

ASSIGNMENT 01 (JDBC)

Q1) 1. Write a program to select specific columns (atleast 2 cols) from customers table with multiple condition(atleast two conditions)

The screenshot shows the Eclipse IDE interface. The top window displays a Java code editor with the following code:

```
File Edit Source Refactor Navigate Search Project Run Window Help
Registration.java Registration.html package-info.java Registration.html Arrays.java Demo1.java module-info.java *Demo1.java package-info.java

1 package Assignment;
2 import java.sql.*;
3 public class Demo1 {
4     public static void main(String[] args) throws Exception {
5         Class.forName("com.mysql.cj.jdbc.Driver");
6         Connection con = DriverManager.getConnection(
7             "jdbc:mysql://localhost:3306/jdbc",
8             "root",
9             "Yash@1234"
10        );
11        if (con != null)
12            System.out.println("Connection Successful!!!");
13        else
14            System.out.println("Connection Unsuccessful!!!");
15        Statement st = con.createStatement();
16        String query = "SELECT cid, cname FROM customers WHERE city = 'hyd' AND balance > 50000";
17        ResultSet rs = st.executeQuery(query);
18        while (rs.next()) {
19            System.out.println(rs.getInt("cid") + " " + rs.getString("cname"));
20        }
21        con.close();
22    }
23 }
24 }
```

The bottom window is the Console view, showing the output of the program:

```
Console X | 
<terminated> Demo1 [Java Application] C:\Users\yadav\Downloads\eclipse-jee-2025-09-R-win32-x86_64\eclipse\plugins\org.eclipse.jst.jdt.core\hotspot.jre.full.win32.x86_64_21.0.8.v20250724-1545
Connection Successful!!!
1 Ram
4 Kiran
```

Q2) 2. Write a program to demonstrate sub queries. Provide customer details who is having maximum balance using subquery.

```
1 package Assignment;
2 import java.sql.*;
3
4 public class Demo2 {
5     public static void main(String[] args) throws Exception {
6         Class.forName("com.mysql.cj.jdbc.Driver");
7
8         Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc", "root", "Y");
9
10        if(con!=null)
11            System.out.print("Connection Successfull!!! \n");
12        else
13            System.out.print("Connection UnSuccessfull!!! \n");
14
15        Statement st = con.createStatement();
16        String query = "select * from customers where balance = (select max(balance) from customers";
17
18        ResultSet rs = st.executeQuery(query);
19        while (rs.next()) {
20            System.out.println(rs.getInt(1) + " " + rs.getString(2));
21        }
22        con.close();
23    }
24
25 }
26 }
```

```
<terminated> Demo2 [Java Application] C:\Users\yadav\Downloads\eclipse-jee-2025-09-R-win32-x86_64\eclipse\plugins\org.eclipse.jst\openjdk.hotspot.jre.full.win32\x86_64_21.0.8.v20250724-1444 Connection Successfull!!!
3 Aman
```

Q3) Write a program to select data from multiple columns of multiple tables. Use customers and student tables to select 2 fields from each table

The screenshot shows the Eclipse IDE interface with the Demo3.java file open in the editor. The code connects to a MySQL database and prints student records to the console. The output window shows the successful connection and the printed student data.

```
1 package Assignment;
2
3 import java.sql.*;
4
5
6 public class Demo3 {
7     public static void main(String[] args) throws Exception {
8         Class.forName("com.mysql.cj.jdbc.Driver");
9
10        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc", "root", "root");
11
12        if(con!=null)
13            System.out.print("Connection Successfull!!! \n");
14        else
15            System.out.print("Connection UnSuccessfull!!! \n");
16
17        Statement st = con.createStatement();
18        String query = "select c.cid, c cname, s.sid, s.sname " +
19                      "from customers c, student s where c.city = s.city";
20
21        ResultSet rs = st.executeQuery(query);
22        while (rs.next()) {
23            System.out.println(rs.getInt(1) + " " + rs.getString(2) + " | "
24                + rs.getInt(3) + " " + rs.getString(4));
25        }
26        con.close();
27    }
28 }
29
```

```
<terminated> Demo3 [Java Application] C:\Users\yadav\Downloads\eclipse-jee-2025-09-R-win32-x86_64\eclipse\plugins\org.eclipse.jst\openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v20250724-1454 Connection Successfull!!!
4 Kiran | 101 Aakash
2 Sita | 101 Aakash
1 Ram | 101 Aakash
3 Aman | 102 Meena
4 Kiran | 104 Sonu
2 Sita | 104 Sonu
1 Ram | 104 Sonu
```

