

## Assignment: Database Connectivity using JDBC

**Name:** Saurav Ugalkar

**Roll No:** 66

---

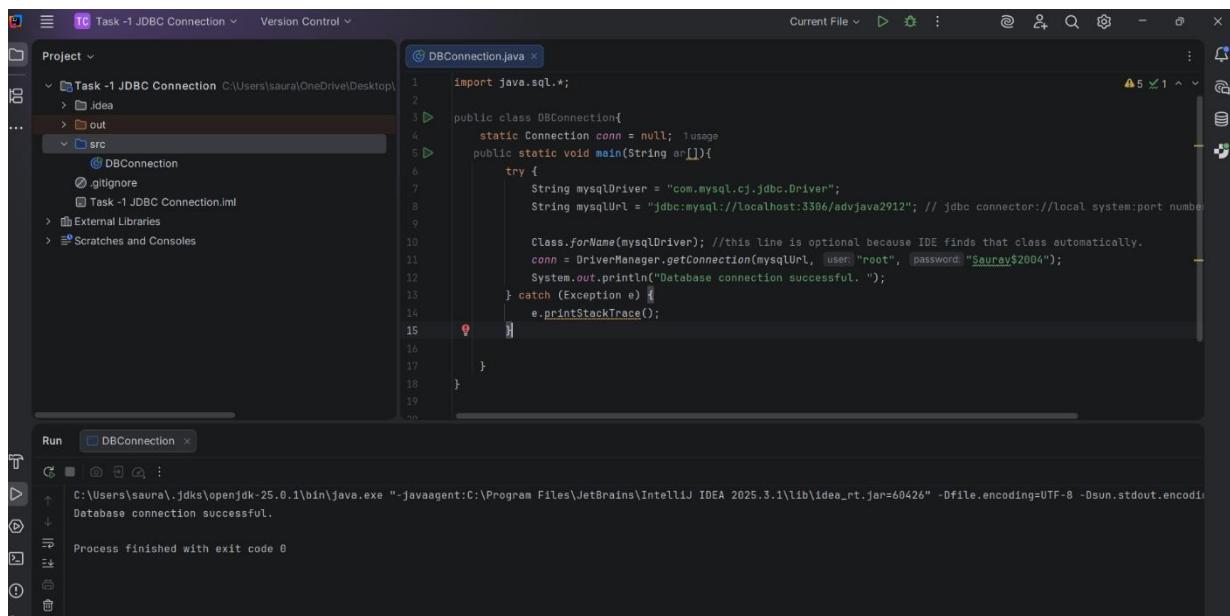
### Task 1: Establishing Database Connection

**File Name:** DBConnection.java

#### 1. Description

The primary objective of this task is to establish a connection between a Java application and a MySQL database named advjava. It utilizes the DriverManager class and the MySQL JDBC driver.

#### 2. Source Code with output



The screenshot shows the IntelliJ IDEA interface with the project 'Task -1 JDBC Connection' open. The 'src' directory contains a single Java file, 'DBConnection.java'. The code in the file is as follows:

```
1 import java.sql.*;
2
3 public class DBConnection{
4     static Connection conn = null; //usage
5     public static void main(String args[]){
6         try {
7             String mysqlDriver = "com.mysql.cj.jdbc.Driver";
8             String mysqlUrl = "jdbc:mysql://localhost:3306/advjava"; // jdbc connector://local system:port number
9
10            Class.forName(mysqlDriver); //this line is optional because IDE finds that class automatically.
11            conn = DriverManager.getConnection(mysqlUrl, "root", "Saurav$2004");
12            System.out.println("Database connection successful. ");
13        } catch (Exception e) {
14            e.printStackTrace();
15        }
16    }
17 }
18 }
```

In the bottom right corner of the code editor, there are two status indicators: a yellow warning icon with '5' and a green checkmark icon with '1'. The 'Run' tool window at the bottom shows the command run: 'C:\Users\saura\.jdks\openjdk-25.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.1\lib\idea\_rt.jar=60426" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8'. The output shows 'Database connection successful.' and 'Process finished with exit code 0'.

