1.Write a Java program to connect to a MySQL database using JDBC.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class MySQLJDBCConnection {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    Connection connection = null;

    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

      if (connection != null) {
        System.out.println("Connected to the database successfully!");
      } else {
        System.out.println("Failed to connect to the database.");
      }

    } catch (ClassNotFoundException e) {
      System.out.println("MySQL JDBC Driver not found.");
      e.printStackTrace();
    } catch (SQLException e) {
      System.out.println("SQL exception occurred while connecting.");
      e.printStackTrace();
    } finally {
      try {
```

```
            if (connection != null && !connection.isClosed()) {
                connection.close();
                System.out.println("Database connection closed.");
            }
        } catch (SQLException ex) {
            ex.printStackTrace();
        }
    }
  }
}
```

2.Create a Java class to insert student records into a database table.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class StudentDAO {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public void insertStudent(String name, int age, String email) {
    Connection connection = null;
    PreparedStatement statement = null;

    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
```

```
        String sql = "INSERT INTO students (name, age, email) VALUES (?, ?, ?)";
        statement = connection.prepareStatement(sql);
        statement.setString(1, name);
        statement.setInt(2, age);
        statement.setString(3, email);

        int rowsInserted = statement.executeUpdate();
        if (rowsInserted > 0) {
            System.out.println("Student inserted successfully!");
        }

    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if (statement != null) statement.close();
            if (connection != null && !connection.isClosed()) connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
  }

  public static void main(String[] args) {
    StudentDAO dao = new StudentDAO();
    dao.insertStudent("Alice Johnson", 20, "alice@example.com");
  }
}
```

3.Write a JDBC program to fetch and display all student records from the database.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.SQLException;

public class FetchStudents {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    Connection connection = null;
    Statement statement = null;
    ResultSet resultSet = null;

    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

      String query = "SELECT * FROM students";
      statement = connection.createStatement();
      resultSet = statement.executeQuery(query);

      System.out.println("ID\tName\t\tAge\tEmail");
      System.out.println("-------------------------------------------------");

      while (resultSet.next()) {
        int id = resultSet.getInt("id");
        String name = resultSet.getString("name");
```

```
            int age = resultSet.getInt("age");
            String email = resultSet.getString("email");

            System.out.printf("%d\t%-15s\t%d\t%s%n", id, name, age, email);
        }

    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if (resultSet != null) resultSet.close();
            if (statement != null) statement.close();
            if (connection != null && !connection.isClosed()) connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
  }
}
```

4.Develop a program to search a student by ID using JDBC.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class SearchStudentById {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
```

```
static final String USER = "your_username";
static final String PASSWORD = "your_password";

public static void main(String[] args) {
  int searchId = 1;

  Connection connection = null;
  PreparedStatement statement = null;
  ResultSet resultSet = null;

  try {
    Class.forName("com.mysql.cj.jdbc.Driver");
    connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

    String sql = "SELECT * FROM students WHERE id = ?";
    statement = connection.prepareStatement(sql);
    statement.setInt(1, searchId);
    resultSet = statement.executeQuery();

    if (resultSet.next()) {
      int id = resultSet.getInt("id");
      String name = resultSet.getString("name");
      int age = resultSet.getInt("age");
      String email = resultSet.getString("email");

      System.out.println("Student Found:");
      System.out.println("ID: " + id);
      System.out.println("Name: " + name);
      System.out.println("Age: " + age);
      System.out.println("Email: " + email);
    } else {
      System.out.println("No student found with ID: " + searchId);
```

```
        }

    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if (resultSet != null) resultSet.close();
            if (statement != null) statement.close();
            if (connection != null && !connection.isClosed()) connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
  }
}
```

5.Implement an update operation to modify student details in the database using JDBC.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class UpdateStudent {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    int studentId = 1; // ID of the student to update
```

```
String newName = "Updated Name";
int newAge = 21;
String newEmail = "updatedemail@example.com";

Connection connection = null;
PreparedStatement statement = null;

try {
    Class.forName("com.mysql.cj.jdbc.Driver");
    connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

    String sql = "UPDATE students SET name = ?, age = ?, email = ? WHERE id = ?";
    statement = connection.prepareStatement(sql);
    statement.setString(1, newName);
    statement.setInt(2, newAge);
    statement.setString(3, newEmail);
    statement.setInt(4, studentId);

    int rowsUpdated = statement.executeUpdate();

    if (rowsUpdated > 0) {
        System.out.println("Student record updated successfully.");
    } else {
        System.out.println("No student found with ID: " + studentId);
    }