Q4) Program to delete and update a record of student table based on data read from user

```
1 package Assignment;
2 import java.sql.*;
3 import java.util.Scanner;
4 public class Demo4 {
5     public static void main(String[] args) throws Exception {
6         Class.forName("com.mysql.cj.jdbc.Driver");
7         Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc", "root", "y");
8         if (con != null)
9             System.out.println("Connection Successful!!!");
10        else
11            System.out.println("Connection Unsuccessful!!!");
12        Scanner sc = new Scanner(System.in);
13        System.out.print("Enter student id to update name: ");
14        int id = sc.nextInt();
15        System.out.print("Enter new name: ");
16        String name = sc.next();
17        PreparedStatement pst = con.prepareStatement("UPDATE student SET sname = ? WHERE sid = ?");
18        pst.setString(1, name);
19        pst.setInt(2, id);
20        int updated = pst.executeUpdate();
21        System.out.println(updated + " record updated");
22        System.out.print("Enter student id to delete: ");
23        int delId = sc.nextInt();
24        pst = con.prepareStatement("DELETE FROM student WHERE sid = ?");
25        pst.setInt(1, delId);
26        int deleted = pst.executeUpdate();
27        System.out.println(deleted + " record deleted");
28
29        con.close();
}
```

```
<terminated> Demo4 [Java Application] C:\Users\yadav\Downloads\eclipse-jee-2025-09-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.i
Connection Successful!!!
Enter student id to update name: 101
Enter new name: ROHAN
1 record updated
Enter student id to delete: 103
1 record deleted
```

```

mysql> select * from Student;
+----+-----+-----+
| sid | sname | city  |
+----+-----+-----+
| 101 | Aakash | Hyd   |
| 102 | Meena  | Delhi  |
| 103 | Rohit  | Pune   |
| 104 | Sonu   | Hyd   |
+----+-----+-----+
4 rows in set (0.00 sec)

mysql> select * from Student;
+----+-----+-----+
| sid | sname | city  |
+----+-----+-----+
| 101 | ROHAN | Hyd   |
| 102 | Meena  | Delhi  |
| 104 | Sonu   | Hyd   |
+----+-----+-----+
3 rows in set (0.00 sec)

```

Q5) Write program to read multiple student details (10 records) from user and insert them to student table. Use for loop and PreparedStatement

```

7• public static void main(String[] args) throws Exception {
8•     Class.forName("com.mysql.cj.jdbc.Driver");
9•     Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/jdbc", "root", "Y
10•    if (con != null)
11•        System.out.println("Connection Successful!");
12•    else {
13•        System.out.println("Connection Failed!");
14•        return;
15•    }
16•    String query = "insert into student (sid, sname, gpa, city) values (?, ?, ?, ?)";
17•    PreparedStatement pst = con.prepareStatement(query);
18•    Scanner sc = new Scanner(System.in);
19•    for (int i = 8; i <= 10; i++) {
20•        System.out.println("\nEnter details for Student " + i);
21•        System.out.print("Enter ID: ");
22•        int sid = sc.nextInt();
23•        sc.nextLine();
24•        System.out.print("Enter Name: ");
25•        String sname = sc.nextLine();
26•        System.out.print("Enter City: ");
27•        String city = sc.nextLine();
28•        System.out.println("Enter Student marks");
29•        double gpa = sc.nextDouble();
30•        pst.setInt(1, sid);
31•        pst.setString(2, sname);
32•        pst.setDouble(3, gpa);
33•        pst.setString(4, city);
34•        pst.executeUpdate();
35•    }