#### 3. Execution Logic

- Driver Loading:** Uses “com.mysql.cj.jdbc.Driver” to communicate with the MySQL server.
  - Connection String:** The URL “jdbc:mysql://localhost:3306/advjava” specifies the protocol, host, port, and database name.
  - Authentication:** Connects using the username root and the specified password.
-

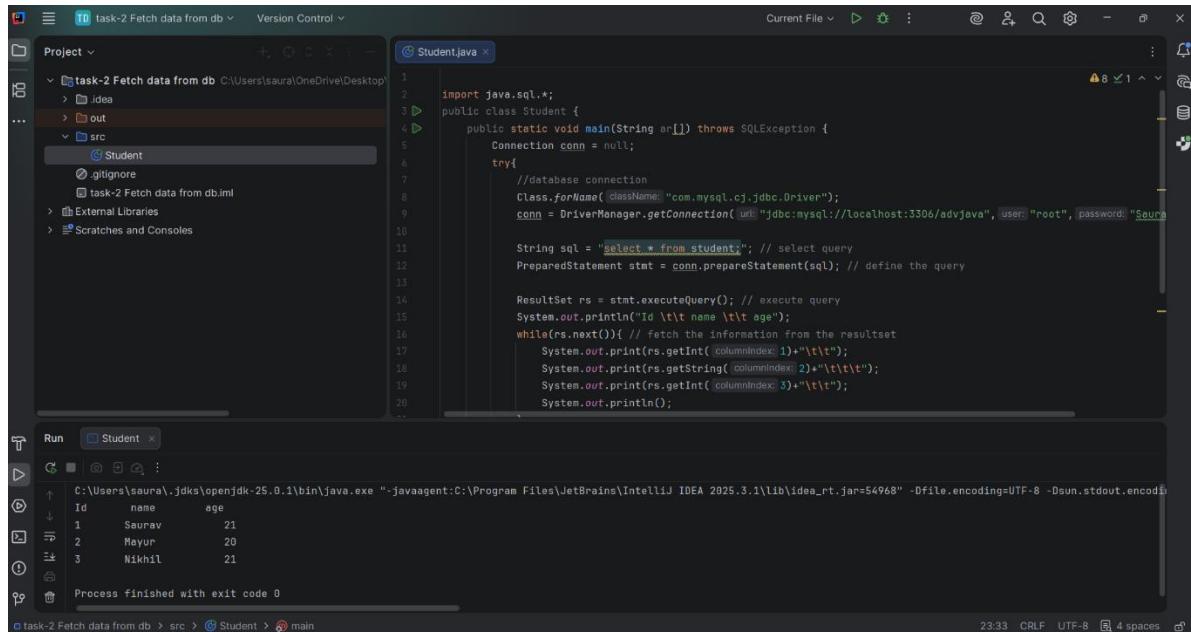
## **Task 2: Retrieving and Displaying Data**

**File Name:** Student.java

### **1. Description**

This task focuses on fetching all records from the student table and displaying them in a formatted tabular manner in the Java console.

### **2. Source Code with output**



The screenshot shows the IntelliJ IDEA interface with the following details:

- Project View:** Shows a project named "task-2 Fetch data from db" with a "src" folder containing a "Student" package.
- Code Editor:** Displays the `Student.java` file with the following code:

```
1 import java.sql.*;
2 public class Student {
3     public static void main(String args[]) throws SQLException {
4         Connection conn = null;
5         try{
6             //database connection
7             Class.forName("com.mysql.cj.jdbc.Driver");
8             conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/advjava", "root", "Saurav");
9
10            String sql = "select * from student"; // select query
11            PreparedStatement stat = conn.prepareStatement(sql); // define the query
12
13            ResultSet rs = stat.executeQuery(); // execute query
14            System.out.println("Id \t name \t\t age");
15            while(rs.next()){ // fetch the information from the resultset
16                System.out.print(rs.getInt( columnIndex: 1)+"\t");
17                System.out.print(rs.getString( columnIndex: 2)+"\t\t");
18                System.out.print(rs.getInt( columnIndex: 3)+"\t");
19                System.out.println();
20            }
21        } catch (Exception e) {
22            e.printStackTrace();
23        }
24    }
25 }
```
- Run Tab:** Shows the output of the program:

```
C:\Users\saurav\.jdks\openjdk-25.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.1\lib\idea_rt.jar=54968" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8
Id      name   age
1      Saurav  21
2      Mayur   20
3      Nikhil  21
Process finished with exit code 0
```
- Status Bar:** Shows the time as 23:33, CPU as 0%, and encoding as UTF-8.

