desc **Schedules;**

+---------------+----------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+---------------+----------+------+-----+---------+-------+

| ScheduleID | int | YES | | NULL | |

| TrainID | int | YES | | NULL | |

| RouteID | int | YES | | NULL | |

| DepartureTime | datetime | YES | | NULL | |

| ArrivalTime | datetime | YES | | NULL | |

+---------------+----------+------+-----+---------+-------+

1. **Insert sample data into the tables for at least:** 
   * **5 trains o 3 routes o 5 schedules o 10 passengers o 5 bookings**

**Ans :**

* **Trains :**

mysql> **INSERT INTO Trains VALUES**

-> (1, 'Rajdhani Express', 'Express', 300),

-> (2, 'Tejas Express', 'Superfast', 200),

-> (3, 'Shatabdi Express', 'Passenger', 250),

-> (4, 'Duronto Express', 'Superfast', 150),

-> (5, 'Garib Rath', 'Express', 350);

Query OK, 5 rows affected (0.01 sec)

Records: 5 Duplicates: 0 Warnings: 0

* **mysql> SELECT \* FROM Trains;**

**+---------+------------------+------------+------------+**

**| TrainID | TrainName | TraintType | TotalSeats |**

**+---------+------------------+------------+------------+**

**| 1 | Rajdhani Express | Express | 300 |**

**| 2 | Tejas Express | Superfast | 200 |**

**| 3 | Shatabdi Express | Passenger | 250 |**

**| 4 | Duronto Express | Superfast | 150 |**

**| 5 | Garib Rath | Express | 350 |**

**+---------+------------------+------------+-----------**

* **Routes :**

mysql> **INSERT INTO Routes VALUES**

-> (1,'Delhi','Mumbai',1400),

-> (2,'Kolkata','Chennai',1650),

-> (3,'Jaipur','Ahmedabad',650);

Query OK, 3 rows affected (0.01 sec)

Records: 3 Duplicates: 0 Warnings: 0

* **mysql> SELECT \* FROM Routes;**

**+---------+--------------+------------+----------+**

**| RouteID | StartStation | EndStation | Distance |**

**+---------+--------------+------------+----------+**

**| 1 | Delhi | Mumbai | 1400 |**

**| 2 | Kolkata | Chennai | 1650 |**

**| 3 | Jaipur | Ahmedabad | 650 |**

**+---------+--------------+------------+----------+**

* **Schedules :**

mysql> INSERT INTO Schedules VALUES

-> (1, 1, 1, '2024-10-20 09:00:00', '2024-10-20 21:00:00'),

-> (2, 2, 2, '2024-10-21 08:30:00', '2024-10-21 22:00:00'),

-> (3, 3, 3, '2024-10-22 06:00:00', '2024-10-22 14:00:00'),

-> (4, 4, 1, '2024-10-23 10:00:00', '2024-10-23 22:00:00'),

-> (5, 5, 2, '2024-10-24 07:00:00', '2024-10-24 21:00:00');

Query OK, 5 rows affected (0.02 sec)