} catch (ClassNotFoundException | SQLException e) {
    e.printStackTrace();
} finally {
    try {
        if (statement != null) statement.close();
        if (connection != null && !connection.isClosed()) connection.close();
```

```
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
  }
}
```

6.Write a Java program to delete a student record from the database using JDBC.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class DeleteStudent {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    int studentId = 1;

    Connection connection = null;
    PreparedStatement statement = null;

    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

      String sql = "DELETE FROM students WHERE id = ?";
```

```
            statement = connection.prepareStatement(sql);
            statement.setInt(1, studentId);

            int rowsDeleted = statement.executeUpdate();

            if (rowsDeleted > 0) {
                System.out.println("Student record deleted successfully.");
            } else {
                System.out.println("No student found with ID: " + studentId);
            }

        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                if (statement != null) statement.close();
                if (connection != null && !connection.isClosed()) connection.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

7.Design a Java application to perform all CRUD (Create, Read, Update, Delete) operations on an Employee table using JDBC.

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

public class EmployeeCRUD {
```

```
static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
static final String USER = "your_username";
static final String PASSWORD = "your_password";

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

    while (true) {
        System.out.println("\n=== Employee Management ===");
        System.out.println("1. Add Employee");
        System.out.println("2. View All Employees");
        System.out.println("3. Update Employee");
        System.out.println("4. Delete Employee");
        System.out.println("5. Exit");
        System.out.print("Choose an option: ");
        int choice = scanner.nextInt();
        scanner.nextLine();

        switch (choice) {
            case 1:
                System.out.print("Name: ");
                String name = scanner.nextLine();
                System.out.print("Position: ");
                String position = scanner.nextLine();
                System.out.print("Salary: ");
                double salary = scanner.nextDouble();
                app.createEmployee(name, position, salary);
                break;
            case 2:
                app.readEmployees();
                break;
```

```
        case 3:
          System.out.print("Employee ID to update: ");
          int idToUpdate = scanner.nextInt();
          scanner.nextLine();
          System.out.print("New Name: ");
          String newName = scanner.nextLine();
          System.out.print("New Position: ");
          String newPosition = scanner.nextLine();
          System.out.print("New Salary: ");
          double newSalary = scanner.nextDouble();
          app.updateEmployee(idToUpdate, newName, newPosition, newSalary);
          break;
        case 4:
          System.out.print("Employee ID to delete: ");
          int idToDelete = scanner.nextInt();
          app.deleteEmployee(idToDelete);
          break;
        case 5:
          System.out.println("Exiting...");
          return;
        default:
          System.out.println("Invalid choice.");
      }
    }
}

public void createEmployee(String name, String position, double salary) {
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD)) {
      String sql = "INSERT INTO employee (name, position, salary) VALUES (?, ?, ?)";
      try (PreparedStatement stmt = conn.prepareStatement(sql)) {
        stmt.setString(1, name);
        stmt.setString(2, position);
```

```
            stmt.setDouble(3, salary);
            int rows = stmt.executeUpdate();
            System.out.println(rows > 0 ? "Employee added." : "Failed to add employee.");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public void readEmployees() {
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD)) {
        String sql = "SELECT * FROM employee";
        try (Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(sql)) {
            System.out.println("\nID\tName\t\tPosition\tSalary");
            System.out.println("-----------------------------------------------");
            while (rs.next()) {
                int id = rs.getInt("id");
                String name = rs.getString("name");
                String position = rs.getString("position");
                double salary = rs.getDouble("salary");
                System.out.printf("%d\t%-10s\t%-10s\t%.2f%n", id, name, position, salary);
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public void updateEmployee(int id, String name, String position, double salary) {
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD)) {
        String sql = "UPDATE employee SET name = ?, position = ?, salary = ? WHERE id = ?";
```

```
        try (PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setString(1, name);
            stmt.setString(2, position);
            stmt.setDouble(3, salary);
            stmt.setInt(4, id);
            int rows = stmt.executeUpdate();
            System.out.println(rows > 0 ? "Employee updated." : "Employee not found.");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
  }

  public void deleteEmployee(int id) {
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD)) {
        String sql = "DELETE FROM employee WHERE id = ?";
        try (PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setInt(1, id);
            int rows = stmt.executeUpdate();
            System.out.println(rows > 0 ? "Employee deleted." : "Employee not found.");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
  }
}
```

