```java
import java.sql.*;
import java.util.Scanner;

public class PomonaTransitSystem {
    static String url = "jdbc:mysql://localhost:3306/LAB4";
    static String username = "root";
    static String password = "clarissa";
    static Connection connection = null;
    static Statement statement = null;

    public static ResultSet executeStatement(String sql) throws SQLException {
        statement = connection.createStatement();
        return statement.executeQuery(sql);
    }

    // executeUpate helper method
    public static void executeUpdate(String sql) {
        try {
            // Assuming you have a Connection object called "conn"
            Statement stmt = connection.createStatement();
            stmt.executeUpdate(sql);
            stmt.close();
        } catch (SQLException e) {
            System.out.println("Database error: " + e.getMessage());
        }
    }


    public static void dropAllTables() throws SQLException {
        try (Statement stmt = connection.createStatement()) {
            // disable FK checks
            stmt.executeUpdate("SET FOREIGN_KEY_CHECKS = 0;");
            stmt.executeUpdate("DROP TABLE IF EXISTS ActualTripStopInfo;");
            stmt.executeUpdate("DROP TABLE IF EXISTS TripStopInfo;");
            stmt.executeUpdate("DROP TABLE IF EXISTS TripOffering;");
            stmt.executeUpdate("DROP TABLE IF EXISTS Stop;");
            stmt.executeUpdate("DROP TABLE IF EXISTS Driver;");
            stmt.executeUpdate("DROP TABLE IF EXISTS Bus;");
            stmt.executeUpdate("DROP TABLE IF EXISTS Trip;");
            // re-enable FK checks
            stmt.executeUpdate("SET FOREIGN_KEY_CHECKS = 1;");
            System.out.println("All tables dropped.");
        }
    }



    public static void createTables() throws SQLException {
        try (Statement stmt = connection.createStatement()) {
            stmt.executeUpdate(
                    "CREATE TABLE IF NOT EXISTS Trip (" +
                            "  TripNumber INT PRIMARY KEY," +
                            "  StartLocationName VARCHAR(100)," +
                            "  DestinationName VARCHAR(100)" +
                            ");");
            stmt.executeUpdate(
                    "CREATE TABLE IF NOT EXISTS Bus (" +
                            "  BusID INT PRIMARY KEY," +
                            "  Model VARCHAR(100)," +
                            "  Year INT" +
                            ");");
            stmt.executeUpdate(
                    "CREATE TABLE IF NOT EXISTS Driver (" +
                            "  DriverName VARCHAR(100) PRIMARY KEY," +
                            "  DriverTelephoneNumber VARCHAR(20)" +
                            ");");
            stmt.executeUpdate(
                    "CREATE TABLE IF NOT EXISTS Stop (" +
                            "  StopNumber INT PRIMARY KEY," +
                            "  StopAddress VARCHAR(200)" +
                            ");");

            stmt.executeUpdate(
                    "CREATE TABLE IF NOT EXISTS TripOffering (" +
                            "  TripNumber INT," +
                            "  Date DATE," +
                            "  ScheduledStartTime TIME," +
                            "  ScheduledArrivalTime TIME," +
                            "  DriverName VARCHAR(100)," +
                            "  BusID INT," +
                            "  PRIMARY KEY (TripNumber, Date, ScheduledStartTime),"
+
                            "  FOREIGN KEY (TripNumber) REFERENCES
Trip(TripNumber)," +
                            "  FOREIGN KEY (DriverName) REFERENCES
Driver(DriverName)," +
                            "  FOREIGN KEY (BusID) REFERENCES Bus(BusID)" +
                            ");");
            stmt.executeUpdate(
                    "CREATE TABLE IF NOT EXISTS TripStopInfo (" +
                            "  TripNumber INT," +
                            "  StopNumber INT," +
                            "  SequenceNumber INT," +
                            "  DrivingTime INT," +
                            "  PRIMARY KEY (TripNumber, StopNumber)," +
                            "  FOREIGN KEY (TripNumber) REFERENCES
Trip(TripNumber)," +
                            "  FOREIGN KEY (StopNumber) REFERENCES
Stop(StopNumber)" +
                            ");");
            stmt.executeUpdate(
                    "CREATE TABLE IF NOT EXISTS ActualTripStopInfo (" +
                            "  TripNumber INT," +
                            "  Date DATE," +
