

# JDBC ASSISTED PRACTICE PROJECTS:

## 1. Set up a JDBC Environment

The image consists of three vertically stacked screenshots illustrating the setup of a JDBC environment.

**Top Screenshot:** A terminal window titled "Terminal - root@ip-172-31-29-226". It shows the following commands being run:

```
mittalsonal04gm@ip-172-31-29-226:~$ sudo su -
root@ip-172-31-29-226:~#
root@ip-172-31-29-226:~# mysql -u root -p
Enter password: [REDACTED]
```

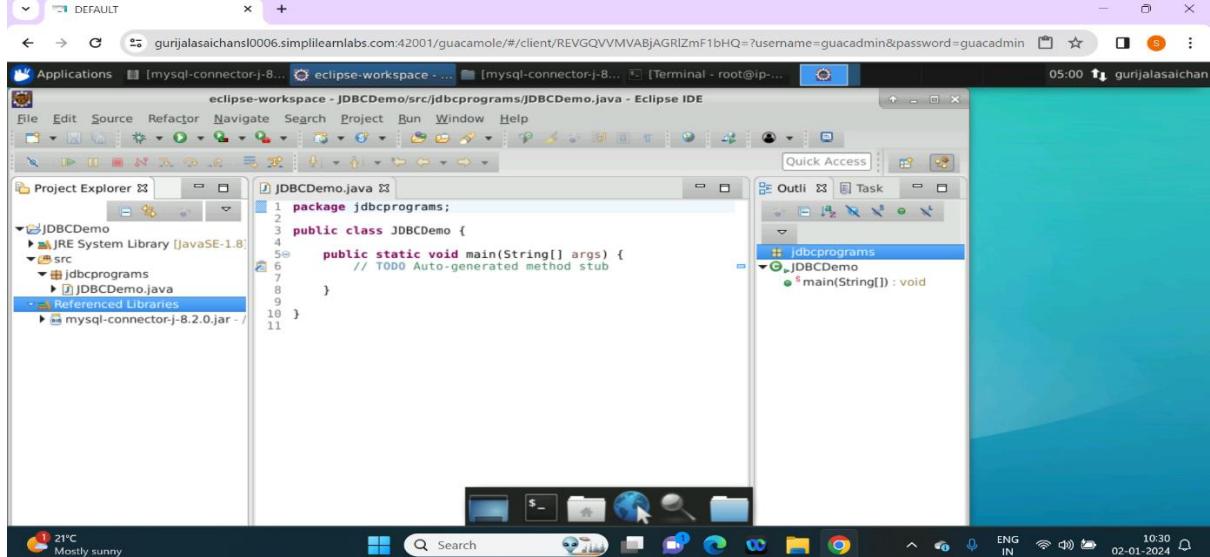
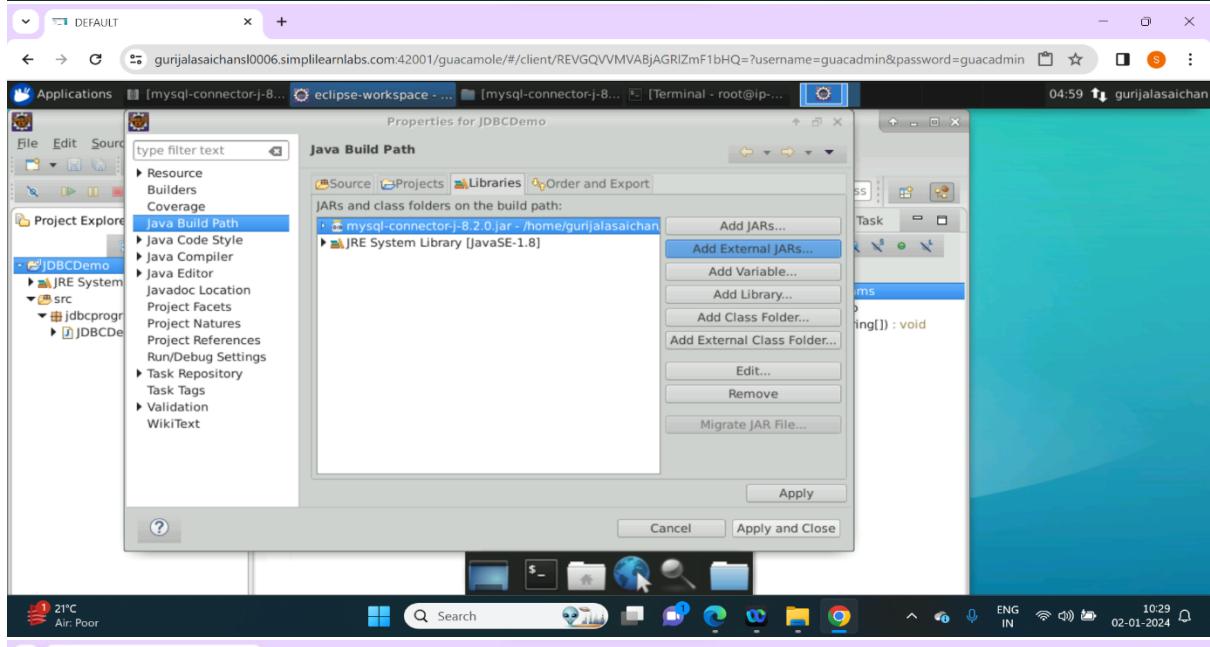
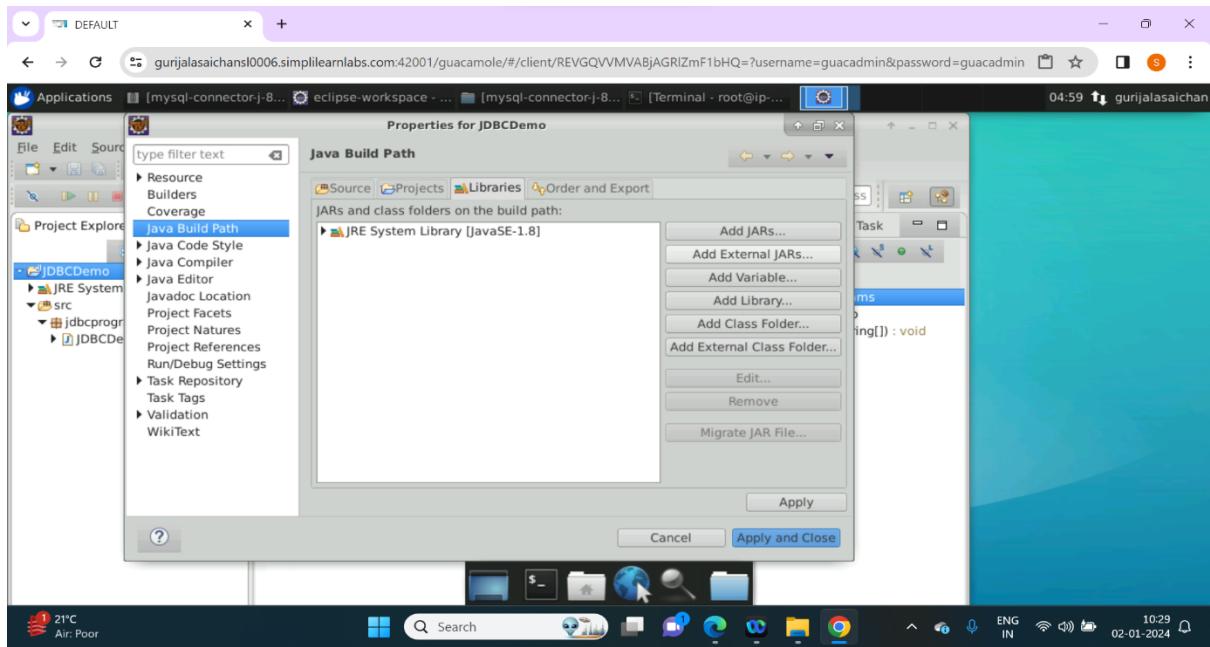
**Middle Screenshot:** A file manager window titled "mysql-connector-j-8.2.0 - File Manager". It displays the contents of the "mysql-connector-j-8.2.0" directory, which includes:

- DEVICES
- File System
- thinclient\_dr...
- src
- build.xml
- </>
- CHANGES
- INFO\_BIN
- INFO\_SRC
- LICENSE
- mysql-connector-j-8.2.0.jar
- README

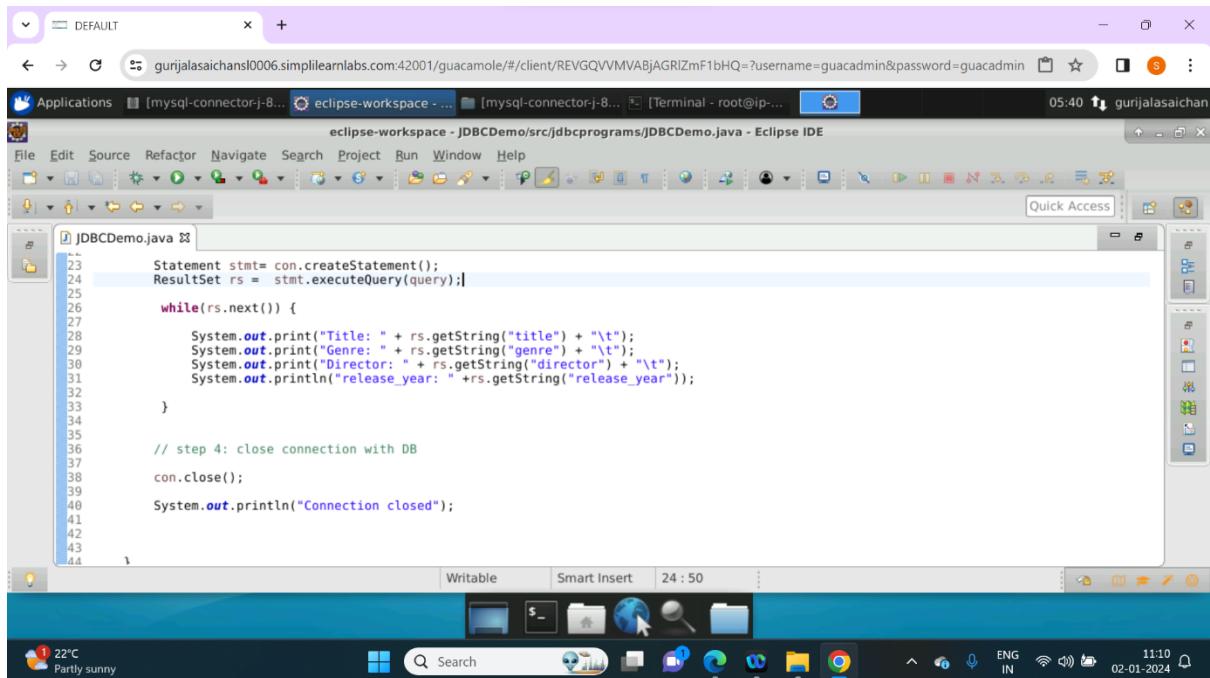
**Bottom Screenshot:** An Eclipse IDE window titled "eclipse-workspace - JDBC Demo/src/jdbcprograms/JDBC Demo - Eclipse IDE". The code editor shows the following Java code:

```
programs;
JDBC Demo {
    static void main(String[] args) {
        // TODO Auto-generated method stub
    }
}
```