```

```
Console X Progress i Install Java 25 Support
<terminated> Demo5 [Java Application] C:\Users\DAC-User

Enter details for Student 1
Enter ID: 1
Enter Name: Yash
Enter City: Akola
Enter Student marks
9.9

Enter details for Student 2
Enter ID: 2
Enter Name: Jay
Enter City: MP
Enter Student marks
7.6

Enter details for Student 3
Enter ID: 3
Enter Name: Raj
Enter City: Tukuguda
Enter Student marks
7.7

Enter details for Student 4
Enter ID: 4
Enter Name: Satvik
Enter City: Lucknow
Enter Student marks
5.5

Enter details for Student 5
Enter ID: 5
Enter Name: Harsh
Enter City: Gorakhpur
Enter Student marks
7.8

Enter details for Student 6
Enter ID: 6
Enter Name: Priya
Enter City: Raibarily
Enter Student marks
6.6
```

```

terminated> Demo>java Application C:\Users\DAC-User3\.p2\pool\plugins\org.eclipse.jdt
Connection Successful!

Enter details for Student 8
Enter ID: 8
Enter Name: Rohit
Enter City: Jaipur
Enter Student marks
9.9

Enter details for Student 9
Enter ID:
9
Enter Name: Isha
Enter City: Akola
Enter Student marks
5.67

Enter details for Student 10
Enter ID: 10
Enter Name: Ram
Enter City: Ayodhya
Enter Student marks
10.00

All 10 records inserted successfully!

```

```

mysql> select * from student;
+----+-----+-----+-----+
| sid | sname | gpa | city
+----+-----+-----+-----+
| 1  | Yash  | 9.9 | Akola
| 2  | Jay   | 7.6 | MP
| 3  | Raj   | 7.7 | Tukuguda
| 4  | Satvik| 5.5 | Lucknow
| 5  | Harsh | 7.8 | Gorakhpur
| 6  | Priya | 6.6 | Raibarily
| 7  | tejas | 7.4 | Mumbai
| 8  | Rohit | 9.9 | Jaipur
| 9  | Isha  | 5.67 | Akola
| 10 | Ram   | 10  | Ayodhya
+----+-----+-----+-----+
10 rows in set (0.00 sec)

```

Q6) 6. Demonstrate select query using Prepared Statement by reading data from user(Read gpa and provide details of students whose gpa less than that entered)

```
7 public class Demo6 {  
8•     public static void main(String[] args) throws Exception {  
9         Class.forName("com.mysql.cj.jdbc.Driver");  
10        Connection con = DriverManager.getConnection(  
11                "jdbc:mysql://localhost:3306/jdbc", "root", "Yash@1234");  
12  
13        if (con != null)  
14            System.out.println("Connection Successful!");  
15•    else {  
16            System.out.println("Connection Failed!");  
17            return;  
18        }  
19        String query = "SELECT sid, sname, gpa, city FROM student WHERE gpa < ?";  
20        PreparedStatement pst = con.prepareStatement(query);  
21        Scanner sc = new Scanner(System.in);  
22        System.out.print("\nEnter GPA value: ");  
23        double inputGpa = sc.nextDouble();  
24        pst.setDouble(1, inputGpa);  
25        ResultSet rs = pst.executeQuery();  
26        System.out.println("\nStudents with GPA less than " + inputGpa + ":");  
27        System.out.println("-----");  
28        boolean found = false;  
29•    while (rs.next()) {  
30            found = true;  
31            System.out.println("ID : " + rs.getInt("sid"));  
32            System.out.println("Name : " + rs.getString("sname"));  
33            System.out.println("GPA : " + rs.getDouble("gpa"));  
34            System.out.println("City : " + rs.getString("city"));  
-----  
Connection Successful!  
  
Enter GPA value: 8.8  
  
Students with GPA less than 8.8:  
-----  
ID : 101  
Name : Aakash  
GPA : 8.5  
City : Hyd  
-----  
ID : 103  
Name : Rohit  
GPA : 7.8  
City : Pune  
-----
```