                            "  ScheduledStartTime TIME," +
                            "  StopNumber INT," +
                            "  ScheduledArrivalTime TIME," +
                            "  ActualStartTime TIME," +
                            "  ActualArrivalTime TIME," +
                            "  NumberOfPassengerIn INT," +
                            "  NumberOfPassengerOut INT," +
                            "  PRIMARY KEY (TripNumber, Date, ScheduledStartTime,
StopNumber)," +
                            "  FOREIGN KEY (TripNumber, Date, ScheduledStartTime) "
+
                            "    REFERENCES TripOffering(TripNumber, Date,
ScheduledStartTime) ON DELETE CASCADE," +
                            "  FOREIGN KEY (StopNumber) REFERENCES
```

```
Stop(StopNumber)" +
                            ");");
            System.out.println("All tables created (if not existed).");
        }
    }

    public static void populateDummyData() throws SQLException {
        try (Statement stmt = connection.createStatement()) {
            // Trip
            stmt.executeUpdate(
                    "INSERT INTO Trip (TripNumber, StartLocationName,
DestinationName) VALUES " +
                            "(101, 'City A', 'City B'), " +
                            "(102, 'City C', 'City D');");
            // Bus
            stmt.executeUpdate(
                    "INSERT INTO Bus (BusID, Model, Year) VALUES " +
                            "(1, 'Volvo 9700', 2019), " +
                            "(2, 'Scania Touring', 2020);");
            // Driver
            stmt.executeUpdate(
                    "INSERT INTO Driver (DriverName, DriverTelephoneNumber) VALUES
" +
                            "('John Doe', '555-1234'), " +
                            "('Jane Smith', '555-5678');");
            // Stop
            stmt.executeUpdate(
                    "INSERT INTO Stop (StopNumber, StopAddress) VALUES " +
                            "(1, '123 Main St'), " +
                            "(2, '456 Elm St'), " +
                            "(3, '789 Oak St'), " +
                            "(4, '101 Maple Ave');");
            // TripOffering
            stmt.executeUpdate(
                    "INSERT INTO TripOffering " +
                            "(TripNumber, Date, ScheduledStartTime,
ScheduledArrivalTime, DriverName, BusID) VALUES " +
                            "(101, '2025-05-01', '08:00:00', '10:00:00', 'John
Doe', 1), " +
                            "(102, '2025-05-02', '09:00:00', '11:30:00', 'Jane
Smith', 2);");
            // ActualTripStopInfo
            stmt.executeUpdate(
                    "INSERT INTO ActualTripStopInfo " +
                            "(TripNumber, Date, ScheduledStartTime, StopNumber,
ScheduledArrivalTime, ActualStartTime, ActualArrivalTime, NumberOfPassengerIn,
NumberOfPassengerOut) VALUES "
                            +
                            "(101, '2025-05-01', '08:00:00', 1, '08:15:00',
'08:05:00', '08:20:00', 5, 0), " +
                            "(101, '2025-05-01', '08:00:00', 2, '08:30:00',
'08:25:00', '08:35:00', 3, 1), " +
                            "(102, '2025-05-02', '09:00:00', 3, '09:20:00',
'09:05:00', '09:25:00', 7, 2), " +
                            "(102, '2025-05-02', '09:00:00', 4, '09:40:00',
'09:30:00', '09:45:00', 4, 3);");
            // TripStopInfo
            stmt.executeUpdate(
                    "INSERT INTO TripStopInfo (TripNumber, StopNumber,
SequenceNumber, DrivingTime) VALUES " +
                            "(101, 1, 1, 15), " +
                            "(101, 2, 2, 15), " +
                            "(102, 3, 1, 20), " +
                            "(102, 4, 2, 20);");
            System.out.println("Dummy data populated successfully.");
        }
    }
    /*
     * Helper Functions END
     */

    public static void displaySchedule() {
        Scanner scanner = new Scanner(System.in);
        try {
            // Get user input
            System.out.print("Enter Start Location Name: ");
            String startLocation = scanner.nextLine();

            System.out.print("Enter Destination Name: ");
            String destination = scanner.nextLine();

            System.out.print("Enter Date (YYYY-MM-DD): ");
            String date = scanner.nextLine();

            // Build the SQL query
            String sql = "SELECT tf.ScheduledStartTime, tf.ScheduledArrivalTime, 
tf.DriverName, tf.BusID " +
                    "FROM TripOffering tf " +
                    "JOIN Trip t ON tf.TripNumber = t.TripNumber " +