The "Project" tool window on the left shows a "Build Path" context menu open over the "JDBC" project. Other options visible in the menu include New, Go Into, Copy, Paste, Delete, Remove from Context, Import..., Export..., Refresh, Close Project, Close Unrelated Project, Show in Remote Systems view, Validate, Coverage As, and Run As.

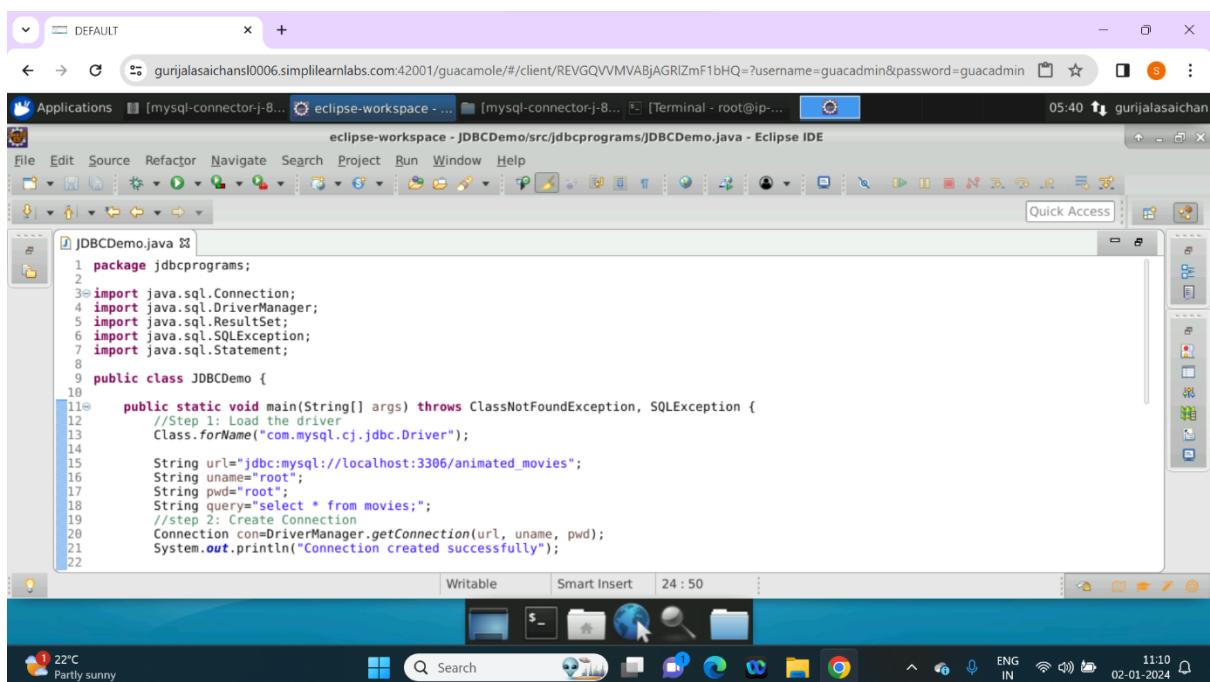


## 2. Demonstrate Connection, Statement, and ResultSet in JDBC.



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC Demo/src/jdbcprograms/JDBC Demo.java - Eclipse IDE". The code editor displays the following Java code:

```
1 package jdbcprograms;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8
9 public class JDBC Demo {
10
11     public static void main(String[] args) throws ClassNotFoundException, SQLException {
12         //Step 1: Load the driver
13         Class.forName("com.mysql.cj.jdbc.Driver");
14
15         String url="jdbc:mysql://localhost:3306/animated_movies";
16         String uname="root";
17         String pwd="root";
18         String query="select * from movies;";
19         //step 2: Create Connection
20         Connection con=DriverManager.getConnection(url, uname, pwd);
21         System.out.println("Connection created successfully");
22     }
23
24     Statement stmt= con.createStatement();
25     ResultSet rs = stmt.executeQuery(query);
26
27     while(rs.next()) {
28
29         System.out.print("Title: " + rs.getString("title") + "\t");
30         System.out.print("Genre: " + rs.getString("genre") + "\t");
31         System.out.print("Director: " + rs.getString("director") + "\t");
32         System.out.println("release_year: " +rs.getString("release_year"));
33     }
34
35
36     // step 4: close connection with DB
37
38     con.close();
39
40     System.out.println("Connection closed");
41
42
43
44 }
```



