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

```
         import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

import java.sql.SQLException;


public class JdbcMySQLConnection {

   public static void main(String[] args) {

      String jdbcURL = "jdbc:mysql://localhost:3306/testdb"; name

      String username = "root";  // Your MySQL username

      String password = "password"; // Your MySQL password


      Connection connection = null;


      try {


         Class.forName("com.mysql.cj.jdbc.Driver");


         connection = DriverManager.getConnection(jdbcURL, username, password);

         System.out.println("Connected to the MySQL database!");



         Statement stmt = connection.createStatement();


         String sql = "SELECT id, name FROM users";

         ResultSet rs = stmt.executeQuery(sql);
```

```
            while (rs.next()) {
                int id = rs.getInt("id");
                String name = rs.getString("name");
                System.out.println("ID: " + id + ", Name: " + name);
            }



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

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (connection != null && !connection.isClosed()) {
                    connection.close();
                    System.out.println("Database connection closed.");
                }
            } catch (SQLException e) {
                e.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.ResultSet;

import java.sql.Statement;

import java.sql.SQLException;


public class JdbcMySQLConnection {

    public static void main(String[] args) {

        String jdbcURL = "jdbc:mysql://localhost:3306/testdb";

        String username = "root";

        String password = "password";

        Connection connection = null;

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");

            connection = DriverManager.getConnection(jdbcURL, username, password);

            System.out.println("Connected to the MySQL database!");

            Statement stmt = connection.createStatement();

            String sql = "SELECT id, name FROM users";

            ResultSet rs = stmt.executeQuery(sql);

            while (rs.next()) {

                int id = rs.getInt("id");

                String name = rs.getString("name");

                System.out.println("ID: " + id + ", Name: " + name);

            }

            rs.close();

            stmt.close();

        } catch (Exception e) {

            e.printStackTrace();

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

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 {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static void main(String[] args) {
        Connection conn = null;
        Statement stmt = null;
        ResultSet rs = null;
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
            stmt = conn.createStatement();
            String sql = "SELECT id, name, age, grade FROM students";
            rs = stmt.executeQuery(sql);
            while (rs.next()) {
                int id = rs.getInt("id");
                String name = rs.getString("name");
                int age = rs.getInt("age");
                String grade = rs.getString("grade");
                System.out.println("ID: " + id + ", Name: " + name + ", Age: " + age + ", Grade: " +
grade);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (rs != null) rs.close();
                if (stmt != null) stmt.close();
                if (conn != null) conn.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;

import java.util.Scanner;


public class SearchStudentByID {

    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";

    private static final String JDBC_USER = "root";

    private static final String JDBC_PASSWORD = "password";


    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter Student ID to search: ");

        int studentId = scanner.nextInt();

        searchStudent(studentId);

        scanner.close();

    }


    public static void searchStudent(int id) {

        Connection conn = null;

        PreparedStatement pstmt = null;

        ResultSet rs = null;

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
            String sql = "SELECT id, name, age, grade FROM students WHERE id = ?";
            pstmt = conn.prepareStatement(sql);
            pstmt.setInt(1, id);
            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("Grade: " + rs.getString("grade"));
            } else {
                System.out.println("No student found with ID: " + id);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (rs != null) rs.close();
                if (pstmt != null) pstmt.close();
                if (conn != null) conn.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;

import java.util.Scanner;


public class UpdateStudent {

    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";

    private static final String JDBC_USER = "root";

    private static final String JDBC_PASSWORD = "password";


    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);


        System.out.print("Enter Student ID to update: ");

        int id = scanner.nextInt();

        scanner.nextLine();


        System.out.print("Enter new Name: ");

        String name = scanner.nextLine();


        System.out.print("Enter new Age: ");

        int age = scanner.nextInt();

        scanner.nextLine();


        System.out.print("Enter new Grade: ");

        String grade = scanner.nextLine();
        updateStudent(id, name, age, grade);
        scanner.close();
    }

    public static void updateStudent(int id, String name, int age, String grade) {
        Connection conn = null;
        PreparedStatement pstmt = null;
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
            String sql = "UPDATE students SET name = ?, age = ?, grade = ? WHERE id = ?";
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, name);
            pstmt.setInt(2, age);
            pstmt.setString(3, grade);
            pstmt.setInt(4, id);
            int rows = pstmt.executeUpdate();
            if (rows > 0) {
                System.out.println("Student record updated successfully.");
            } else {
                System.out.println("No student found with ID: " + id);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (pstmt != null) pstmt.close();
            if (conn != null) conn.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;
import java.util.Scanner;

public class DeleteStudent {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter Student ID to delete: ");
        int id = scanner.nextInt();
        deleteStudent(id);
        scanner.close();
    }
    }

    public static void deleteStudent(int id) {
        Connection conn = null;
        PreparedStatement pstmt = null;
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
            String sql = "DELETE FROM students WHERE id = ?";
            pstmt = conn.prepareStatement(sql);
            pstmt.setInt(1, id);
            int rows = pstmt.executeUpdate();
            if (rows > 0) {
                System.out.println("Student with ID " + id + " deleted successfully.");
            } else {
                System.out.println("No student found with ID: " + id);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (pstmt != null) pstmt.close();