Records: 5 Duplicates: 0 Warnings: 0

* **mysql> SELECT \* FROM Schedules;**

**+------------+---------+---------+---------------------+---------------------+**

**| ScheduleID | TrainID | RouteID | DepartureTime | ArrivalTime |**

**+------------+---------+---------+---------------------+---------------------+**

**| 1 | 1 | 1 | 2024-10-20 09:00:00 | 2024-10-20 21:00:00 |**

**| 2 | 2 | 2 | 2024-10-21 08:30:00 | 2024-10-21 22:00:00 |**

**| 3 | 3 | 3 | 2024-10-22 06:00:00 | 2024-10-22 14:00:00 |**

**| 4 | 4 | 1 | 2024-10-23 10:00:00 | 2024-10-23 22:00:00 |**

**| 5 | 5 | 2 | 2024-10-24 07:00:00 | 2024-10-24 21:00:00 |**

**+------------+---------+---------+--------------------**

* **Passengers :**

mysql> INSERT INTO Passengers VALUES

-> (1, 'Rajesh', 'Sharma', 45, 'rajesh.sharma@specialforce.com'),

-> (2, 'Priya', 'Mehra', 32, 'priya.mehra@specialforce.com'),

-> (3, 'Ankit', 'Verma', 29, 'ankit.verma@specialforce.com'),

-> (4, 'Kavita', 'Gupta', 40, 'kavita.gupta@specialforce.com'),

-> (5, 'Arun', 'Patel', 50, 'arun.patel@specialforce.com'),

-> (6, 'Neha', 'Joshi', 27, 'neha.joshi@specialforce.com'),

-> (7, 'Suresh', 'Nair', 33, 'suresh.nair@specialforce.com'),

-> (8, 'Pooja', 'Reddy', 36, 'pooja.reddy@specialforce.com'),

-> (9, 'Vikram', 'Singh', 42, 'vikram.singh@specialforce.com'),

-> (10, 'Aarti', 'Desai', 25, 'aarti.desai@specialforce.com');

Query OK, 10 rows affected (0.01 sec)

Records: 10 Duplicates: 0 Warnings: 0

* **mysql> SELECT \* FROM Passengers;**

**+-------------+-----------+----------+------+--------------------------------+**

**| PassengerID | FirstName | LastName | Age | Email |**

**+-------------+-----------+----------+------+--------------------------------+**

**| 1 | Rajesh | Sharma | 45 | rajesh.sharma@specialforce.com |**

**| 2 | Priya | Mehra | 32 | priya.mehra@specialforce.com |**

**| 3 | Ankit | Verma | 29 | ankit.verma@specialforce.com |**

**| 4 | Kavita | Gupta | 40 | kavita.gupta@specialforce.com |**

**| 5 | Arun | Patel | 50 | arun.patel@specialforce.com |**

**| 6 | Neha | Joshi | 27 | neha.joshi@specialforce.com |**

**| 7 | Suresh | Nair | 33 | suresh.nair@specialforce.com |**

**| 8 | Pooja | Reddy | 36 | pooja.reddy@specialforce.com |**

**| 9 | Vikram | Singh | 42 | vikram.singh@specialforce.com |**

**| 10 | Aarti | Desai | 25 | aarti.desai@specialforce.com |**

**+-------------+-----------+----------+------+--------------------------------+**

* **Bookings :**

mysql> INSERT INTO Bookings VALUES

-> (1, 1, 1, '2024-10-10', 12),

-> (2, 2, 1, '2024-10-11', 34),

-> (3, 3, 2, '2024-10-12', 56),

-> (4, 4, 3, '2024-10-13', 18),

-> (5, 5, 4, '2024-10-14', 22);

Query OK, 5 rows affected (0.01 sec)

Records: 5 Duplicates: 0 Warnings: 0

* **mysql> SELECT \* FROM Bookings;**

**+-----------+--------------+------------+-------------+------------+**

**| BookingID | PassengersID | ScheduleID | BookingDate | SeatNumber |**

**+-----------+--------------+------------+-------------+------------+**

**| 1 | 1 | 1 | 2024-10-10 | 12 |**

**| 2 | 2 | 1 | 2024-10-11 | 34 |**

**| 3 | 3 | 2 | 2024-10-12 | 56 |**

**| 4 | 4 | 3 | 2024-10-13 | 18 |**

**| 5 | 5 | 4 | 2024-10-14 | 22 |**

**+-----------+--------------+------------+-------------+------------+**

# **Task 2: Add Constraints After Data Insertion (Strictly write after data insertion)**

1. **Add a Primary Key to each table.**

**Ans:-**

* **mysql> ALTER TABLE trains ADD CONSTRAINT PK\_Trains PRIMARY KEY (TrainID);**

mysql> desc trains;

+------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+------------+--------------+------+-----+---------+-------+

| TrainID | int | NO | PRI | NULL | |

| TrainName | varchar(100) | YES | | NULL | |

| TraintType | varchar(50) | YES | | NULL | |

| TotalSeats | int | YES | | NULL | |

+------------+--------------+------+-----+---------+-------+

* **mysql> ALTER TABLE routes ADD CONSTRAINT PK\_Routes PRIMARY KEY(RouteID);**

**mysql> DESC ROUTES;**

+--------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------------+--------------+------+-----+---------+-------+

| RouteID | int | NO | PRI | NULL | |

| StartStation | varchar(100) | YES | | NULL | |

| EndStation | varchar(100) | YES | | NULL | |

| Distance | int | YES | | NULL | |

+--------------+--------------+------+-----+---------+-------+

* **mysql> ALTER TABLE Passengers ADD CONSTRAINT PK\_Passengers PRIMARY KEY (PassengerID);**

