

Insert user details, then finally tables be like:

The screenshot shows the MySQL Workbench interface. On the left, the Navigator pane displays the database schema, specifically the 'mysql_jdb' schema which contains a single table named 'user_details'. The main area is the 'Query 1' editor, which contains the SQL query: '1 • select * from User_Details;'. Below the editor is the 'Result Grid' showing the data from the 'user_details' table:

	UserName	Score	Email
FirstUserName	23.45	FUN123@gmail.com	
SecondUserName	123.45	sUN123@gmail.com	
ThirdUserName	678.91	TUN123@gmail.com	
ForthUserName	987.65	FRUN123@gmail.com	
FifthUserName	543.21	FIUN123@gmail.com	

Initially we have the database like:

UserName	Score	Email
FirstUserName	23.45	FUN123@gmail.com
SecondUserName	123.45	sUN123@gmail.com
ThirdUserName	678.91	TUN123@gmail.com
ForthUserName	987.65	FRUN123@gmail.com
FifthUserName	543.21	FIUN123@gmail.com

For updating cell in Table:

```
// Updating within table
    float score = 90f; String email =
"TUN123@gmail.com";
    PreparedStatement ps = ctn.prepareStatement("update
User_Details set Score=? where Email=?");
    ps.setFloat(1, score);
    ps.setString(2, email);
    int eu = ps.executeUpdate();
    if (eu > 0) System.out.println("Updated " + score +
" at email " + email);
    else System.out.println("Update at email " + email
+ " got failed.");
```

```
    ctn.close();
```

Run the code:

The screenshot shows the Eclipse IDE interface during the execution of a Java application. The top part displays the Java code being run, specifically the line `ctn.close();`. The code is part of a larger method that updates a database record. The bottom part shows the Eclipse Console tab, which displays the output of the application's execution, indicating that the score was updated to 90.0.

```
13 Connection ctn = Dr
14
15 // Updating within
16 float score = 90f; S
17 PreparedStatement ps
18 ps.setFloat(1, score)
19 ps.setString(2, email)
20 int eu = ps.executeUpdate()
21 if (eu > 0) System.e
22 else System.out.prin
23     ctn.close();
24 }
25
```

Problems Javadoc Declaration Console X
<terminated> _02 CRUD JDBC [Java Application] C:\Users\kiran\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.w

Updated 90.0 at email TUN123@gmail.com

Click as shown by arrow:

The screenshot shows the MySQL Workbench interface. On the left is the Navigator pane with 'SCHEMAS' expanded to show 'mysql_jdb' containing 'Tables', 'Views', 'Stored Procedures', and 'Functions'. One table named 'user_details' is selected. The main area is the 'Query 1' editor with the SQL command: '1 • select * from User_Details;'. Below the editor is the 'Result Grid' showing the following data:

	UserName	Score	Email
▶	FirstUserName	23.45	FUN123@gmail.com
	SecondUserName	123.45	sUN123@gmail.com
	ThirdUserName	678.91	TUN123@gmail.com
	ForthUserName	987.65	FRUN123@gmail.com
	FifthUserName	543.21	FIUN123@gmail.com

Then the score gets updated as expected without using MySQL workbench:

	UserName	Score	Email
▶	FirstUserName	23.45	FUN123@gmail.com
	SecondUserName	123.45	sUN123@gmail.com
	ThirdUserName	90.00	TUN123@gmail.com
	ForthUserName	987.65	FRUN123@gmail.com
	FifthUserName	543.21	FIUN123@gmail.com

For **Deleting** the row based on email check:

```
// Deleting row based on email
String email = "TUN123@gmail.com";
PreparedStatement ps = ctn.prepareStatement("delete from User_Details where Email=?");
ps.setString(1, email);
int eu = ps.executeUpdate();
if (eu > 0) System.out.println("Deleted at email " + email);
else System.out.println("Not Deleted at email " + email);
ctn.close();
```

Run the code:

```
> _03_Delete_C
  > src/main/resources
  > src/test/java
  > src/test/resources
> JRE System Library [JavaSE-1.8]
> Maven Dependencies [Maven Dependencies]
> src
  > target
  pom.xml

15    // Deleting row based on email
16    String email = "TUN123@gmail.com";
17    PreparedStatement ps = ctn.prepareStatement();
18    ps.setString(1, email);
19    int eu = ps.executeUpdate();
20    if (eu > 0) System.out.println("Deleted at email " + email);
21    else System.out.println("Not Deleted");
22    ctn.close();
23
24
25
26
```

Problems Javadoc Declaration Console Outline Task List

<terminated> _03_Delete CRUD JDBC [Java Application] C:\Users\kiran\p2\pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full

Deleted at email TUN123@gmail.com

Query 1

The screenshot shows the MySQL Workbench interface. A red arrow points to the execute icon (a yellow lightning bolt) in the toolbar above the query editor. The query editor contains the SQL command: `1 select * from User_Details;`

Result Grid | Filter Rows: Export: | Wrap Cell Content:

	UserName	Score	Email
▶	FirstUserName	23.45	FUN123@gmail.com
	SecondUserName	123.45	sUN123@gmail.com
	ThirdUserName	90.00	TUN123@gmail.com
	ForthUserName	987.65	FRUN123@gmail.com
	FifthUserName	543.21	FIUN123@gmail.com

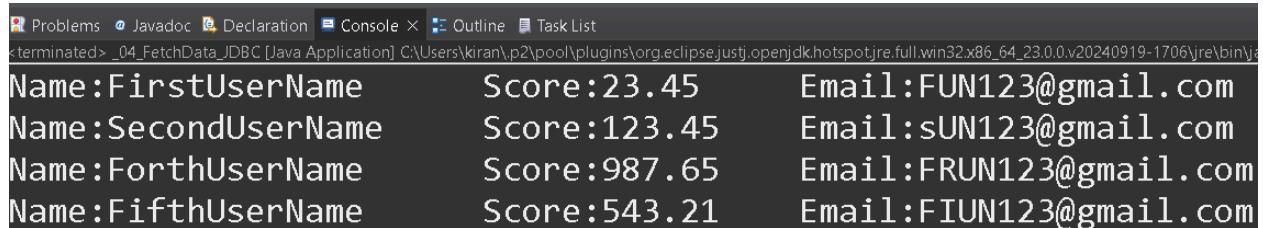
	UserName	Score	Email
	FirstUserName	23.45	FUN123@gmail.com
	SecondUserName	123.45	sUN123@gmail.com
	ForthUserName	987.65	FRUN123@gmail.com
	FifthUserName	543.21	FIUN123@gmail.com

Similarly, Fetching the data from database:

```
Connection ctn = DriverManager.getConnection(url, user, pwd);

PreparedStatement ps = ctn.prepareStatement("select * from User_Details");
ResultSet rs = ps.executeQuery();
while (rs.next()) {
    System.out.print("Name:" + rs.getString("UserName"));
    System.out.print("\tScore:" + rs.getString("Score"));
    System.out.println("\tEmail:" + rs.getString("Email"));
}
ctn.close();
```

Then output looks like:



The screenshot shows the Eclipse IDE interface with the 'Console' tab selected. The output window displays the results of the JDBC query execution. The output is as follows:

Name	Score	Email
FirstUserName	23.45	FUN123@gmail.com
SecondUserName	123.45	SUN123@gmail.com
ForthUserName	987.65	FRUN123@gmail.com
FifthUserName	543.21	FIUN123@gmail.com