                    "WHERE t.StartLocationName = '" + startLocation + "' " +
                    "AND t.DestinationName = '" + destination + "' " +
                    "AND tf.Date = '" + date + "';";

            // Execute and get results
            ResultSet rs = executeStatement(sql);

            boolean found = false;
            System.out.println("\nSchedule:");
            System.out.printf("%-15s %-20s %-20s %-10s\n", "Start Time", "Arrival 
Time", "Driver Name", "BusID");

System.out.println("-----------------------------------------------------------------
--------");

            while (rs.next()) {
                String startTime = rs.getString("ScheduledStartTime");
                String arrivalTime = rs.getString("ScheduledArrivalTime");
                String driverName = rs.getString("DriverName");
                int busId = rs.getInt("BusID");

                System.out.printf("%-15s %-20s %-20s %-10d\n", startTime, 
arrivalTime, driverName, busId);
                found = true;
            }

            if (!found) {
                System.out.println("No trips found for the given Start Location, 
Destination, and Date.");
```

```
            }

        } catch (SQLException e) {
            System.out.println("Error retrieving trip schedule: " +
e.getMessage());
        }
    }

    public static void editSchedule() {
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("\n--- Edit Trip Offerings ---");
            System.out.println("1. Delete a Trip Offering");
            System.out.println("2. Add Trip Offerings");
            System.out.println("3. Change Driver for a Trip Offering");
            System.out.println("4. Change Bus for a Trip Offering");
            System.out.println("5. Exit to main menu");
            System.out.print("Enter your choice: ");
            String choice = scanner.nextLine();

            switch (choice) {
                case "1":
                    // Delete a Trip Offering
                    try {
                        System.out.print("Enter Trip Number: ");
                        String tripNumber = scanner.nextLine();

                        System.out.print("Enter Date (yyyy-mm-dd): ");
                        String date = scanner.nextLine();

                        System.out.print("Enter Scheduled Start Time (hh:mm:ss):
");

                        String startTime = scanner.nextLine();

                        String sql = "DELETE FROM TripOffering " +
                                "WHERE TripNumber = '" + tripNumber + "' " +
                                "AND Date = '" + date + "' " +
                                "AND ScheduledStartTime = '" + startTime + "';";

                        executeUpdate(sql);
                        System.out.println("Trip offering deleted successfully.");

                    } catch (Exception e) {
                        System.out.println("Error deleting trip offering: " +
e.getMessage());
                    }
                    break;

                case "2":
                    // Add Trip Offerings
                    try {
                        while (true) {
                            System.out.print("Enter Trip Number: ");
                            String tripNumber = scanner.nextLine();

                            System.out.print("Enter Date (yyyy-mm-dd): ");
                            String date = scanner.nextLine();
                            System.out.print("Enter Scheduled Start Time
(hh:mm:ss): ");
                            String startTime = scanner.nextLine();

                            System.out.print("Enter Scheduled Arrival Time
(hh:mm:ss): ");
                            String arrivalTime = scanner.nextLine();

                            System.out.print("Enter Driver Name: ");
                            String driverName = scanner.nextLine();

                            System.out.print("Enter Bus ID: ");
                            String busID = scanner.nextLine();

                            String sql = "INSERT INTO TripOffering (TripNumber,
Date, ScheduledStartTime, ScheduledArrivalTime, DriverName, BusID) " +
                                    "VALUES ('" + tripNumber + "', '" + date + "',
'" + startTime + "', '" + arrivalTime + "', '" + driverName + "', '" + busID +
"');";

                            executeUpdate(sql);
                            System.out.println("Trip offering added
successfully.");

                            System.out.print("Do you want to add another trip?