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC Demo/src/jdbcprograms/JDBC Demo.java - Eclipse IDE". The code editor displays the following Java code:

```
1 package jdbcprograms;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8
9 public class JDBC Demo {
10
11     public static void main(String[] args) throws ClassNotFoundException, SQLException {
12         //Step 1: Load the driver
13         Class.forName("com.mysql.cj.jdbc.Driver");
14
15         String url="jdbc:mysql://localhost:3306/animated_movies";
16         String uname="root";
17         String pwd="root";
18         String query="select * from movies;";
19         //step 2: Create Connection
20         Connection con=DriverManager.getConnection(url, uname, pwd);
21         System.out.println("Connection created successfully");
22     }
23
24     Statement stmt= con.createStatement();
25     ResultSet rs = stmt.executeQuery(query);
26
27     while(rs.next()) {
28
29         System.out.print("Title: " + rs.getString("title") + "\t");
30         System.out.print("Genre: " + rs.getString("genre") + "\t");
31         System.out.print("Director: " + rs.getString("director") + "\t");
32         System.out.println("release_year: " +rs.getString("release_year"));
33     }
34
35
36     // step 4: close connection with DB
37
38     con.close();
39
40     System.out.println("Connection closed");
41
42
43
44 }
```

A screenshot of the Eclipse IDE interface. The central workspace shows a Java file named `JDBCIntro.java` with code demonstrating JDBC operations. The right-hand margin displays the execution output of the code, showing the connection creation and the results of a query on a `movies` table. The output includes the titles, genres, directors, and release years of movies like `Inside Out`, `Toy Story 4`, etc.

```
String uname="root";
String pwd="root";
String query="select * from movies;";
//step 2: Create Connection
Connection con=DriverManager.getConnection(url, uname, pwd);
System.out.println("Connection created successfully");
Statement stmt= con.createStatement();
ResultSet rs = stmt.executeQuery(query);
while(rs.next()) {
    System.out.print("Title: " + rs.getString("title") + " ");
    System.out.print("Genre: " + rs.getString("genre") + " ");
    System.out.print("Director: " + rs.getString("director") );
    System.out.println("release_year: " +rs.getString("rel"));
}
// step 4: close connection with DB
```

<terminated> JDBCIntro [Java Application] /usr/lib/vm/java-8-openjdk-amd64/bin/java [jdb] Connection created successfully  
Title: Inside Out    Genre: Comedy    Director: Pete Doctor    release\_year: 2015  
Title: Toy Story 4    Genre: Comedy    Director: Josh Cooley    release\_year: 1995  
Connection closed

### 3. Demonstrate stored procedures and exception handling in JDBC.

A screenshot of a terminal window titled `Terminal - root@ip-172-31-23-233: ~`. The window displays a MySQL session where a stored procedure named `SelectAllMovies()` is being created. The command entered is:

```
mysql> Create procedure SelectAllMovies()
-> begin
-> select * from movies;
-> end$$
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC...". The code editor displays the following Java code:

```
1 package jdbcprograms;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8
9 public class ProcedureDemo {
10     public static void main(String[] args) throws ClassNotFoundException, SQLException {
11         //Step 1: Load the driver
12         Class.forName("com.mysql.cj.jdbc.Driver");
13
14         String url="jdbc:mysql://localhost:3306/animated_movies";
15         String uname="root";
16         String pwd="root";
17         String query="call SelectALLMovies();";
18
19         //step 2: Create Connection
20         Connection con=DriverManager.getConnection(url, uname, pwd);
21         System.out.println("Connection created successfully");
22     }
23 }
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC...". The code editor displays the following Java code, continuing from the previous screenshot:

```
13
14     String url="jdbc:mysql://localhost:3306/animated_movies";
15     String uname="root";
16     String pwd="root";
17     String query="call SelectALLMovies();";
18
19     //step 2: Create Connection
20     Connection con=DriverManager.getConnection(url, uname, pwd);
21     System.out.println("Connection created successfully");
22
23     Statement stmt= con.createStatement();
24     ResultSet rs = stmt.executeQuery(query);
25
26     while(rs.next()) {
27
28         System.out.print("Title: " + rs.getString("title") + "\t");
29         System.out.print("Genre: " + rs.getString("genre") + "\t");
30         System.out.print("Director: " + rs.getString("director") + "\t");
31         System.out.println("release_year: " +rs.getString("release_year"));
32     }
33 }
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC...". The code editor displays the following Java code, continuing from the previous screenshot:

```
34
35     // step 4: close connection with DB
36     con.close();
37
38     System.out.println("Connection closed");
39
40 }
41
42 }
43 }
44 }
45 }
```

The screenshot displays three Eclipse IDE windows side-by-side, each showing a different Java file:

- Top Window:** Shows the `ProcedureDemo.java` file. The code prints movie details from a database:

```
26     System.out.print("Title: " + rs.getString("title") + "\t");
27     System.out.print("Genre: " + rs.getString("genre") + "\t");
28     System.out.print("Director: " + rs.getString("director") + "\t");
29     System.out.println("Release Year: " + rs.getString("release_year"));
30
31 }
32
33 // step 4: close connection
34
35 con.close();
36
37 System.out.println("Connection closed");
```

- Middle Window:** Shows the `JDBCExceptionDemo.java` file. It contains a main method with a SQL injection vulnerability:

```
1 package jdbcprograms;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8
9
10 public class JDBCExceptionDemo {
11     public static void main(String[] args) throws SQLException {
12         try {
13             String dbURL = "jdbc:mysql://localhost:3306/animated_movies";
14             String username = "root";
15             String password = "root";
16             String query = "call SelectALLMovies()";
17
18             Connection con = DriverManager.getConnection(dbURL,username,password);
19             Statement stmt = con.createStatement();
20             ResultSet rs = stmt.executeQuery(query);
21
22             while(rs.next()) {
23                 System.out.print("Title: " + rs.getString("title") + "\t");
24                 System.out.print("Genre: " + rs.getString("genre") + "\t");
25                 System.out.print("Director: " + rs.getString("director") + "\t");
26                 System.out.println("Release Year: " + rs.getString("release_year"));
27             }
28
29         } catch (SQLException e) {
30             e.printStackTrace();
31         }
32     }
33 }
```

- Bottom Window:** Shows the `JDCCDemo.java` file. It registers a JDBC driver and executes a query:

```
1 package jdbcprograms;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8
9
10 public class JDCCDemo {
11     public static void main(String[] args) throws SQLException {
12         String password = "root";
13         String query = "call SelectALLMovies()";
14
15         try {
16             Connection con = DriverManager.getConnection(dbURL,username,password);
17
18             Class.forName("com.mysql.cj.jdbc.Driver");
19             System.out.println("Registered the JDBC Driver");
20
21             Statement stmt = con.createStatement();
22             ResultSet rs = stmt.executeQuery(query);
23
24             while(rs.next()) {
25                 System.out.print("Title: " + rs.getString("title") + "\t");
26                 System.out.print("Genre: " + rs.getString("genre") + "\t");
27                 System.out.print("Director: " + rs.getString("director") + "\t");
28                 System.out.println("Release Year: " + rs.getString("release_year"));
29             }
30
31         } catch (SQLException e) {
32             e.printStackTrace();
33         }
34     }
35 }
```

The screenshot shows the Eclipse IDE interface with two tabs open: 'JDBCExceptionDemo.java' and 'Console'. The code in 'JDBCExceptionDemo.java' is as follows:

```
32
33     ResultSet rs = stmt.executeQuery(query);
34
35     while(rs.next()) {
36         System.out.print("Title: " + rs.getString("title") + "\t");
37         System.out.print("Genre: " + rs.getString("genre") + "\t");
38         System.out.print("Director: " + rs.getString("director") + "\t");
39         System.out.println("release_year: " + rs.getString("release_year"));
40     }
41 } catch (Exception e) {
42     e.printStackTrace();
43 }
44
45
46
47
48
49 }
50
51 }
52 }
```

The 'Console' tab shows the output of the program:

```
<terminated> jdbcExceptionDemo [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/b
Registered JDBC Driver
Title: Inside Out      Genre: Comedy   Director: Pete Doctor   release_ye
Title: Toy Story 4      Genre: Comedy   Director: Josh Cooley   release_ye
```

This screenshot is similar to the one above, but a tooltip is visible over the 'Console' tab header, reading 'Access commands and other items (Ctrl+3)'. The code and console output are identical to the first screenshot.

The screenshot shows a terminal window with the following MySQL session:

```
File Edit View Terminal Tabs Help
Terminal - root@ip-172-31-23-233: ~
mysql> delimiter $$

mysql>
mysql> use animated_movies $$

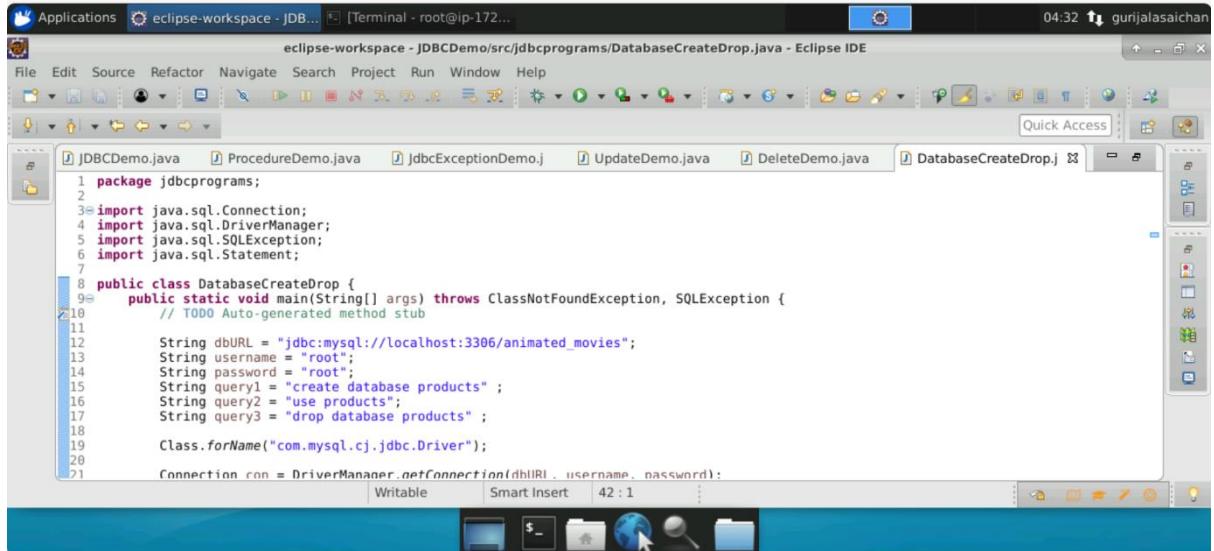
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
mysql> create procedure myprocedure(IN title varchar(50),IN genre varchar(50), IN director varchar(50),IN year int)
->
-> Begin
->
-> insert into movies values(title,genre,director,year);
->
-> select * from movies;
->
-> End $$