**Query OK, 0 rows affected (0.08 sec)**

**Records: 0 Duplicates: 0 Warnings: 0**

**mysql> DESC Passengers;**

+-------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------------+--------------+------+-----+---------+-------+

| PassengerID | int | NO | PRI | NULL | |

| FirstName | varchar(50) | YES | | NULL | |

| LastName | varchar(50) | YES | | NULL | |

| Age | int | YES | | NULL | |

| Email | varchar(100) | YES | | NULL | |

+-------------+--------------+------+-----+---------+-------+

* **mysql> ALTER TABLE schedules**

**-> ADD CONSTRAINT PK\_Schedules PRIMARY KEY (ScheduleID);**

**mysql> desc schedules;**

+---------------+----------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+---------------+----------+------+-----+---------+-------+

| ScheduleID | int | NO | PRI | NULL | |

| TrainID | int | YES | | NULL | |

| RouteID | int | YES | | NULL | |

| DepartureTime | datetime | YES | | NULL | |

| ArrivalTime | datetime | YES | | NULL | |

+---------------+----------+------+-----+---------+-------+

* **mysql> ALTER TABLE bookings**

**-> ADD CONSTRAINT PK\_Bookings PRIMARY KEY (BookingID);**

**mysql> desc bookings;**

+--------------+------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------------+------+------+-----+---------+-------+

| BookingID | int | NO | PRI | NULL | |

| PassengersID | int | YES | | NULL | |

| ScheduleID | int | YES | | NULL | |

| BookingDate | date | YES | | NULL | |

| SeatNumber | int | YES | | NULL | |

+--------------+------+------+-----+---------+-------

1. **Add Foreign Keys to establish relationships between: o Schedules and Trains (on TrainID). o Schedules and Routes (on RouteID).**

**o Bookings and Passengers (on PassengerID). o Bookings and Schedules (on ScheduleID).**

**Ans:-**

* **mysql> ALTER TABLE Schedules**

**-> ADD CONSTRAINT FK\_Schedules\_Trains**

**-> FOREIGN KEY (TrainID) REFERENCES Trains(TrainID);**

**mysql> desc schedules;**

+---------------+----------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+---------------+----------+------+-----+---------+-------+

| ScheduleID | int | NO | PRI | NULL | |

| TrainID | int | YES | MUL | NULL | |

| RouteID | int | YES | | NULL | |

| DepartureTime | datetime | YES | | NULL | |

| ArrivalTime | datetime | YES | | NULL | |

+---------------+----------+------+-----+---------+-------+

* **mysql> ALTER TABLE Schedules ADD CONSTRAINT FK\_Schedules\_Routes FOREIGN KEY (RouteID) REFERENCES Routes(RouteID);**

**mysql> DESC SCHEDULES;**

+---------------+----------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+---------------+----------+------+-----+---------+-------+

| ScheduleID | int | NO | PRI | NULL | |

| TrainID | int | YES | MUL | NULL | |

| RouteID | int | YES | MUL | NULL | |

| DepartureTime | datetime | YES | | NULL | |

| ArrivalTime | datetime | YES | | NULL | |

+---------------+----------+------+-----+---------+-------+

* **mysql> ALTER TABLE Bookings**

**-> ADD CONSTRAINT FK\_Bookings\_Passengers**

**-> FOREIGN KEY (PassengersID) REFERENCES Passengers(PassengerID);**

**mysql> desc Bookings;**

+--------------+------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------------+------+------+-----+---------+-------+

| BookingID | int | NO | PRI | NULL | |

| PassengersID | int | YES | MUL | NULL | |

| ScheduleID | int | YES | | NULL | |

| BookingDate | date | YES | | NULL | |

| SeatNumber | int | YES | | NULL | |

+--------------+------+------+-----+---------+-------+

* **mysql> ALTER TABLE Bookings**

**-> ADD CONSTRAINT FK\_Bookings\_Schedules**

**-> FOREIGN KEY (ScheduleID) REFERENCES Schedules(ScheduleID);**

**mysql> desc Bookings;**

+--------------+------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------------+------+------+-----+---------+-------+