Q7) Write a program to perform jdbc batch processing -

a. update customers balance (add 10000) where city is hyd and pune

b. Delete customers if balance is less than 50000

c & d. insert 2 customer records -

e. update customers with balance with + 15000 if cid between 3000 and 7000

```
9•     public static void main(String[] args) throws Exception {
10         Class.forName("com.mysql.cj.jdbc.Driver");
11         Connection con = DriverManager.getConnection(
12             "jdbc:mysql://localhost:3306/jdbc", "root", "Yash@1234");
13         if (con != null)
14             System.out.println("Connection Successful!");
15•     else {
16         System.out.println("Connection Failed!");
17         return;
18     }
19     Statement stmt = con.createStatement();
20     stmt.addBatch(
21         "UPDATE customer SET balance = balance + 10000 "
22         + "WHERE city IN ('hyd', 'pune')"
23     );
24     stmt.addBatch(
25         "DELETE FROM customer WHERE balance < 50000"
26     );
27     stmt.addBatch(
28         "INSERT INTO customer (cid, cname, balance, city) "
29         + "VALUES (2010, 'Rohit', 75000, 'mumbai')"
30     );
31     stmt.addBatch(
32         "INSERT INTO customer (cid, cname, balance, city) "
33         + "VALUES (2011, 'Neha', 82000, 'pune')"
34     );
35     stmt.addBatch(
36         "UPDATE customer SET balance = balance + 15000 "
37         + "WHERE cid BETWEEN 3000 AND 7000"
+
            "INSERT INTO customer (cid, cname, balance, city) "
            + "VALUES (2011, 'Neha', 82000, 'pune')"
);
38     stmt.addBatch(
39         "UPDATE customer SET balance = balance + 15000 "
40         + "WHERE cid BETWEEN 3000 AND 7000"
);
41     int results[] = stmt.executeBatch();
42
43     System.out.println("\nBatch Processing Completed!");
44•     for (int i = 0; i < results.length; i++) {
45         System.out.println("Statement " + (i + 1) + " executed. Result: " + results[i]);
46     }
47     stmt.close();
48     con.close();
49 }
```

```
Connection Successful!
```

```
Batch Processing Completed!
Statement 1 executed. Result: 4
Statement 2 executed. Result: 0
Statement 3 executed. Result: 1
Statement 4 executed. Result: 1
Statement 5 executed. Result: 3
```

```
mysql> select * from customer;
```

cid	cname	balance	city
1001	Aakash	45000	hyd
1002	Meena	60000	pune
2001	Suresh	80000	delhi
3005	Ramesh	55000	mumbai
4500	Neeraj	40000	hyd
6500	Pooja	70000	pune
9001	Vijay	90000	chennai

```
7 rows in set (0.00 sec)
```

```
mysql> select * from customer;
```

cid	cname	balance	city
1001	Aakash	55000	hyd
1002	Meena	70000	pune
2001	Suresh	80000	delhi
2010	Rohit	75000	mumbai
2011	Neha	82000	pune
3005	Ramesh	70000	mumbai
4500	Neeraj	65000	hyd
6500	Pooja	95000	pune
9001	Vijay	90000	chennai

```
9 rows in set (0.00 sec)
```

Q8) 8. Demonstrate transaction management - with student table (take suitable case study)

```
1 package Assignment;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.PreparedStatement;
6 import java.sql.SQLException;
7
8 public class Demo8 {
9
10•    public static void main(String[] args) {
11        Connection con = null;
12        PreparedStatement pst = null;
13
14•        try {
15            // 1. Load driver and connect
16            Class.forName("com.mysql.cj.jdbc.Driver");
17            con = DriverManager.getConnection(
18                "jdbc:mysql://localhost:3306/jdbc", "root", "Yash@1234");
19
20            if (con != null)
21                System.out.println("Connection Successful!");
22
23            // 2. Turn off auto-commit
24            con.setAutoCommit(false);
25
26            // 3. Prepare update statement
27            String updateQuery = "UPDATE student SET gpa = ? WHERE sid = ?";
28            pst = con.prepareStatement(updateQuery);
29
30
31            con.commit();
32            System.out.println("\nTransaction committed successfully!");
33        } catch (Exception e) {
34            System.out.println("\nError occurred! Rolling back transaction...");
35            try {
36                if (con != null) con.rollback();
37            } catch (SQLException ex) {
38                ex.printStackTrace();
39            }
40            e.printStackTrace();
41        } finally {
42            try {
43                if (pst != null) pst.close();
44                if (con != null) con.close();
45            } catch (SQLException e) {
46                e.printStackTrace();
47            }
48        }
49    }
50}
```

Connection Successful!