                if (conn != null) conn.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 EmployeeCRUDApp {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/companydb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        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. Search Employee by ID");
            System.out.println("4. Update Employee");
            System.out.println("5. Delete Employee");
            System.out.println("6. Exit");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();
            scanner.nextLine();

            switch (choice) {
```

```
            case 1:
                addEmployee(scanner);
                break;
            case 2:
                viewEmployees();
                break;
            case 3:
                searchEmployee(scanner);
                break;
            case 4:
                updateEmployee(scanner);
                break;
            case 5:
                deleteEmployee(scanner);
                break;
            case 6:
                System.out.println("Exiting...");
                scanner.close();
                return;
            default:
                System.out.println("Invalid choice! Try again.");
        }
    }
}

private static Connection getConnection() throws Exception {
    Class.forName("com.mysql.cj.jdbc.Driver");
    return DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
    }

    private static void addEmployee(Scanner scanner) {
        System.out.print("Enter name: ");
        String name = scanner.nextLine();
        System.out.print("Enter salary: ");
        double salary = scanner.nextDouble();
        scanner.nextLine();
        System.out.print("Enter department: ");
        String department = scanner.nextLine();

        try (Connection conn = getConnection();
             PreparedStatement pstmt = conn.prepareStatement(
                     "INSERT INTO employees (name, salary, department) VALUES (?, ?, ?)")) {
            pstmt.setString(1, name);
            pstmt.setDouble(2, salary);
            pstmt.setString(3, department);
            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Employee added successfully." : "Failed to add
employee.");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void viewEmployees() {
        try (Connection conn = getConnection();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery("SELECT id, name, salary, department FROM
employees")) {
            while (rs.next()) {
                System.out.println("ID: " + rs.getInt("id") +
                        ", Name: " + rs.getString("name") +
                        ", Salary: " + rs.getDouble("salary") +
                        ", Department: " + rs.getString("department"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void searchEmployee(Scanner scanner) {
        System.out.print("Enter Employee ID: ");
        int id = scanner.nextInt();
        try (Connection conn = getConnection();
             PreparedStatement pstmt = conn.prepareStatement("SELECT * FROM employees
WHERE id = ?")) {
            pstmt.setInt(1, id);
            ResultSet rs = pstmt.executeQuery();
            if (rs.next()) {
                System.out.println("ID: " + rs.getInt("id") +
                        ", Name: " + rs.getString("name") +
                        ", Salary: " + rs.getDouble("salary") +
                        ", Department: " + rs.getString("department"));
            } else {
                System.out.println("Employee not found.");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void updateEmployee(Scanner scanner) {
        System.out.print("Enter Employee ID to update: ");
        int id = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Enter new Name: ");
        String name = scanner.nextLine();
        System.out.print("Enter new Salary: ");
        double salary = scanner.nextDouble();
        scanner.nextLine();
        System.out.print("Enter new Department: ");
        String department = scanner.nextLine();

        try (Connection conn = getConnection();
             PreparedStatement pstmt = conn.prepareStatement(
                     "UPDATE employees SET name=?, salary=?, department=? WHERE id=?")) {
            pstmt.setString(1, name);
            pstmt.setDouble(2, salary);
            pstmt.setString(3, department);
            pstmt.setInt(4, id);
            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Employee updated successfully." : "Employee not
found.");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void deleteEmployee(Scanner scanner) {
        System.out.print("Enter Employee ID to delete: ");
        int id = scanner.nextInt();
        try (Connection conn = getConnection();
             PreparedStatement pstmt = conn.prepareStatement("DELETE FROM employees
WHERE id=?")) {
            pstmt.setInt(1, id);
            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Employee deleted successfully." : "Employee not
found.");
        } catch (Exception 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.ResultSet;
import java.sql.Statement;

public class RowCount {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
private static final String JDBC_USER = "root";
private static final String JDBC_PASSWORD = "password";

public static void main(String[] args) {
    Connection conn = null;
    Statement stmt = null;
    ResultSet rs = null;
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
        conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
        stmt = conn.createStatement();
        String sql = "SELECT COUNT(*) AS total FROM students";
        rs = stmt.executeQuery(sql);
        if (rs.next()) {
            int count = rs.getInt("total");
            System.out.println("Total number of rows in 'students' table: " + count);
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {
            if (rs != null) rs.close();
            if (stmt != null) stmt.close();
            if (conn != null) conn.close();
        } catch (Exception 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;

public class SortStudentsByName {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static void main(String[] args) {
        Connection conn = null;
        Statement stmt = null;
        ResultSet rs = null;

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
            stmt = conn.createStatement();
            String sql = "SELECT id, name, age, grade FROM students ORDER BY name ASC";
            rs = stmt.executeQuery(sql);
            while (rs.next()) {
                int id = rs.getInt("id");
                String name = rs.getString("name");
                int age = rs.getInt("age");
                String grade = rs.getString("grade");

                System.out.println("ID: " + id + ", Name: " + name + ", Age: " + age + ", Grade: " +
grade);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (rs != null) rs.close();
                if (stmt != null) stmt.close();
                if (conn != null) conn.close();
            } catch (Exception 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;


public class StudentsAbovePercentage {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";


    public static void main(String[] args) {