| BookingID | int | NO | PRI | NULL | |

| PassengersID | int | YES | MUL | NULL | |

| ScheduleID | int | YES | MUL | NULL | |

| BookingDate | date | YES | | NULL | |

| SeatNumber | int | YES | | NULL | |

+--------------+------+------+-----+---------+-------+

# **Task 3: Joins and Queries**

1. **Query 1: Write a query to retrieve the train name, route details, and schedule for all trains using an INNER JOIN between the Trains, Routes, and Schedules tables.**

**Ans :-**

**mysql> SELECT**

**-> t.TrainName,**

**-> r.StartStation,**

**-> r.EndStation,**

**-> r.Distance,**

**-> s.DepartureTime,**

**-> s.ArrivalTime**

**-> FROM**

**-> Trains t**

**-> INNER JOIN**

**-> Schedules s ON t.TrainID = s.TrainID**

**-> INNER JOIN**

**-> Routes r ON s.RouteID = r.RouteID;**

+------------------+--------------+------------+----------+---------------------+---------------------+

| TrainName | StartStation | EndStation | Distance | DepartureTime | ArrivalTime |

+------------------+--------------+------------+----------+---------------------+---------------------+

| Rajdhani Express | Delhi | Mumbai | 1400 | 2024-10-20 09:00:00 | 2024-10-20 21:00:00 |

| Duronto Express | Delhi | Mumbai | 1400 | 2024-10-23 10:00:00 | 2024-10-23 22:00:00 |

| Tejas Express | Kolkata | Chennai | 1650 | 2024-10-21 08:30:00 | 2024-10-21 22:00:00 |

| Garib Rath | Kolkata | Chennai | 1650 | 2024-10-24 07:00:00 | 2024-10-24 21:00:00 |

| Shatabdi Express | Jaipur | Ahmedabad | 650 | 2024-10-22 06:00:00 | 2024-10-22 14:00:00 |

+------------------+--------------+------------+----------+---------------------+---------------------+

1. **Query 2: Write a query to retrieve all trains that don't have any bookings using a LEFT JOIN between the Trains and Bookings tables.**

**Ans :**

mysql> **SELECT t.TrainID, t.TrainName**

**-> FROM Trains t**

**-> LEFT JOIN Schedules s ON t.TrainID = s.TrainID**

**-> LEFT JOIN Bookings b ON s.ScheduleID = b.ScheduleID**

**-> WHERE b.BookingID IS NULL;**

+---------+------------+

| TrainID | TrainName |

+---------+------------+

| 5 | Garib Rath |

+---------+------------+

1. **Query 3: Write a query to find all passengers who have booked seats for trains traveling a distance of more than 500 km using a RIGHT JOIN and subquery.**

**Ans :**

**mysql> SELECT p.PassengerID, p.FirstName, p.LastName**

**-> FROM Bookings b**

**-> RIGHT JOIN Passengers p ON b.PassengersID = p.PassengerID**

**-> WHERE b.ScheduleID IN (**

**-> SELECT s.ScheduleID**

**-> FROM Schedules s**

**-> JOIN Routes r ON s.RouteID = r.RouteID**

**-> WHERE r.Distance > 500);**

+-------------+-----------+----------+

| PassengerID | FirstName | LastName |

+-------------+-----------+----------+

| 1 | Rajesh | Sharma |

| 2 | Priya | Mehra |

| 5 | Arun | Patel |

| 3 | Ankit | Verma |

| 4 | Kavita | Gupta |

+-------------+-----------+----------+

1. **Query 4: Write a query to list all train schedules, even if there are no passengers booked, using an**