Transaction committed successfully!

```

mysql> select * from student;
+----+-----+-----+-----+
| sid | sname | gpa  | city  |
+----+-----+-----+-----+
| 101 | Aakash | 9.2  | Hyd   |
| 102 | Meena  | 8.5  | Delhi |
| 103 | Rohit  | 9     | Pune  |
| 104 | Sonu   | 8.9  | Hyd   |
| 105 | Kiran  | 9.4  | Mumbai|
+----+-----+-----+-----+
5 rows in set (0.01 sec)

```

Q9) 9. program to Read data from a text file (student.txt) and insert into student table(Use prepared statement)

```

1 package Assignment;
2
3 import java.io.BufferedReader;
4 import java.io.FileReader;
5 import java.sql.Connection;
6 import java.sql.DriverManager;
7 import java.sql.PreparedStatement;
8
9 public class Demo9 {
10
11    public static void main(String[] args) {
12        String fileName = "D:\\CDAC Hyderabad\\bin\\src\\java\\AssignmentJDBC\\src\\Assignment\\stud-
13        Connection con = null;
14        PreparedStatement pst = null;
15        BufferedReader br = null;
16        try {
17            Class.forName("com.mysql.cj.jdbc.Driver");
18            con = DriverManager.getConnection(
19                "jdbc:mysql://localhost:3306/jdbc", "root", "Yash@1234");
20            System.out.println("Connection Successful!");
21            String query = "INSERT INTO student (sid, sname, gpa, city) VALUES (?, ?, ?, ?)";
22            pst = con.prepareStatement(query);
23            br = new BufferedReader(new FileReader(fileName));
24            String line;
25            int count = 0;
26            while ((line = br.readLine()) != null) {
27                String[] data = line.split(",");
28                if (data.length != 4) continue;
29

```

```

27         String[] data = line.split(",");
28         if (data.length != 4) continue;
29
30         int sid = Integer.parseInt(data[0].trim());
31         String sname = data[1].trim();
32         double gpa = Double.parseDouble(data[2].trim());
33         String city = data[3].trim();
34         pst.setInt(1, sid);
35         pst.setString(2, sname);
36         pst.setDouble(3, gpa);
37         pst.setString(4, city);
38         pst.executeUpdate();
39         count++;
40     }
41     System.out.println("\n" + count + " records inserted successfully!");
42 } catch (Exception e) {
43     e.printStackTrace();
44 } finally {
45     try {
46         if (br != null) br.close();
47         if (pst != null) pst.close();
48         if (con != null) con.close();
49     } catch (Exception e) {
50         e.printStackTrace();
51     }
52 }
53 }
54 }
55 }
```

```

1 sid,sname,gpa,city
2 101,Yash Yadav,9.5,Hyderabad
3 102,Neha,9.2,Pune
4 103,Manoj,5.8,Mumbai
5 |
```

Q10)Program to demonstrate where, group by, order by, having with customers table

(find customers group by city and provide details)

(find rich customer group by city)

(Use aggregate functions)

```

1 package Assignment;
2 import java.sql.*;
3
4 public class Demo10 {
5•     public static void main(String[] args) throws Exception {
6
7     Class.forName("com.mysql.cj.jdbc.Driver");
8     Connection con = DriverManager.getConnection(
9         "jdbc:mysql://localhost:3306/jdbc", "root", "Yash@1234"
10    );
11
12     Statement stmt = con.createStatement();
13     System.out.println("\n GROUP BY CITY");
14     ResultSet rs1 = stmt.executeQuery(
15         "SELECT city, COUNT(*) AS total_customers, SUM(balance) AS total_balance " +
16         "FROM customers GROUP BY city"
17    );
18
19•     while (rs1.next()) {
20         System.out.println(
21             "City: " + rs1.getString("city") +
22             " | Customers: " + rs1.getInt("total_customers") +
23             " | Total Balance: " + rs1.getDouble("total_balance")
24        );
25    }