8.Create a JDBC-based program to count the total number of rows in a table.

```
import java.sql.Connection;
import java.sql.DriverManager;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class RowCounter {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    String tableName = "employee";

    Connection connection = null;
    PreparedStatement statement = null;
    ResultSet resultSet = null;

    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

      String sql = "SELECT COUNT(*) AS total FROM " + tableName;
      statement = connection.prepareStatement(sql);
      resultSet = statement.executeQuery();

      if (resultSet.next()) {
        int count = resultSet.getInt("total");
        System.out.println("Total rows in '" + tableName + "' table: " + count);
      }

    } catch (ClassNotFoundException | SQLException e) {
      e.printStackTrace();
```

```
    } finally {
        try {
            if (resultSet != null) resultSet.close();
            if (statement != null) statement.close();
            if (connection != null && !connection.isClosed()) connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
  }
}
```

9.Develop a program to sort student data in ascending order by name using SQL in JDBC.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.SQLException;

public class SortStudentsByName {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    Connection connection = null;
    Statement statement = null;
    ResultSet resultSet = null;
```

```
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

            String sql = "SELECT * FROM students ORDER BY name ASC";
            statement = connection.createStatement();
            resultSet = statement.executeQuery(sql);

            System.out.println("ID\tName\t\tAge\tEmail");
            System.out.println("-----------------------------------------------");

            while (resultSet.next()) {
                int id = resultSet.getInt("id");
                String name = resultSet.getString("name");
                int age = resultSet.getInt("age");
                String email = resultSet.getString("email");

                System.out.printf("%d\t%-15s\t%d\t%s%n", id, name, age, email);
            }
        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                if (resultSet != null) resultSet.close();
                if (statement != null) statement.close();
                if (connection != null && !connection.isClosed()) connection.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

10.Write a program to display all students whose percentage is greater than 75 using JDBC and SQL WHERE clause.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.SQLException;

public class StudentsAbove75Percent {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    Connection connection = null;
    Statement statement = null;
    ResultSet resultSet = null;

    try {
      Class.forName("com.mysql.cj.jdbc.Driver");
      connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);

      String sql = "SELECT * FROM students WHERE percentage > 75";
      statement = connection.createStatement();
      resultSet = statement.executeQuery(sql);

      System.out.println("ID\tName\t\tPercentage");
      System.out.println("---------------------------------");

      while (resultSet.next()) {
        int id = resultSet.getInt("id");
        String name = resultSet.getString("name");
```

```
            double percentage = resultSet.getDouble("percentage");

            System.out.printf("%d\t%-15s\t%.2f%n", id, name, percentage);
        }

    } catch (ClassNotFoundException | SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            if (resultSet != null) resultSet.close();
            if (statement != null) statement.close();
            if (connection != null && !connection.isClosed()) connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
  }
}
```

11.Use PreparedStatement to insert multiple student records into the database.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class InsertMultipleStudents {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";
```

```
    public static void main(String[] args) {

        String sql = "INSERT INTO students (name, age, email) VALUES (?, ?, ?)";
        Object[][] students = {
            {"Alice", 20, "alice@example.com"},
            {"Bob", 22, "bob@example.com"},
            {"Charlie", 19, "charlie@example.com"}
        };

        try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            for (Object[] student : students) {
                pstmt.setString(1, (String) student[0]);
                pstmt.setInt(2, (int) student[1]);
                pstmt.setString(3, (String) student[2]);
                pstmt.addBatch();
            }

            int[] counts = pstmt.executeBatch();

            System.out.println(counts.length + " student records inserted successfully.");

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

12.Implement a program using transaction management in JDBC (i.e., commit and rollback).

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;

public class JDBCTransactionExample {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    String sql = "INSERT INTO students (name, age, email) VALUES (?, ?, ?)";

    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
        PreparedStatement pstmt = conn.prepareStatement(sql)) {

      conn.setAutoCommit(false);

      pstmt.setString(1, "John Doe");
      pstmt.setInt(2, 23);
      pstmt.setString(3, "john@example.com");
      pstmt.executeUpdate();

      pstmt.setString(1, "Jane Smith");
      pstmt.setInt(2, 21);
      pstmt.setString(3, "jane@example.com");
      pstmt.executeUpdate();

      conn.commit();
      System.out.println("Transaction committed successfully.");
```

```
    } catch (SQLException e) {
      System.out.println("Exception occurred, rolling back transaction...");
      e.printStackTrace();
      try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD)) {
        if (conn != null) {
          conn.rollback();
          System.out.println("Transaction rolled back.");
        }
      } catch (SQLException ex) {
        ex.printStackTrace();
      }
    }
  }
}
```