**OUTER JOIN.**

**Ans :-**

**mysql> SELECT s.ScheduleID, s.TrainID, s.DepartureTime, b.BookingID**

**-> FROM Schedules s**

**-> LEFT JOIN Bookings b ON s.ScheduleID = b.ScheduleID**

**->**

**-> UNION**

**->**

**-> -- RIGHT JOIN part**

**-> SELECT s.ScheduleID, s.TrainID, s.DepartureTime, b.BookingID**

**-> FROM Schedules s**

**-> RIGHT JOIN Bookings b ON s.ScheduleID = b.ScheduleID;**

**+------------+---------+---------------------+-----------+**

| ScheduleID | TrainID | DepartureTime | BookingID |

+------------+---------+---------------------+-----------+

| 1 | 1 | 2024-10-20 09:00:00 | 1 |

| 1 | 1 | 2024-10-20 09:00:00 | 2 |

| 2 | 2 | 2024-10-21 08:30:00 | 3 |

| 3 | 3 | 2024-10-22 06:00:00 | 4 |

| 4 | 4 | 2024-10-23 10:00:00 | 5 |

| 5 | 5 | 2024-10-24 07:00:00 | NULL |

+------------+---------+---------------------+-----------+

**Task 4: Normalization**

1. Normalize the tables to the 3rd Normal Form (3NF) to eliminate redundancy and ensure data integrity.

**Ans :- database is already in 3rd Normal Form (3NF). You’ve done everything correctly in your table design!**

# Task 5: Sub Queries

1. Query 5: Write a query to calculate the total number of passengers for each train route.

**Ans :**

**mysql> SELECT**

**-> r.RouteID,**

**-> r.StartStation,**

**-> r.EndStation,**

**-> COUNT(b.BookingID) AS TotalPassengers**

**-> FROM**

**-> Routes r**

**-> JOIN**

**-> Schedules s ON r.RouteID = s.RouteID**

**-> JOIN**

**-> Bookings b ON s.ScheduleID = b.ScheduleID**

**-> GROUP BY**

**-> r.RouteID, r.StartStation, r.EndStation;**

+---------+--------------+------------+-----------------+

| RouteID | StartStation | EndStation | TotalPassengers |

+---------+--------------+------------+-----------------+

| 1 | Delhi | Mumbai | 3 |

| 2 | Kolkata | Chennai | 1 |

| 3 | Jaipur | Ahmedabad | 1 |

+---------+--------------+------------+-----------------+

1. Query 6: Write a query to find the average number of passengers booked per train.

**Ans :-**

mysql> SELECT

-> t.TrainID,

-> t.TrainName,

-> AVG(passenger\_count) AS AvgPassengers

-> FROM

-> Trains t

-> JOIN

-> Schedules s ON t.TrainID = s.TrainID

-> LEFT JOIN (

-> SELECT

-> ScheduleID,

-> COUNT(\*) AS passenger\_count

-> FROM

-> Bookings

-> GROUP BY

-> ScheduleID

-> ) AS booking\_counts ON s.ScheduleID = booking\_counts.ScheduleID

-> GROUP BY

-> t.TrainID, t.TrainName;

+---------+------------------+---------------+

| TrainID | TrainName | AvgPassengers |

+---------+------------------+---------------+

| 1 | Rajdhani Express | 2.0000 |

| 2 | Tejas Express | 1.0000 |

| 3 | Shatabdi Express | 1.0000 |

| 4 | Duronto Express | 1.0000 |

| 5 | Garib Rath | NULL |

1. Query 7: Write a query to find the train with the highest number of bookings.

**Ans :**

**mysql> SELECT**

**-> t.TrainID,**

**-> t.TrainName,**

**-> COUNT(b.BookingID) AS TotalBookings**

**-> FROM**

**-> Trains t**

**-> JOIN**

**-> Schedules s ON t.TrainID = s.TrainID**

**-> JOIN**

**-> Bookings b ON s.ScheduleID = b.ScheduleID**

**-> GROUP BY**

**-> t.TrainID, t.TrainName**

**-> ORDER BY**

**-> TotalBookings DESC**

**-> LIMIT 1;**

+---------+------------------+---------------+

| TrainID | TrainName | TotalBookings |

+---------+------------------+---------------+

| 1 | Rajdhani Express | 2 |

+---------+------------------+---------------+

1. Query 8: Write a query to find the total seats booked per train route where the booking date is between 01-Jan-2023 and 31-Dec-2023.