```

```

System.out.println("\n RICH CUSTOMERS CITY WISE (balance > 60000) ");
ResultSet rs2 = stmt.executeQuery(
    "SELECT city, COUNT(*) AS rich_count, AVG(balance) AS avg_rich_balance " +
    "FROM customers WHERE balance > 60000 GROUP BY city"
);

```

```

while (rs2.next()) {
    System.out.println(
        "City: " + rs2.getString("city") +
        " | Rich Count: " + rs2.getInt("rich_count") +
        " | Avg Rich Balance: " + rs2.getDouble("avg_rich_balance")
    );
}

```

```

System.out.println("\nCITIES WITH TOTAL BALANCE > 1,00,000 ");
ResultSet rs3 = stmt.executeQuery(
    "SELECT city, SUM(balance) AS total_balance " +
    "FROM customers GROUP BY city HAVING SUM(balance) > 100000"
);

```

```

while (rs3.next()) {
    System.out.println(
        "City: " + rs3.getString("city") +
        " | Total Balance: " + rs3.getDouble("total_balance")
    );
}

```

```

System.out.println("\n== CUSTOMERS ORDER BY BALANCE DESC ==");

```

```

ResultSet rs4 = stmt.executeQuery(
    "SELECT * FROM customers ORDER BY balance DESC"
);
while (rs4.next()) {
    System.out.println("ID: " + rs4.getInt("cid"));
    System.out.println("Name: " + rs4.getString("cname"));
    System.out.println("Balance: " + rs4.getDouble("balance"));
    System.out.println("City: " + rs4.getString("city"));
    System.out.println("-----");
}

```

```

    con.close();
}

```

```
<terminated> Demo10 [Java Application] C:\Users\yadav\Downloads\eclipse-jee-2023-09-R-win32-x86_64\eclipse\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v20280724-1412\jre\bin\javaw.exe [Dec 1, 2023 9:59:58 PM - 9:59:58 PM elapsed: 0:00:01.742] [pid: 20944]
GROUP BY CITY
City: pune | Customers: 3 | Total Balance: 175000.0
City: mumbai | Customers: 2 | Total Balance: 137000.0
City: hyd | Customers: 2 | Total Balance: 125000.0
City: delhi | Customers: 1 | Total Balance: 78000.0

RICH CUSTOMERS CITY WISE (balance > 60000)
City: pune | Rich Count: 1 | Avg Rich Balance: 70000.0
City: mumbai | Rich Count: 1 | Avg Rich Balance: 82000.0
City: hyd | Rich Count: 1 | Avg Rich Balance: 95000.0
City: delhi | Rich Count: 1 | Avg Rich Balance: 78000.0

CITIES WITH TOTAL BALANCE > 1,00,000
City: pune | Total Balance: 175000.0
City: mumbai | Total Balance: 137000.0
City: hyd | Total Balance: 125000.0

==== CUSTOMERS ORDER BY BALANCE DESC ====
ID: 106
Name: Kiran
Balance: 95000.0
City: hyd
-----
ID: 103
Name: Neha
Balance: 82000.0
City: mumbai
```

```
<terminated> Demo10 [Java Application] C:\Users\yadav\Downloads\eclipse-jee-2023-09-R-win32-x86_64\eclipse\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v20280724-1412\jre\bin\javaw.exe [Dec 1, 2023 9:59:58 PM - 9:59:58 PM elapsed: 0:00:01.742] [pid: 20944]
City: delhi
-----
ID: 102
Name: Rohit
Balance: 70000.0
City: pune
-----
ID: 107
Name: Meena
Balance: 60000.0
City: pune
-----
ID: 104
Name: Suresh
Balance: 55000.0
City: mumbai
-----
ID: 101
Name: Amit
Balance: 45000.0
City: pune
-----
ID: 105
Name: Pooja
Balance: 30000.0
City: hyd
```