### **3. Execution Logic**

- ResultSet:** Stores the data returned by the SELECT query.
- While Loop:** Iterates through the `rs.next()` cursor to print each row's ID, Name, and Age.
- Formatting:** Uses `\t` (tabs) to align columns in the output.

### **4. MySQL Verification**

```
select * from student;
```

id   name   age
+---+-----+-----+
1   Saurav   21
2   Mayur   20
3   Nikhil   21

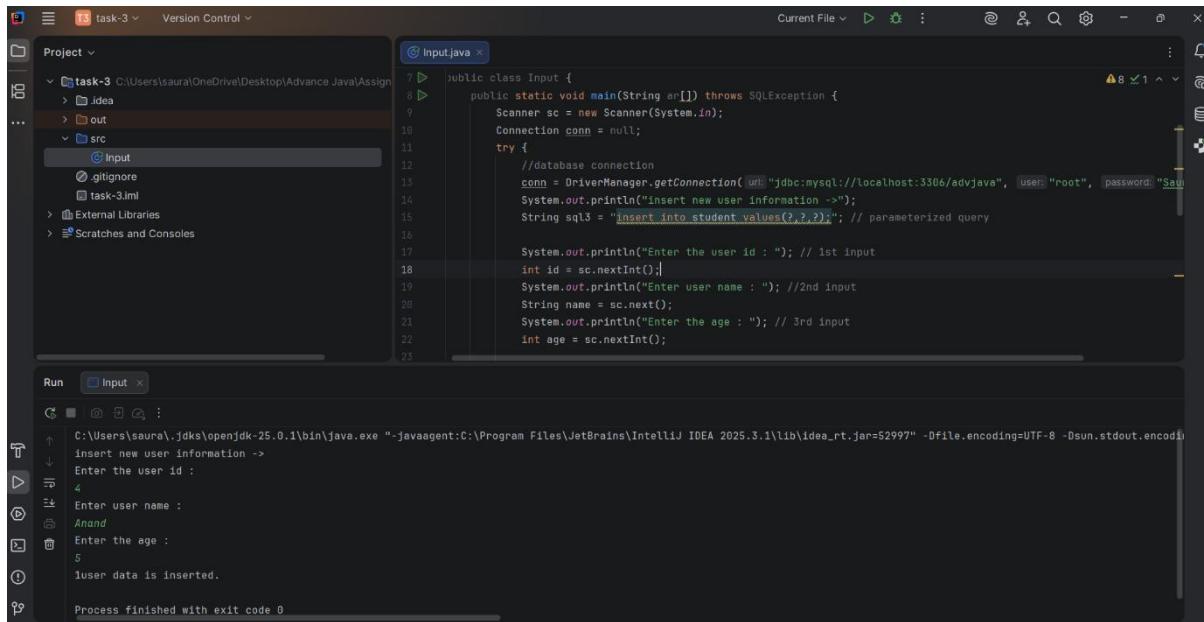
### **Task 3: Inserting Data into Database**

**File Name:** Input.java

#### **1. Description**

This task demonstrates how to accept user input from the console and insert a new record into the student table using a “PreparedStatement”.