**Ans :-**

**mysql> SELECT**

**-> r.RouteID,**

**-> r.StartStation,**

**-> r.EndStation,**

**-> COUNT(b.BookingID) AS SeatsBooked**

**-> FROM**

**-> Routes r**

**-> JOIN**

**-> Schedules s ON r.RouteID = s.RouteID**

**-> JOIN**

**-> Bookings b ON s.ScheduleID = b.ScheduleID**

**-> WHERE**

**-> b.BookingDate BETWEEN '2023-01-01' AND '2023-12-31'**

**-> GROUP BY**

**-> r.RouteID, r.StartStation, r.EndStation;**

1. Query 9: Write a query to list all bookings where the passenger's age is greater than 60.

**Ans :**

**mysql> SELECT**

**-> b.BookingID,**

**-> p.PassengerID,**

**-> p.FirstName,**

**-> p.LastName,**

**-> p.Age,**

**-> b.BookingDate,**

**-> b.SeatNumber**

**-> FROM**

**-> Bookings b**

**-> JOIN**

**-> Passengers p ON b.PassengersID = p.PassengerID**

**-> WHERE**

**-> p.Age > 60;**

**Empty set (0.00 sec)**

# Task 6: Stored Procedures and Functions

1. Write a stored procedure to update the number of available seats in a train after a booking has been made.

**Ans :-**

**mysql> DELIMITER //**

**mysql> CREATE PROCEDURE UpdateAvailableSeates(IN tID INT)**

**-> BEGIN**

**-> UPDATE Trains**

**-> SET AvailableSeats = AvailableSeats - 1 WHERE TrainID = tID;**

**-> END //**

**Query OK, 0 rows affected (0.03 sec)**

1. Write a function to calculate the total travel time (in hours) between two stations based on departure and arrival times from the Schedules table.

**Ans :-**

**DETERMINIS' at line 1**

**mysql> DELIMITER //**

**mysql>**

**mysql> CREATE FUNCTION GetTravelTime(depTime TIME, arrTime TIME)**

**-> RETURNS DECIMAL(5,2)**

**-> DETERMINISTIC**

**-> BEGIN**

**-> DECLARE hours DECIMAL(5,2);**

**-> SET hours = TIME\_TO\_SEC(TIMEDIFF(arrTime, depTime)) / 3600;**

**-> RETURN hours;**

**-> END //**

**Query OK, 0 rows affected (0.02 sec)**

# Task 7: Views and Indexes

1. Create a view named PassengerBookingsView that combines passenger details, train information, and booking details in one query.
2. Create an index on the Bookings table to improve the performance of queries filtering by BookingDate.

**SELECT**

**b.BookingID,**

**p.PassengerID,**

**p.FirstName,**

**p.LastName,**

**p.Age,**

**b.BookingDate,**

**b.SeatNumber**

**FROM**

**Bookings b**

**JOIN**

**Passengers p ON b.PassengersID = p.PassengerID**

**WHERE**

**p.Age > 60;**

# **Task 8: Temporary Tables**

1. **Create a temporary table to store the schedule of all trains departing on a specific day (for example, 15-Oct-2023), and then query it.**

**Ans :-**

**mysql> CREATE TEMPORARY TABLE TempTrainSchedule AS**

**-> SELECT**

**-> s.ScheduleID,**

**-> s.TrainID,**

**-> s.RouteID,**

**-> s.DepartureTime,**

**-> s.ArrivalTime**

**-> FROM**

**-> Schedules s**

**-> WHERE**

**-> DATE(s.DepartureTime) = '2023-10-15';**

**Query OK, 0 rows affected (0.02 sec)**

**Records: 0 Duplicates: 0 Warnings: 0**

**mysql> SELECT \* FROM TempTrainSchedule;**

**Empty set (0.00 sec)**

**Task 9: Cursors**

1. **Write a basic cursor to iterate over the passengers who have booked more than 5 tickets.**

**Ans :-**

mysql> DELIMITER $$

mysql>

mysql> CREATE PROCEDURE Cursor\_PassengersWithMoreThan5Bookings()

-> BEGIN

-> -- Declare variables to hold data from the cursor

-> DECLARE done INT DEFAULT 0;