13.Write a JDBC program to handle exceptions (like invalid ID, connection errors) gracefully.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class StudentFetcherWithExceptionHandling {
  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    int searchId = 10;
```

```
        try {
            fetchStudentById(searchId);
        } catch (SQLException e) {
            System.out.println("Database error occurred: " + e.getMessage());
        } catch (ClassNotFoundException e) {
            System.out.println("JDBC Driver not found.");
        } catch (Exception e) {
            System.out.println("Unexpected error: " + e.getMessage());
        }
    }
    public static void fetchStudentById(int id) throws SQLException, ClassNotFoundException {
        Class.forName("com.mysql.cj.jdbc.Driver");

        try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD)) {
            String sql = "SELECT * FROM students WHERE id = ?";
            try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
                pstmt.setInt(1, id);
                try (ResultSet rs = pstmt.executeQuery()) {
                    if (rs.next()) {
                        System.out.println("Student found:");
                        System.out.println("ID: " + rs.getInt("id"));
                        System.out.println("Name: " + rs.getString("name"));
                        System.out.println("Age: " + rs.getInt("age"));
                        System.out.println("Email: " + rs.getString("email"));
                    } else {
                        System.out.println("No student found with ID: " + id);
                    }
                }
            }
        }
    }
}
```

14.Create a login system using JDBC where user credentials are verified from the database.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;

public class LoginSystem {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter username: ");
    String inputUsername = scanner.nextLine();
    System.out.print("Enter password: ");
    String inputPassword = scanner.nextLine();

    boolean isValid = authenticateUser(inputUsername, inputPassword);
    if (isValid) {
      System.out.println("Login successful. Welcome, " + inputUsername + "!");
    } else {
      System.out.println("Invalid username or password.");
    }

    scanner.close();
  }

  public static boolean authenticateUser(String username, String password) {
```

```
    String sql = "SELECT * FROM users WHERE username = ? AND password = ?";
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
         PreparedStatement pstmt = conn.prepareStatement(sql)) {

        pstmt.setString(1, username);
        pstmt.setString(2, password);

        try (ResultSet rs = pstmt.executeQuery()) {
            return rs.next();
        }

    } catch (SQLException e) {
        System.out.println("Database error: " + e.getMessage());
        return false;
    }
  }
}
```

15.Implement a Java application to take dynamic input from the user and perform insertion, search, or update using menu-driven logic.

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

public class StudentManagementApp {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