        Connection conn = null;
        Statement stmt = null;
        ResultSet rs = null;


        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD);
            stmt = conn.createStatement();
            String sql = "SELECT id, name, age, grade, percentage FROM students WHERE
percentage > 75";
            rs = stmt.executeQuery(sql);


            while (rs.next()) {
                int id = rs.getInt("id");
                String name = rs.getString("name");
                int age = rs.getInt("age");
                String grade = rs.getString("grade");
                double percentage = rs.getDouble("percentage");


                System.out.println("ID: " + id + ", Name: " + name + ", Age: " + age +
                                ", Grade: " + grade + ", Percentage: " + percentage);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            try {
                if (rs != null) rs.close();
                if (stmt != null) stmt.close();
                if (conn != null) conn.close();
            } catch (Exception 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;

public class BatchInsertStudents {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

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

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

            Class.forName("com.mysql.cj.jdbc.Driver");

            conn.setAutoCommit(false);

            pstmt.setString(1, "Amit Sharma");
            pstmt.setInt(2, 20);
            pstmt.setString(3, "A");
            pstmt.addBatch();

            pstmt.setString(1, "Priya Verma");
            pstmt.setInt(2, 19);
            pstmt.setString(3, "B");
            pstmt.addBatch();

            pstmt.setString(1, "Rohit Singh");
            pstmt.setInt(2, 21);
            pstmt.setString(3, "A");
            pstmt.addBatch();

            int[] counts = pstmt.executeBatch();
            conn.commit();
            System.out.println("Inserted records count: " + counts.length);

        } catch (Exception 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;

public class JdbcTransactionExample {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

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

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

            Class.forName("com.mysql.cj.jdbc.Driver");
            conn.setAutoCommit(false);

            pstmt.setString(1, "John Doe");
            pstmt.setInt(2, 22);
            pstmt.setString(3, "B");
            pstmt.executeUpdate();

            // Intentionally cause an error in the second insert (e.g., null for NOT NULL column)
            pstmt.setString(1, null); // This will cause SQL exception if name is NOT NULL
            pstmt.setInt(2, 25);
            pstmt.setString(3, "A");
            pstmt.executeUpdate();

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

        } catch (Exception e) {
            e.printStackTrace();
            try (Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
    JDBC_PASSWORD)) {
                if (conn != null) {
                    conn.rollback();
                    System.out.println("Transaction rolled back due to error.");
                }
            } catch (Exception rollbackEx) {
                rollbackEx.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;
import java.util.Scanner;

public class JdbcExceptionHandling {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter Student ID to search: ");
        int studentId = 0;
        try {
            studentId = Integer.parseInt(scanner.nextLine());
        } catch (NumberFormatException e) {
            System.out.println("Invalid input. Please enter a valid numeric ID.");
            scanner.close();
            return;
        }

        try (Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
JDBC_PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement("SELECT id, name, age, grade
FROM students WHERE id = ?")) {

            Class.forName("com.mysql.cj.jdbc.Driver");

            pstmt.setInt(1, studentId);
            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("Grade: " + rs.getString("grade"));
            } else {
                System.out.println("No student found with ID: " + studentId);
            }
            rs.close();

        } catch (ClassNotFoundException e) {
            System.out.println("JDBC Driver not found. Please make sure the driver is included.");
        } catch (SQLException e) {
            System.out.println("Database error occurred: " + e.getMessage());
        } catch (Exception e) {
            System.out.println("Unexpected error: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

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.util.Scanner;

public class JdbcLoginSystem {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

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

        System.out.print("Enter username: ");
        String username = scanner.nextLine();

        System.out.print("Enter password: ");
        String password = scanner.nextLine();
        boolean isAuthenticated = authenticateUser(username, password);


        if (isAuthenticated) {
            System.out.println("Login successful. Welcome, " + username + "!");
        } else {
            System.out.println("Invalid username or password. Login failed.");
        }


        scanner.close();
    }


    private static boolean authenticateUser(String username, String password) {
        String sql = "SELECT * FROM users WHERE username = ? AND password = ?";
        try (Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
JDBC_PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {


            Class.forName("com.mysql.cj.jdbc.Driver");


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


            ResultSet rs = pstmt.executeQuery();
            boolean userExists = rs.next();
            rs.close();
            return userExists;


        } catch (Exception e) {
            System.out.println("Error during authentication: " + e.getMessage());
            return false;
        }
    }
}

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

public class JdbcLoginSystem {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

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

        System.out.print("Enter username: ");
        String username = scanner.nextLine();

        System.out.print("Enter password: ");
        String password = scanner.nextLine();

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

        scanner.close();
    }

    private static boolean authenticateUser(String username, String password) {
        String sql = "SELECT * FROM users WHERE username = ? AND password = ?";
        try (Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
JDBC_PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            Class.forName("com.mysql.cj.jdbc.Driver");

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

            ResultSet rs = pstmt.executeQuery();
            boolean userExists = rs.next();
            rs.close();
            return userExists;

        } catch (Exception e) {
            System.out.println("Error during authentication: " + e.getMessage());
            return false;
        }
    }
}
```