-> DECLARE pass\_id INT;

-> DECLARE fname VARCHAR(100);

-> DECLARE total\_bookings INT;

->

-> -- Declare the cursor

-> DECLARE passenger\_cursor CURSOR FOR

-> SELECT p.PassengerID, p.FirstName, COUNT(b.BookingID) AS total\_bookings

-> FROM Passengers p

-> JOIN Bookings b ON p.PassengerID = b.PassengersID

-> GROUP BY p.PassengerID, p.FirstName

-> HAVING COUNT(b.BookingID) > 5;

->

-> -- Declare the handler for end of data

-> DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;

->

-> -- Open cursor

-> OPEN passenger\_cursor;

->

-> read\_loop: LOOP

-> FETCH passenger\_cursor INTO pass\_id, fname, total\_bookings;

-> IF done THEN

-> LEAVE read\_loop;

-> END IF;

->

-> -- You can replace this with any custom processing logic, here we show output

-> SELECT CONCAT('Passenger ID: ', pass\_id, ', Name: ', fname, ', Total Bookings: ', total\_bookings) AS Info;

-> END LOOP;

->

-> CLOSE passenger\_cursor;

-> END$$

Query OK, 0 rows affected (0.01 sec)

mysql>

mysql> DELIMITER ;

mysql> CALL Cursor\_PassengersWithMoreThan5Bookings();

Query OK, 0 rows affected (0.01 sec)

# **Task 10: ACID Properties and Transactions**

**1. Ensure the database follows ACID properties by using transactions: o Begin a transaction before updating seat availability. o Use a savepoint before deleting any booking records. o Rollback if an error occurs, ensuring no changes are made to the database. o Commit the transaction only after all changes have been successfully applied.**

# **Task 11: Triggers (Understand which trigger to use when and for what)**

1. **Create a Trigger that automatically assigns a seat number to passengers when a booking is created.**
2. **Create a Trigger that updates the total available seats in the train after a booking is confirmed.**

# **Task 12: UNION and UNION ALL**

1. **Write a query to combine the results of two queries that return passengers booked on trains for two different routes.**
2. **Write a query to combine the results of all bookings made on different dates.**

# **Task 13: Copying Tables**

1. **Copy the structure of the Passengers table into a new table named OldPassengers.**
2. **Copy all the data from the Bookings table into another table named ArchivedBookings.**

# **Additional Assignment: Keys Practice (You should know exact difference and how to use them)**

**Duration: 15-20 Minutes**

**Scenario:**

**SpecialForce Private Limited has an Employee Management System that stores employee data. The management system uses various keys to maintain data integrity and ensure fast retrieval of information. You are tasked with identifying and implementing different types of keys in the database schema provided below.**

**Employee Table:**

**EmpID (PK) FirstName LastName Email PhoneNumber Department**

1. **Rajesh Sharma rajesh@specialforce.com 9876543210 HR**
2. **Priya Mehra priya@specialforce.com 8765432109 Finance**
3. **Ankit Verma ankit@specialforce.com 7654321098 IT 104 Kavita Gupta kavita@specialforce.com 6543210987 IT**

**105 Suresh Nair suresh@specialforce.com 5432109876 Sales**

**Assignment Tasks:**

1. **Super Key:**

**Identify all possible Super Keys from the Employee table. Remember, a Super Key is any combination of columns that uniquely identifies each record.**

1. **Candidate Key:**

**Determine which of the Super Keys qualify as Candidate Keys. A Candidate Key is a minimal Super Key, meaning it contains no unnecessary columns.**

1. **Primary Key:**

**What is the Primary Key for the Employee table? Explain why this key was chosen as the Primary Key.**

1. **Alternate Key:**

**Identify any Alternate Keys. Alternate Keys are Candidate Keys that were not chosen as the Primary Key.**

1. **Composite Key:**

**Suppose the company wants to track employees working on multiple projects, where both the EmpID and**

**ProjectID are needed to uniquely identify records in a new EmployeeProjects table. Define a Composite Key for this table using EmpID and ProjectID.**

**EmployeeProjects Table:**

**EmpID ProjectID**

* 1. **P01**
  2. **P02**
  3. **P03**
  4. **P01**
  5. **P03**