(y/n): ");
                            String another = scanner.nextLine();
                            if (!another.equalsIgnoreCase("y")) {
                                break;
                            }
                        }
                    } catch (Exception e) {
                        System.out.println("Error adding trip offering: " +
e.getMessage());
                    }
                    break;

                case "3":
                    // Change Driver
                    try {
                        System.out.print("Enter Trip Number: ");
                        String tripNumber = scanner.nextLine();

                        System.out.print("Enter Date (yyyy-mm-dd): ");
                        String date = scanner.nextLine();

                        System.out.print("Enter Scheduled Start Time (hh:mm:ss):
");
                        String startTime = scanner.nextLine();

                        System.out.print("Enter New Driver Name: ");
                        String newDriver = scanner.nextLine();

                        String sql = "UPDATE TripOffering " +
                                "SET DriverName = '" + newDriver + "' " +
                                "WHERE TripNumber = '" + tripNumber + "' " +
                                "AND Date = '" + date + "' " +
                                "AND ScheduledStartTime = '" + startTime + "';";
                        executeUpdate(sql);
                        System.out.println("Driver updated successfully.");

                    } catch (Exception e) {
                        System.out.println("Error updating driver: " +
e.getMessage());
                    }
                    break;

                case "4":
                    // Change Bus
                    try {
                        System.out.print("Enter Trip Number: ");
                        String tripNumber = scanner.nextLine();

                        System.out.print("Enter Date (yyyy-mm-dd): ");
                        String date = scanner.nextLine();

                        System.out.print("Enter Scheduled Start Time (hh:mm:ss):
");

                        String startTime = scanner.nextLine();

                        System.out.print("Enter New Bus ID: ");
                        String newBusID = scanner.nextLine();

                        String sql = "UPDATE TripOffering " +
                                "SET BusID = '" + newBusID + "' " +
                                "WHERE TripNumber = '" + tripNumber + "' " +
                                "AND Date = '" + date + "' " +
                                "AND ScheduledStartTime = '" + startTime + "';";

                        executeUpdate(sql);
                        System.out.println("Bus updated successfully.");

                    } catch (Exception e) {
                        System.out.println("Error updating bus: " +
e.getMessage());
                    }
                    break;

                case "5":
                    // Exit to main menu
                    System.out.println("Returning to main menu.");
                    return;

                default:
                    System.out.println("Invalid choice. Please select 1-5.");
            }
        }
    }

    public static void displayStops() {
        Scanner scanner = new Scanner(System.in);

        try {
            System.out.print("Enter Trip Number: ");
            String tripNumber = scanner.nextLine();

            String sql = "SELECT * FROM TripStopInfo WHERE TripNumber = '" +
tripNumber + "' ORDER BY SequenceNumber";

            Statement stmt = connection.createStatement();
            ResultSet rs = stmt.executeQuery(sql);

            boolean found = false;
            System.out.println("\nStops for Trip Number " + tripNumber + ":");
            System.out.println("-------------------------------------------");

            while (rs.next()) {
                found = true;
                int sequenceNumber = rs.getInt("SequenceNumber");
                int stopNumber = rs.getInt("StopNumber");
                String drivingTime = rs.getString("DrivingTime");

                System.out.println("Sequence Number: " + sequenceNumber);
                System.out.println("Stop Number: " + stopNumber);
                System.out.println("Driving Time: " + drivingTime);
                System.out.println("-------------------------------------------");
            }

            if (!found) {
                System.out.println("No stops found for Trip Number " + tripNumber);
            }

            rs.close();
            stmt.close();
        } catch (SQLException e) {
            System.out.println("Database error: " + e.getMessage());
        }
    }


    public static void weeklySchedule() {
        Scanner kb = new Scanner(System.in);
        System.out.print("Please enter Driver Name (First and last name: ");
        String driverName = kb.nextLine().trim();
        System.out.print("Please enter Date (YYYY-MM-DD): ");
        String date = kb.nextLine().trim();

        try {
            Statement stmt = connection.createStatement();
            String sql = "SELECT * " +
                    "FROM TripOffering " +
                    "WHERE DriverName = '" + driverName + "' " +
                    "AND Date >= '" + date + "' " +
                    "AND Date <= DATE_ADD('" + date + "', INTERVAL 6 DAY) " +
                    "ORDER BY Date ASC, ScheduledStartTime ASC";

            ResultSet rs = stmt.executeQuery(sql);
            ResultSetMetaData rsMetaData = rs.getMetaData();
            String varColNames = "";
            int varColCount = rsMetaData.getColumnCount();

            // Print column headers
            for (int col = 1; col <= varColCount; col++) {
                varColNames += rsMetaData.getColumnName(col) + "\t";
            }
            System.out.println(varColNames);
            boolean hasResults = false;

            // Print rows
            while (rs.next()) {
                hasResults = true;
                for (int col = 1; col <= varColCount; col++) {
                    String str = String.format("%-20s", rs.getString(col));
                    System.out.print(str);
                }
                System.out.println();
            }

            if (!hasResults) {
                System.out.println("No scheduled trips found for driver " +