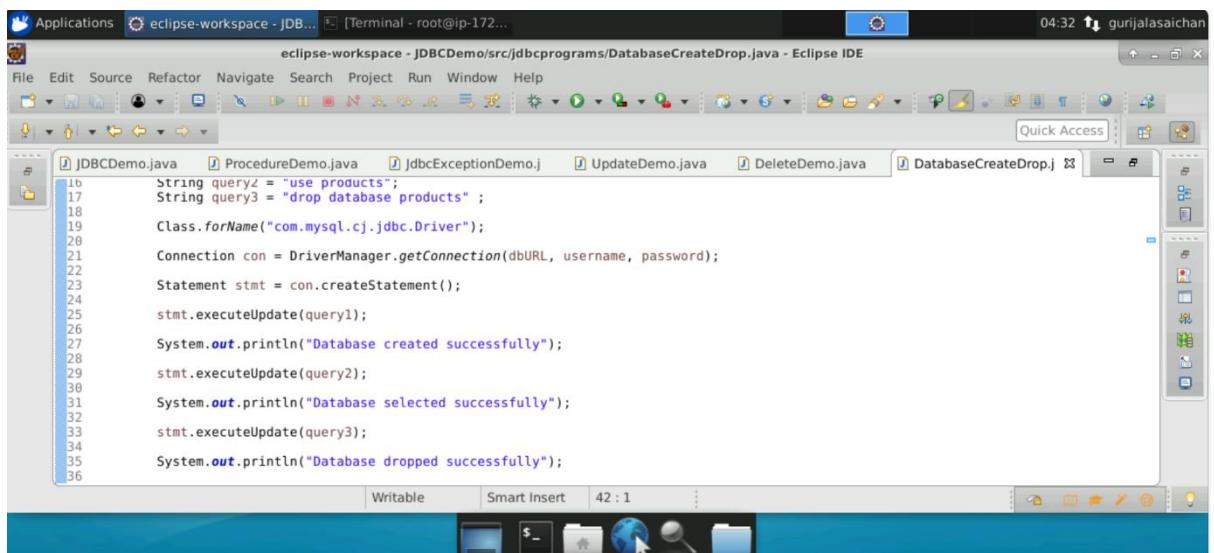
Query OK, 0 rows affected (0.02 sec)

mysql> delimiter ;
mysql>
```

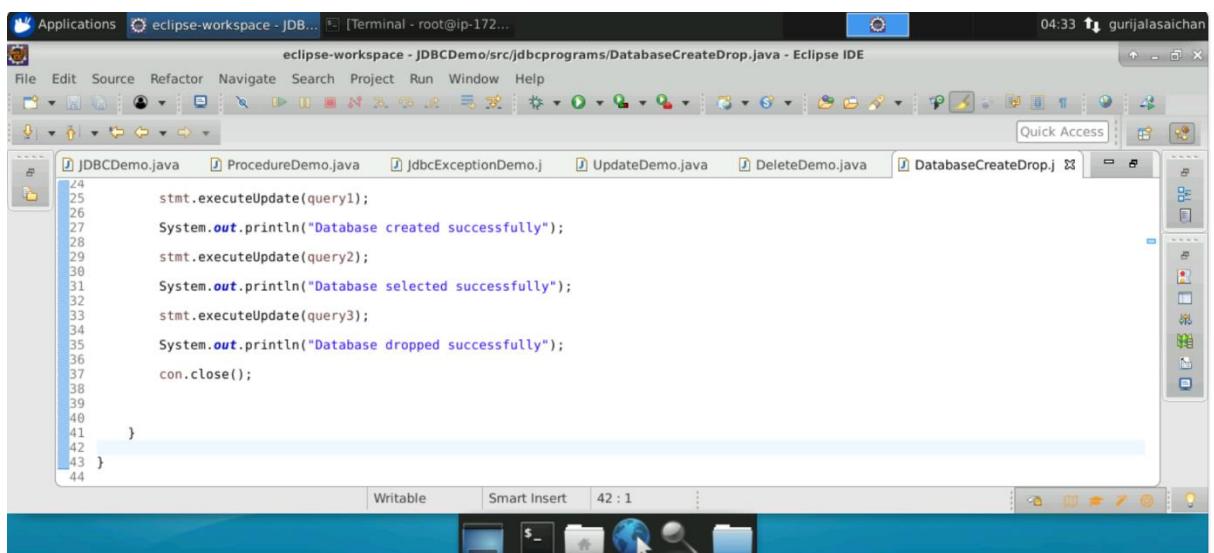
#### 4. Demonstrate how to create, select, and drop a database in JDBC