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

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

public class AddBook {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/librarydb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static void addBook(String title, String author) {
        String sql = "INSERT INTO books (title, author, available) VALUES (?, ?, TRUE)";
        try (Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
JDBC_PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            Class.forName("com.mysql.cj.jdbc.Driver");

            pstmt.setString(1, title);
            pstmt.setString(2, author);
            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Book added successfully." : "Failed to add book.");

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}



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

public class ViewBooks {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/librarydb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static void viewAllBooks() {
        String sql = "SELECT book_id, title, author, available FROM books";
        try (Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
JDBC_PASSWORD);
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(sql)) {
```

```
            Class.forName("com.mysql.cj.jdbc.Driver");


            while (rs.next()) {
                System.out.println("ID: " + rs.getInt("book_id") +
                                   ", Title: " + rs.getString("title") +
                                   ", Author: " + rs.getString("author") +
                                   ", Available: " + rs.getBoolean("available"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}


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


public class IssueBook {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/librarydb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";


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

        try (Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER, 
JDBC_PASSWORD);
             PreparedStatement checkStmt = conn.prepareStatement(checkAvailability);
             PreparedStatement insertStmt = conn.prepareStatement(insertIssue);
             PreparedStatement updateStmt = conn.prepareStatement(updateBook)) {

            Class.forName("com.mysql.cj.jdbc.Driver");

            conn.setAutoCommit(false);

            checkStmt.setInt(1, bookId);
            ResultSet rs = checkStmt.executeQuery();
            if (rs.next()) {
                boolean available = rs.getBoolean("available");
                if (!available) {
                    System.out.println("Book is currently not available.");
                    return;
                }
            } else {
                System.out.println("Book not found.");
                return;
            }

            insertStmt.setInt(1, bookId);
            insertStmt.setInt(2, memberId);
            insertStmt.setDate(3, new Date(System.currentTimeMillis()));
            insertStmt.executeUpdate();