driverName + " starting from " + date);
            }

            rs.close();
            stmt.close();

        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("Error retrieving weekly schedule for driver " +
driverName);
        }
    }


    public static void addDriver() {
        Scanner kb = new Scanner(System.in);

        System.out.print("Enter Driver Name: ");
        String driverName = kb.nextLine().trim();

        System.out.print("Enter Driver Telephone Number: ");
        String driverTelephoneNumber = kb.nextLine().trim();

        try {
            Statement stmt = connection.createStatement();
            String sql = "INSERT INTO Driver (DriverName, DriverTelephoneNumber) "
+
                    "VALUES ('" + driverName + "', '" + driverTelephoneNumber +
"')";

            int rowsInserted = stmt.executeUpdate(sql);

            if (rowsInserted > 0) {
                System.out.println("Driver successfully added!");
            } else {
                System.out.println("Failed to add driver.");
            }

            stmt.close();
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("Error inserting new driver.");
        }
    }
    public static void addBus() {
        Scanner kb = new Scanner(System.in);

        try {
            Statement stmt = connection.createStatement();

            System.out.print("Enter Bus ID: ");
            String busID = kb.nextLine().trim();

            System.out.print("Enter Bus Model: ");
            String model = kb.nextLine().trim();

            System.out.print("Enter Bus Year: ");
            String year = kb.nextLine().trim();

            String sql = "INSERT INTO Bus (BusID, Model, Year) VALUES ('" + busID +
"', '" + model + "', '" + year + "')";

            int rowsInserted = stmt.executeUpdate(sql);

            if (rowsInserted > 0) {
                System.out.println("Bus added successfully.");
            } else {
                System.out.println("Failed to add bus.");
            }
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("Error occurred while adding bus.");
        }
    }


    public static void deleteBus() {
        Scanner kb = new Scanner(System.in);

        System.out.print("Enter the Bus ID to delete: ");
        String busID = kb.nextLine().trim();

        try {
            Statement stmt = connection.createStatement();
            // First delete any TripOffering records that reference the BusID
            String deleteTripOfferingsSQL = "DELETE FROM TripOffering WHERE BusID =
'" + busID + "'";
            int rowsDeleted = stmt.executeUpdate(deleteTripOfferingsSQL);

            if (rowsDeleted > 0) {
                System.out.println("Deleted " + rowsDeleted + " trip offerings
related to bus ID " + busID);
            }

            // Now delete the bus itself
            String deleteBusSQL = "DELETE FROM Bus WHERE BusID = '" + busID + "'";
            int busRowsDeleted = stmt.executeUpdate(deleteBusSQL);

            if (busRowsDeleted > 0) {
                System.out.println("Bus with ID " + busID + " has been deleted
successfully.");
            } else {
                System.out.println("No bus found with ID " + busID + ".");
            }

        } catch (SQLException e) {
            e.printStackTrace();
            System.out.println("Error occurred while deleting bus.");
        }
    }


    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection = DriverManager.getConnection(url, username, password);

            Runtime.getRuntime().addShutdownHook(new Thread(() -> {
                try {
                    dropAllTables();
                    if (connection != null && !connection.isClosed()) {
                        connection.close();
                    }
                } catch (SQLException e) {
                    System.err.println("Error during shutdown: " + e.getMessage());
                }
            }));

            createTables();
            populateDummyData();

            System.out.println("Welcome to Pomona Transit System!");

            int choice = 0;
            boolean invalidInput;

            do {
                invalidInput = true;

                System.out.println("\nPlease enter your desired choice:");
                System.out.println("1. View all trips based on a Source,
Destination, and Date");
                System.out.println("2. Edit a schedule of a Trip Offering");
                System.out.println("3. Display the stops of a given trip");
                System.out.println("4. Display the weekly schedule of a given
driver");
                System.out.println("5. Add a driver");
                System.out.println("6. Add a bus");
                System.out.println("7. Delete");
                System.out.println("8. Exit");

                System.out.print("Enter your choice: ");
                choice = scan.nextInt();

                if (choice <= 8 && choice >= 1) {
                    invalidInput = false;
                } else {
                    System.out.println("Invalid input. Please try again.");
                }
                // Process the selected choice
                switch (choice) {
                    case 1:
                        displaySchedule();
                        break;
                    case 2:
                        editSchedule();
                        break;
                    case 3:
                        displayStops();
                        break;
                    case 4:
                        weeklySchedule();
                        break;
                    case 5:
                        addDriver();
                        break;
                    case 6:
                        addBus();
                        break;
                    case 7:
                        deleteBus();
                        break;
                    case 8:
                        System.out.println("Thank you!");
                        break;
                }
            } while (choice != 8); // Continue until user chooses to exit

        } catch (ClassNotFoundException e) {
            System.err.println("JDBC driver not found: " + e.getMessage());
        } catch (SQLException e) {
            System.err.println("SQL error: " + e.getMessage());
        } finally {
            scan.close();
        }
    }
}
```