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

```
StudentManagementApp app = new StudentManagementApp();

while (true) {
    System.out.println("\n--- Student Management ---");
    System.out.println("1. Insert Student");
    System.out.println("2. Search Student by ID");
    System.out.println("3. Update Student");
    System.out.println("4. Exit");
    System.out.print("Enter choice: ");
    int choice = scanner.nextInt();
    scanner.nextLine(); // consume newline

    switch (choice) {
        case 1:
            System.out.print("Enter name: ");
            String name = scanner.nextLine();
            System.out.print("Enter age: ");
            int age = scanner.nextInt();
            scanner.nextLine();
            System.out.print("Enter email: ");
            String email = scanner.nextLine();
            app.insertStudent(name, age, email);
            break;
        case 2:
            System.out.print("Enter student ID to search: ");
            int searchId = scanner.nextInt();
            scanner.nextLine();
            app.searchStudentById(searchId);
            break;
        case 3:
            System.out.print("Enter student ID to update: ");
            int updateId = scanner.nextInt();
```

```
            scanner.nextLine();
            System.out.print("Enter new name: ");
            String newName = scanner.nextLine();
            System.out.print("Enter new age: ");
            int newAge = scanner.nextInt();
            scanner.nextLine();
            System.out.print("Enter new email: ");
            String newEmail = scanner.nextLine();
            app.updateStudent(updateId, newName, newAge, newEmail);
            break;
        case 4:
            System.out.println("Exiting...");
            scanner.close();
            return;
        default:
            System.out.println("Invalid choice.");
    }
  }
}

public void insertStudent(String name, int age, String email) {
  String sql = "INSERT INTO students (name, age, email) VALUES (?, ?, ?)";
  try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
      PreparedStatement pstmt = conn.prepareStatement(sql)) {

    pstmt.setString(1, name);
    pstmt.setInt(2, age);
    pstmt.setString(3, email);

    int rows = pstmt.executeUpdate();
    if (rows > 0) {
      System.out.println("Student inserted successfully.");
```

```
        } else {
            System.out.println("Insertion failed.");
        }
    } catch (SQLException e) {
        System.out.println("Database error: " + e.getMessage());
    }
}

public void searchStudentById(int id) {
    String sql = "SELECT * FROM students WHERE id = ?";
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
         PreparedStatement pstmt = conn.prepareStatement(sql)) {

        pstmt.setInt(1, id);
        try (ResultSet rs = pstmt.executeQuery()) {
            if (rs.next()) {
                System.out.println("ID: " + rs.getInt("id"));
                System.out.println("Name: " + rs.getString("name"));
                System.out.println("Age: " + rs.getInt("age"));
                System.out.println("Email: " + rs.getString("email"));
            } else {
                System.out.println("Student not found.");
            }
        }
    } catch (SQLException e) {
        System.out.println("Database error: " + e.getMessage());
    }
}

public void updateStudent(int id, String name, int age, String email) {
    String sql = "UPDATE students SET name = ?, age = ?, email = ? WHERE id = ?";
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
```

```
        PreparedStatement pstmt = conn.prepareStatement(sql)) {

        pstmt.setString(1, name);
        pstmt.setInt(2, age);
        pstmt.setString(3, email);
        pstmt.setInt(4, id);

        int rows = pstmt.executeUpdate();
        if (rows > 0) {
            System.out.println("Student updated successfully.");
        } else {
            System.out.println("Update failed or student not found.");
        }
    } catch (SQLException e) {
        System.out.println("Database error: " + e.getMessage());
    }
  }
}
```

16.Design the schema for a Library Management System and write JDBC programs for:

- **Adding a book**
- **Viewing all books**
- **Issuing a book to a member**
- **Returning a book**

// 1.Library Management System Schema (MySQL)

```
CREATE TABLE books (
  book_id INT AUTO_INCREMENT PRIMARY KEY,
  title VARCHAR(255) NOT NULL,
  author VARCHAR(100),
  publisher VARCHAR(100),
```

```
  available BOOLEAN DEFAULT TRUE
);

CREATE TABLE members (
  member_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE issued_books (
  issue_id INT AUTO_INCREMENT PRIMARY KEY,
  book_id INT,
  member_id INT,
  issue_date DATE,
  return_date DATE,
  FOREIGN KEY (book_id) REFERENCES books(book_id),
  FOREIGN KEY (member_id) REFERENCES members(member_id)
);
```

// a) Adding a Book

```
import java.sql.*;
public class LibraryOperations {
  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public void addBook(String title, String author, String publisher) {
    String sql = "INSERT INTO books (title, author, publisher, available) VALUES (?, ?, ?, TRUE)";
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
        PreparedStatement pstmt = conn.prepareStatement(sql)) {
      pstmt.setString(1, title);
      pstmt.setString(2, author);
```

```
            pstmt.setString(3, publisher);
            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Book added successfully." : "Failed to add book.");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

// b) Viewing All Books
public void viewAllBooks() {
    String sql = "SELECT * FROM books";
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
         Statement stmt = conn.createStatement();
         ResultSet rs = stmt.executeQuery(sql)) {

        System.out.println("BookID | Title | Author | Publisher | Available");
        while (rs.next()) {
            System.out.printf("%d | %s | %s | %s | %s%n",
                rs.getInt("book_id"),
                rs.getString("title"),
                rs.getString("author"),
                rs.getString("publisher"),
                rs.getBoolean("available") ? "Yes" : "No");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

//c) Issuing a Book to a Member
import java.time.LocalDate;
```

```
public void issueBook(int bookId, int memberId) {
   String checkAvailability = "SELECT available FROM books WHERE book_id = ?";
   String issueBookSql = "INSERT INTO issued_books (book_id, member_id, issue_date) VALUES (?, ?,
?)";
   String updateBookAvailability = "UPDATE books SET available = FALSE WHERE book_id = ?";

   try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD)) {
      conn.setAutoCommit(false);

      try (PreparedStatement checkStmt = conn.prepareStatement(checkAvailability)) {
         checkStmt.setInt(1, bookId);
         ResultSet rs = checkStmt.executeQuery();
         if (rs.next() && rs.getBoolean("available")) {
            try (PreparedStatement issueStmt = conn.prepareStatement(issueBookSql);
                 PreparedStatement updateStmt = conn.prepareStatement(updateBookAvailability)) {

               issueStmt.setInt(1, bookId);
               issueStmt.setInt(2, memberId);
               issueStmt.setDate(3, Date.valueOf(LocalDate.now()));
               issueStmt.executeUpdate();

               updateStmt.setInt(1, bookId);
               updateStmt.executeUpdate();

               conn.commit();
               System.out.println("Book issued successfully.");
            }
         } else {
            System.out.println("Book is not available for issuing.");
            conn.rollback();
         }
      } catch (SQLException e) {
```

```
            conn.rollback();
            e.printStackTrace();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

// d) Returning a Book