#### **2. Source Code with output**



The screenshot shows the IntelliJ IDEA interface. The left sidebar displays a project structure with a 'task-3' folder containing 'idea', 'out', 'src', and 'Input'. The 'Input.java' file is open in the main editor window. The code implements a 'main' method that connects to a MySQL database, prompts for user ID, name, and age via Scanner, and inserts the data into a 'student' table using a parameterized query. The 'Run' tool window at the bottom shows the execution process and its output, which includes the SQL command, user inputs (ID 4, name Anand, age 5), and the confirmation message 'User data is inserted.'

```
public class Input {
    public static void main(String args[]) throws SQLException {
        Scanner sc = new Scanner(System.in);
        Connection conn = null;
        try {
            //database connection
            conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/advjava", "root", "Saurav");
            System.out.println("insert new user information ->");
            String sql3 = "insert into student values(?, ?, ?);"; // parameterized query
            System.out.println("Enter the user id : "); // 1st input
            int id = sc.nextInt();
            System.out.println("Enter user name : ");
            String name = sc.next();
            System.out.println("Enter the age : ");
            int age = sc.nextInt();
        } catch (Exception e) {
            e.printStackTrace();
        }
        finally {
            if (conn != null) {
                try {
                    conn.close();
                } catch (SQLException ex) {
                    ex.printStackTrace();
                }
            }
        }
    }
}
```

Run

```
C:\Users\saurav\.jdks\openjdk-25.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.1\lib\idea_rt.jar=52997" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8
```

insert new user information ->  
Enter the user id :  
4  
Enter user name :  
Anand  
Enter the age :  
5  
User data is inserted.  
Process finished with exit code 0

#### **3. Execution Logic**

- Parameterized Query:** Uses ? placeholders to prevent SQL injection.
- Scanner Class:** Captures id, name, and age from the user.
- executeUpdate():** This method is called to perform the DML (Insert) operation.

#### **4. MySQL Verification**

```
select * from student;
```

```
| id | name | age |
```

```
+---+-----+----+
```

```
| 1 | Saurav | 21 |
```

```
| 2 | Mayur | 20 |
```

```
| 3 | Nikhil | 21 |
```

```
| 4 | Anand | 5 |
```

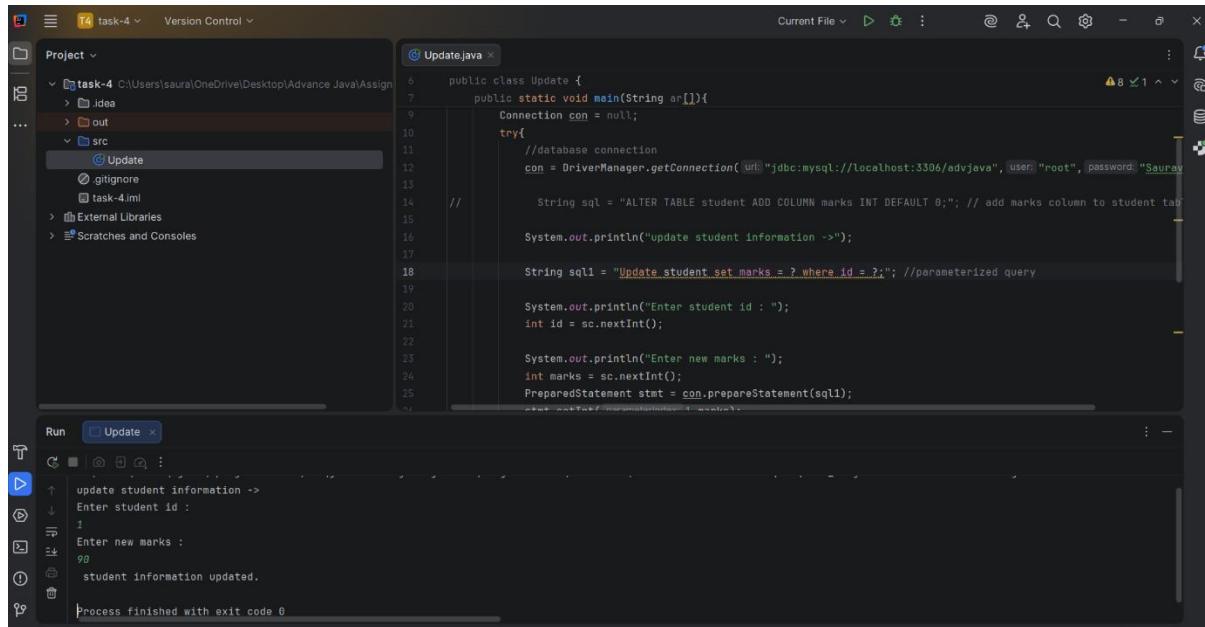
## **Task 4: Updating Existing Records**

**File Name:** Update.java

### **1. Description**

This task performs an update operation. It specifically takes a student's ID and a new value for 'marks', then updates the corresponding record in the database.