```
1 package jdbcprograms;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6 import java.sql.Statement;
7
8 public class DatabaseCreateDrop {
9     public static void main(String[] args) throws ClassNotFoundException, SQLException {
10         // TODO Auto-generated method stub
11
12         String dbURL = "jdbc:mysql://localhost:3306/animated_movies";
13         String username = "root";
14         String password = "root";
15         String query1 = "create database products" ;
16         String query2 = "use products";
17         String query3 = "drop database products" ;
18
19         Class.forName("com.mysql.cj.jdbc.Driver");
20
21         Connection con = DriverManager.getConnection(dbURL, username, password);
22
23         Statement stmt = con.createStatement();
24
25         stmt.executeUpdate(query1);
26
27         System.out.println("Database created successfully");
28
29         stmt.executeUpdate(query2);
30
31         System.out.println("Database selected successfully");
32
33         stmt.executeUpdate(query3);
34
35         System.out.println("Database dropped successfully");
36
37     }
38
39
40 }
41
42
43 }
```



```
16         String query2 = "use products";
17         String query3 = "drop database products" ;
18
19         Class.forName("com.mysql.cj.jdbc.Driver");
20
21         Connection con = DriverManager.getConnection(dbURL, username, password);
22
23         Statement stmt = con.createStatement();
24
25         stmt.executeUpdate(query1);
26
27         System.out.println("Database created successfully");
28
29         stmt.executeUpdate(query2);
30
31         System.out.println("Database selected successfully");
32
33         stmt.executeUpdate(query3);
34
35         System.out.println("Database dropped successfully");
36
37     }
38
39
40 }
41
42
43 }
```



```
24         stmt.executeUpdate(query1);
25
26         System.out.println("Database created successfully");
27
28         stmt.executeUpdate(query2);
29
30         System.out.println("Database selected successfully");
31
32         stmt.executeUpdate(query3);
33
34         System.out.println("Database dropped successfully");
35
36         con.close();
37
38
39
40 }
41
42
43 }
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC... [Terminal - root@ip-172...]" and the status bar "04:33 gurjalasaichan". The central workspace displays the code for DatabaseCreateDrop.java:

```
24     stmt.executeUpdate(query1);
25     System.out.println("Database created successfully");
26     stmt.executeUpdate(query2);
27     System.out.println("Database selected successfully");
28     stmt.executeUpdate(query3);
29     System.out.println("Database dropped successfully");
30     con.close();
31
32 }
33
34 }
```

The right-hand side of the interface shows the terminal output window with the following text:

```
<terminated> DatabaseCreateDrop [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java -Dfile.encoding=UTF-8 -classpath /home/gurjalasaichan/eclipse-workspace/JDBCPrograms/bin:/home/gurjalasaichan/eclipse-workspace/JDBCPrograms/lib/* Database created successfully
Database selected successfully
Database dropped successfully
```

## 5. Demonstrate database record handling using JDBC

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC... [Terminal - root@ip-172...]" and the status bar "04:23 gurjalasaichan". The central workspace displays the code for UpdateDemo.java:

```
1 package jdbcprograms;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8
9 public class UpdateDemo {
10     public static void main(String[] args) throws ClassNotFoundException, SQLException {
11         // TODO Auto-generated method stub
12
13         String dbURL = "jdbc:mysql://localhost:3306/animated_movies";
14         String username = "root";
15         String password = "root";
16         String query = "update movies set release_year='2023' where title='movie1'";
17         String query2 = "select * from movies";
18
19         Class.forName("com.mysql.cj.jdbc.Driver");
20
21     }
22 }
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC... [Terminal - root@ip-172...]" and the status bar "04:23 gurjalasaichan". The central workspace displays the completed code for UpdateDemo.java, including the executed queries:

```
16     String query = "update movies set release_year='2023' where title='movie1'";
17     String query2 = "select * from movies";
18
19     Class.forName("com.mysql.cj.jdbc.Driver");
20
21     Connection con = DriverManager.getConnection(dbURL, username, password);
22
23     Statement stmt = con.createStatement();
24     stmt.executeUpdate(query);
25
26     ResultSet rs = stmt.executeQuery(query2);
27
28     while(rs.next()) {
29
30         System.out.print("Title: " + rs.getString("title") + "\t");
31         System.out.print("Genre: " + rs.getString("genre") + "\t");
32         System.out.print("Director: " + rs.getString("director") + "\t");
33         System.out.println("release_year: " + rs.getString("release_year"));
34
35     }
36 }
```

Applications eclipse-workspace - JDB... [Terminal - root@ip-172... 04:23 gurjalaichan

eclipse-workspace - JDBCPrograms/src/jdbcprograms/UpdateDemo.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access

JDBCPrograms.java ProcedureDemo.java JdbcExceptionDemo.java UpdateDemo.java

```
22
23 Statement stmt = con.createStatement();
24 stmt.executeUpdate(query);
25
26 ResultSet rs = stmt.executeQuery(query2);
27
28 while(rs.next()) {
29
30     System.out.print("Title: " + rs.getString("title") + "\t");
31     System.out.print("Genre: " + rs.getString("genre") + "\t");
32     System.out.print("Director: " + rs.getString("director") + "\t");
33     System.out.println("release_year: " + rs.getString("release_year"));
34 }
35
36 con.close();
37
38 }
39
40 }
41 }
```

Writable Smart Insert 16 : 82

Applications eclipse-workspace - JDB... [Terminal - root@ip-172... 04:23 gurjalaichan

eclipse-workspace - JDBCPrograms/src/jdbcprograms/UpdateDemo.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access

Markers Properties Servers Data Source Explorer Snippets Console