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

            conn.commit();
            System.out.println("Book issued successfully.");

        } catch (Exception e) {
            e.printStackTrace();
            try {
                Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
JDBC_PASSWORD);
                if (conn != null) conn.rollback();
            } catch (SQLException se) {
                se.printStackTrace();
            }
        }
    }
}

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

```
public class ReturnBook {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/librarydb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static void returnBook(int bookId, int memberId) {
        String updateIssue = "UPDATE issued_books SET return_date = ? WHERE book_id = ?
AND member_id = ? AND return_date IS NULL";
        String updateBook = "UPDATE books SET available = TRUE WHERE book_id = ?";

        try (Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
JDBC_PASSWORD);
             PreparedStatement updateIssueStmt = conn.prepareStatement(updateIssue);
             PreparedStatement updateBookStmt = conn.prepareStatement(updateBook)) {

            Class.forName("com.mysql.cj.jdbc.Driver");

            conn.setAutoCommit(false);

            updateIssueStmt.setDate(1, new Date(System.currentTimeMillis()));
            updateIssueStmt.setInt(2, bookId);
            updateIssueStmt.setInt(3, memberId);
            int updatedRows = updateIssueStmt.executeUpdate();

            if (updatedRows == 0) {
                System.out.println("No issued record found for this book and member.");
                conn.rollback();
                return;
            }
            updateBookStmt.setInt(1, bookId);
            updateBookStmt.executeUpdate();

            conn.commit();
            System.out.println("Book returned successfully.");

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

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

· Register new patient

· Assign doctor

· Generate billing

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

public class RegisterPatient {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/hospitaldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static 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, JDBC_USER,
JDBC_PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            Class.forName("com.mysql.cj.jdbc.Driver");

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

            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Patient registered successfully." : "Failed to register
patient.");

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.Date;

public class AssignDoctor {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/hospitaldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";

    public static 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, JDBC_USER,
JDBC_PASSWORD);
             PreparedStatement pstmt = conn.prepareStatement(sql)) {

            Class.forName("com.mysql.cj.jdbc.Driver");

            pstmt.setInt(1, patientId);
            pstmt.setInt(2, doctorId);
            pstmt.setDate(3, new Date(System.currentTimeMillis()));

            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Doctor assigned successfully." : "Failed to assign
doctor.");

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.Date;


public class GenerateBilling {
    private static final String JDBC_URL = "jdbc:mysql://localhost:3306/hospitaldb";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "password";


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


            Class.forName("com.mysql.cj.jdbc.Driver");


            pstmt.setInt(1, patientId);
            pstmt.setDouble(2, amount);
            pstmt.setDate(3, new Date(System.currentTimeMillis()));


            int rows = pstmt.executeUpdate();
            System.out.println(rows > 0 ? "Billing generated successfully." : "Failed to generate
billing.");


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

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

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.io.FileWriter;
import java.io.BufferedWriter;

public class JdbcCsvExport {
  private static final String JDBC_URL = "jdbc:mysql://localhost:3306/schooldb";
  private static final String JDBC_USER = "root";
  private static final String JDBC_PASSWORD = "password";
  private static final String OUTPUT_FILE = "students_export.csv";

  public static void main(String[] args) {
    try (
      Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USER,
JDBC_PASSWORD);
      Statement stmt = conn.createStatement();
      ResultSet rs = stmt.executeQuery("SELECT id, name, age, grade FROM students");
      BufferedWriter writer = new BufferedWriter(new FileWriter(OUTPUT_FILE))
    ) {
      Class.forName("com.mysql.cj.jdbc.Driver");

      // Write CSV header
      writer.write("ID,Name,Age,Grade");
      writer.newLine();
```

```
            // Write data rows
            while (rs.next()) {
                String row = rs.getInt("id") + "," +
                            rs.getString("name") + "," +
                            rs.getInt("age") + "," +
                            rs.getString("grade");
                writer.write(row);
                writer.newLine();
            }

            System.out.println("Data exported successfully to " + OUTPUT_FILE);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```