### **2. Source Code with output**



The screenshot shows the IntelliJ IDEA interface. The left sidebar displays a project structure for 'task-4' containing a 'src' folder with an 'Update' class. The right pane shows the code for 'Update.java'. The code adds a 'marks' column to a 'student' table and then updates a record. The run output window at the bottom shows the execution of the program, prompting for student ID and new marks, and confirming the update was successful.

```
public class Update {
    public static void main(String args[]){
        Connection con = null;
        try{
            //database connection
            con = DriverManager.getConnection("jdbc:mysql://localhost:3306/advjava", "root", "Saurav");
            String sql = "ALTER TABLE student ADD COLUMN marks INT DEFAULT 0;"; // add marks column to student tab
            System.out.println("update student information ->");
            String sql1 = "Update student set marks = ? where id = ?"; //parameterized query
            System.out.println("Enter student id : ");
            int id = sc.nextInt();
            System.out.println("Enter new marks : ");
            int marks = sc.nextInt();
            PreparedStatement stet = con.prepareStatement(sql1);
            stet.setInt(1, marks);
            stet.setInt(2, id);
            stet.executeUpdate();
            System.out.println("student information updated.");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

### **3. Execution Logic**

- SQL Query:** UPDATE student SET marks = ? WHERE id = ?;
- Input Handling:** Allows dynamic updating by asking the user which ID needs a mark modification.
- Result:** Confirms the update with a success message.

### **4. MySQL Verification**

#### **BEFORE UPDATION**

**select \* from student;**

id	name	age	marks
1	Saurav	21	0
2	Mayur	20	0

```
| 3 | Nikhil | 21 | 0 |
```

```
| 4 | Anand | 5 | 0 |
```

#### AFTER UPDATION

```
select * from student;
```

```
| id | name | age | marks |
```

```
+----+-----+-----+-----+
```

```
| 1 | Saurav | 21 | 90 |
```

```
| 2 | Mayur | 20 | 0 |
```

```
| 3 | Nikhil | 21 | 0 |
```

```
| 4 | Anand | 5 | 0 |
```

---

```
| 5 | Akash | 21 | 0 |
```

## **Task 5:** Deleting Records from Database

## **File Name:** Delete.java

## 1. Description

This task handles the removal of data. It prompts the user for a Student ID and deletes that specific record from the student table.

## 2. Source Code with output

```
task-5 C:\Users\saura\OneDrive\Desktop\Advance Java\Assignment
src
Delete.java
5
6     .util.Scanner;
7
8    s Delete {
9        static void main(String ar[]) throws SQLException{
10            Scanner sc = new Scanner(System.in);
11            Connection con = null;
12            {
13                con = DriverManager.getConnection("jdbc:mysql://localhost:3306/advjava", "root", "Saurav$2004");
14                System.out.println("delete student information ->");
15
16                String sql = "delete from student where id=?";
17
18                System.out.println("enter student id: ");
19                int id = sc.nextInt();
20
21                PreparedStatement stmt = con.prepareStatement(sql);
22                stmt.setInt(1, id);
23
24            }
25        }
26    }
27
28
Run Delete
C:\Users\saura\jdks\openjdk-25.0.1\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.1\lib\idea_rt.jar=50898" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8
delete student information ->
enter student id:
5
student information deleted.

Process finished with exit code 0
```

### 3. Execution Logic

- **Input:** The user provides the id of the student to be removed.
  - **Statement Execution:** The PreparedStatement binds the ID and executes the delete command.
  - **Cleanup:** Closes the connection after the operation.

## 4. MySQL Verification

```
select * from student;  
  
| id | name | age | marks |  
+---+-----+-----+-----+  
| 1 | Saurav | 21 | 90 |  
| 2 | Mayur | 20 | 0 |  
| 3 | Nikhil | 21 | 0 |  
| 4 | Anand | 5 | 0 |
```