```
import java.time.LocalDate;

public void returnBook(int bookId, int memberId) {
    String checkIssue = "SELECT issue_id FROM issued_books WHERE book_id = ? AND member_id = ? AND return_date IS NULL";
    String updateReturnDate = "UPDATE issued_books SET return_date = ? WHERE issue_id = ?";
    String updateBookAvailability = "UPDATE books SET available = TRUE WHERE book_id = ?";

    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD)) {
        conn.setAutoCommit(false);

        try (PreparedStatement checkStmt = conn.prepareStatement(checkIssue)) {
            checkStmt.setInt(1, bookId);
            checkStmt.setInt(2, memberId);
            ResultSet rs = checkStmt.executeQuery();

            if (rs.next()) {
                int issueId = rs.getInt("issue_id");

                try (PreparedStatement updateReturnStmt = conn.prepareStatement(updateReturnDate);
                     PreparedStatement updateBookStmt = conn.prepareStatement(updateBookAvailability)) {

                    updateReturnStmt.setDate(1, Date.valueOf(LocalDate.now()));
                    updateReturnStmt.setInt(2, issueId);
```

```
            updateReturnStmt.executeUpdate();

            updateBookStmt.setInt(1, bookId);
            updateBookStmt.executeUpdate();

            conn.commit();
            System.out.println("Book returned successfully.");
          }
        } else {
          System.out.println("No outstanding issue found for this book and member.");
          conn.rollback();
        }
      } catch (SQLException e) {
        conn.rollback();
        e.printStackTrace();
      }
    } catch (SQLException e) {
      e.printStackTrace();
    }
}
```

17.Create a Hospital Management System database. Using JDBC, implement:

- **Register new patient**
- **Assign doctor**
- **Generate billing**

// 1. Database Schema (MySQL)

```
CREATE TABLE patients (
  patient_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  age INT,
```

```
  gender VARCHAR(10),
  contact VARCHAR(15)
);

CREATE TABLE doctors (
  doctor_id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  specialization VARCHAR(100)
);

CREATE TABLE assignments (
  assignment_id INT AUTO_INCREMENT PRIMARY KEY,
  patient_id INT,
  doctor_id INT,
  assign_date DATE,
  FOREIGN KEY (patient_id) REFERENCES patients(patient_id),
  FOREIGN KEY (doctor_id) REFERENCES doctors(doctor_id)
);

CREATE TABLE billing (
  bill_id INT AUTO_INCREMENT PRIMARY KEY,
  patient_id INT,
  amount DECIMAL(10,2),
  bill_date DATE,
  FOREIGN KEY (patient_id) REFERENCES patients(patient_id)
);
```

// 2. Complete Java Program with JDBC

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

```
public class HospitalManagementSystem {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

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

    while (true) {
      System.out.println("\n--- Hospital Management ---");
      System.out.println("1. Register New Patient");
      System.out.println("2. Assign Doctor to Patient");
      System.out.println("3. Generate Billing");
      System.out.println("4. Exit");
      System.out.print("Enter choice: ");

      int choice = scanner.nextInt();
      scanner.nextLine(); // consume newline

      switch (choice) {
        case 1:
          System.out.print("Enter patient name: ");
          String pname = scanner.nextLine();
          System.out.print("Enter age: ");
          int page = scanner.nextInt();
          scanner.nextLine();
          System.out.print("Enter gender: ");
          String gender = scanner.nextLine();
          System.out.print("Enter contact number: ");
          String contact = scanner.nextLine();
```