```
<terminated> UpdateDemo [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Jan 3, 2024, 4:22:50 AM)
Title: Inside Out    Genre: Comedy   Director: Pete Doctor   release_year: 2015
Title: Toy Story 4   Genre: Comedy   Director: Josh Cooley   release_year: 2019
Title: movie1        Genre: romance  Director: rajamouli   release_year: 2023
```

Applications eclipse-workspace - JDB... [Terminal - root@ip-172... 04:26 gurjalaichan

eclipse-workspace - JDBCPrograms/src/jdbcprograms/DeleteDemo.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access

JDBCPrograms.java ProcedureDemo.java JdbcExceptionDemo.java UpdateDemo.java DeleteDemo.java

```
4 import java.sql.DriverManager;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.sql.Statement;
8
9 public class DeleteDemo {
10     public static void main(String[] args) throws ClassNotFoundException, SQLException {
11         // TODO Auto-generated method stub
12
13         String dbURL = "jdbc:mysql://localhost:3306/animated_movies";
14         String username = "root";
15         String password = "root";
16         String query1 = "delete from movies where title='movie1'";
17         String query2 = "select * from movies";
18
19         Class.forName("com.mysql.cj.jdbc.Driver");
20
21         Connection con = DriverManager.getConnection(dbURL, username, password);
22
23         Statement stmt = con.createStatement();
24 }
```

Writable Smart Insert 16 : 64

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC...". The main editor window displays the Java code for "DeleteDemo.java". The code uses JDBC to execute a query and print results to the console. The code is as follows:

```
26     Statement stmt = conn.createStatement();
27     String query1 = "SELECT * FROM movies WHERE title='Inside Out'";
28     ResultSet rs = stmt.executeQuery(query1);
29
30     while(rs.next()) {
31         System.out.print("Title: " + rs.getString("title") + "\t");
32         System.out.print("Genre: " + rs.getString("genre") + "\t");
33         System.out.print("Director: " + rs.getString("director") + "\t");
34         System.out.println("release_year: " + rs.getString("release_year"));
35     }
36
37     con.close();
38
39 }
40
41
42 }
43
44
45 }
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC...". The main editor window displays the Java code for "DeleteDemo.java". The "Console" tab is selected, showing the output of the program. The output shows two movie records being printed to the console.

```
[terminated> DeleteDemo [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Jan 3, 2024, 4:26:27 AM)
Title: Inside Out    Genre: Comedy   Director: Pete Doctor   release_year: 2015
Title: Toy Story 4   Genre: Comedy   Director: Josh Cooley   release_year: 2019
```

## 6. Demonstrate Transaction Management in JDBC

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBC...". The main editor window displays the Java code for "TransactionDemo.java". The code demonstrates transaction management by updating a database and then committing or rolling back the transaction. The code is as follows:

```
8 public class TransactionDemo {
9 public static void main(String[] args) throws ClassNotFoundException, SQLException {
10
11     String dbURL = "jdbc:mysql://localhost:3306/animated_movies";
12     String username = "root";
13     String password = "root";
14     String query1 = "update movies set release_year='2003' where title='Inside Out'";
15     String query2 = "update movies set release_year='2001' where title='Toy Story 4'";
16     // give a query that gives output as 0
17     String query3 = "update movies set release_year='2015' where title = 'movie1'";
18     boolean flag = false;
19
20     Class.forName("com.mysql.cj.jdbc.Driver");
21
22     Connection con = DriverManager.getConnection(dbURL, username, password);
23
24     con.setAutoCommit(false);
25     Statement stmt = con.createStatement();
26     stmt.addBatch(query1);
27     stmt.addBatch(query2);
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBCPrograms/TransactionDemo.java - Eclipse IDE". The code editor displays the following Java code:

```
26     stmt.addBatch(query1);
27     stmt.addBatch(query2);
28     stmt.addBatch(query3);
29
30     int [] result = stmt.executeBatch();
31
32     // create a loop to iterate over each integer store in the array result
33
34     for(int i=0; i<result.length;i++)
35     {
36         System.out.println(result[i]);
37         if(result[i] == 0) // if any index is storing the value ==0
38         {
39             flag = true;
40             break;
41         }
42     }
43
44
45     if(flag == false) {
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBCPrograms/TransactionDemo.java - Eclipse IDE". The code editor displays the following Java code:

```
42
43
44     if(flag == false) {
45
46         con.commit();
47         System.out.println("transaction Complete");
48     }
49     else {
50         // if flag == true
51         con.rollback();
52         System.out.println("transaction Failure");
53     }
54
55
56
57
58 }
59
60
61 }
```

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - JDBCPrograms/TransactionDemo.java - Eclipse IDE". The perspective is set to "Console". The console output window displays the following text:

```
<terminated> TransactionDemo [Java Application] /usr/lib/jvm/java-8-openjdk-amd64/bin/java (Jan 3, 2024, 10:52:25 AM)
1
0
transaction Failure
```