```
            hms.registerPatient(pname, page, gender, contact);
            break;

        case 2:
            System.out.print("Enter patient ID: ");
            int pid = scanner.nextInt();
            System.out.print("Enter doctor ID: ");
            int did = scanner.nextInt();
            scanner.nextLine();
            hms.assignDoctor(pid, did);
            break;

        case 3:
            System.out.print("Enter patient ID for billing: ");
            int billPid = scanner.nextInt();
            System.out.print("Enter amount: ");
            double amount = scanner.nextDouble();
            scanner.nextLine();
            hms.generateBilling(billPid, amount);
            break;

        case 4:
            System.out.println("Exiting...");
            scanner.close();
            return;

        default:
            System.out.println("Invalid choice.");
      }
    }
}
```

```
  public void registerPatient(String name, int age, String gender, String contact) {
    String sql = "INSERT INTO patients (name, age, gender, contact) VALUES (?, ?, ?, ?)";
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
        PreparedStatement pstmt = conn.prepareStatement(sql,
Statement.RETURN_GENERATED_KEYS)) {

      pstmt.setString(1, name);
      pstmt.setInt(2, age);
      pstmt.setString(3, gender);
      pstmt.setString(4, contact);

      int rows = pstmt.executeUpdate();
      if (rows > 0) {
        try (ResultSet keys = pstmt.getGeneratedKeys()) {
          if (keys.next()) {
            System.out.println("Patient registered with ID: " + keys.getInt(1));
          }
        }
      } else {
        System.out.println("Failed to register patient.");
      }
    } catch (SQLException e) {
      System.out.println("Error registering patient: " + e.getMessage());
    }
  }

  public void assignDoctor(int patientId, int doctorId) {
    String sql = "INSERT INTO assignments (patient_id, doctor_id, assign_date) VALUES (?, ?, ?)";
    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
        PreparedStatement pstmt = conn.prepareStatement(sql)) {

      pstmt.setInt(1, patientId);
      pstmt.setInt(2, doctorId);
```

```
            pstmt.setDate(3, Date.valueOf(LocalDate.now()));

            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Doctor assigned successfully." : "Failed to assign doctor.");
        } catch (SQLException e) {
            System.out.println("Error assigning doctor: " + e.getMessage());
        }
    }

    public void generateBilling(int patientId, double amount) {
        String sql = "INSERT INTO billing (patient_id, amount, bill_date) VALUES (?, ?, ?)";
        try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            pstmt.setInt(1, patientId);
            pstmt.setDouble(2, amount);
            pstmt.setDate(3, Date.valueOf(LocalDate.now()));

            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Billing generated successfully." : "Failed to generate billing.");
        } catch (SQLException e) {
            System.out.println("Error generating billing: " + e.getMessage());
        }
    }
}
```

18.Write a JDBC-based report generator that exports data from a MySQL table to a text or CSV file.

```
import java.sql.*;
import java.io.FileWriter;
import java.io.IOException;
```

```
public class ExportToCSV {

  static final String JDBC_URL = "jdbc:mysql://localhost:3306/your_database_name";
  static final String USER = "your_username";
  static final String PASSWORD = "your_password";

  public static void main(String[] args) {
    String tableName = "your_table_name";
    String outputFile = "exported_data.csv";

    try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
      Statement stmt = conn.createStatement();
      ResultSet rs = stmt.executeQuery("SELECT * FROM " + tableName);
      FileWriter fw = new FileWriter(outputFile)) {

      ResultSetMetaData meta = rs.getMetaData();
      int columnCount = meta.getColumnCount();

      for (int i = 1; i <= columnCount; i++) {
        fw.append(meta.getColumnName(i));
        if (i < columnCount) fw.append(",");
      }
      fw.append("\n");

      while (rs.next()) {
        for (int i = 1; i <= columnCount; i++) {
          String data = rs.getString(i);
          if (data != null) {
            data = data.replaceAll("\"", "\"\"");
            if (data.contains(",") || data.contains("\"")) {
              data = "\"" + data + "\"";
```

```
                }
            }
            fw.append(data != null ? data : "");
            if (i < columnCount) fw.append(",");
          }
          fw.append("\n");
        }

        System.out.println("Data exported to " + outputFile);

      } catch (SQLException e) {
        System.out.println("SQL Error: " + e.getMessage());
      } catch (IOException e) {
        System.out.println("File Error: " + e.getMessage());
      }
    }
}
```