

Write a code for connecting JDBC to MySql and execute to display the table?

SQL CODE:

```
CREATE DATABASE ATM;
```

```
USE ATM;
```

```
CREATE TABLE ACCOUNT(
```

```
ID INT PRIMARY KEY,
```

```
ATM_PIN INT,
```

```
BANK_NAME VARCHAR(30)
```

```
);
```

```
INSERT INTO ACCOUNT VALUES(101,1788,"INDIAN BANK");
```

```
INSERT INTO ACCOUNT VALUES(102,1695,"ICICI BANK");
```

```
INSERT INTO ACCOUNT VALUES(103,8731,"KVB BANK");
```

```
SELECT * FROM ACCOUNT;
```

OUTPUT

The screenshot displays the MySQL Workbench interface. The SQL editor contains the following code:

```
1 CREATE DATABASE ATM;
2 USE ATM;
3 CREATE TABLE ACCOUNT(
4 ID INT PRIMARY KEY,
5 ATM_PIN INT,
6 BANK_NAME VARCHAR(30)
7 );
8 INSERT INTO ACCOUNT VALUES(101,1788,"INDIAN BANK");
9 INSERT INTO ACCOUNT VALUES(102,1695,"ICICI BANK");
10 INSERT INTO ACCOUNT VALUES(103,8731,"KVB BANK");
11 SELECT * FROM ACCOUNT;
```

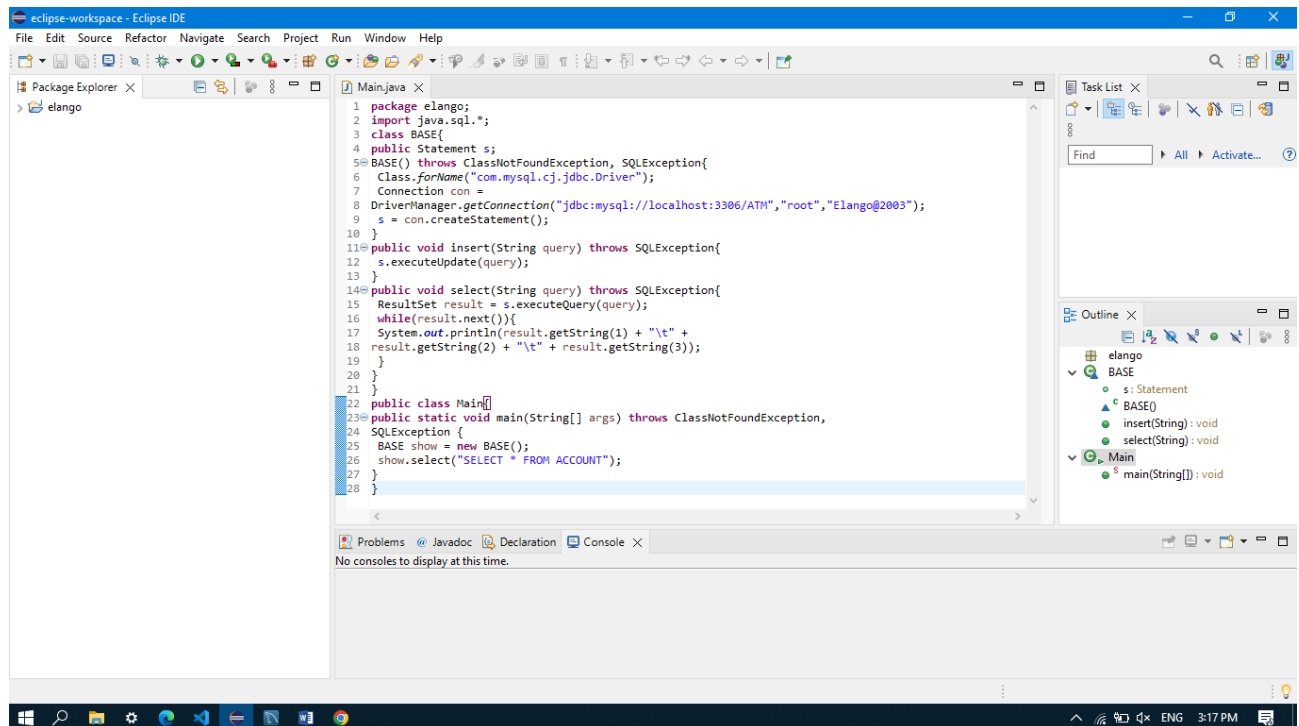
The Results window shows the output of the SELECT statement:

ID	ATM_PIN	BANK_NAME
101	1788	INDIAN BANK
102	1695	ICICI BANK
103	8731	KVB BANK

The Output window shows the execution log:

#	Time	Action	Message	Duration / Fetch
43	14:29:56	INSERT INTO ACCOUNT VALUES(101,1788,"INDIAN BANK")	1 row(s) affected	0.157 sec
44	14:30:00	INSERT INTO ACCOUNT VALUES(102,1695,"ICICI BANK")	1 row(s) affected	0.219 sec
45	14:30:04	INSERT INTO ACCOUNT VALUES(103,8731,"KVB BANK")	1 row(s) affected	0.109 sec
46	14:30:07	SELECT * FROM ACCOUNT LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec

CONNECTING JDBC:

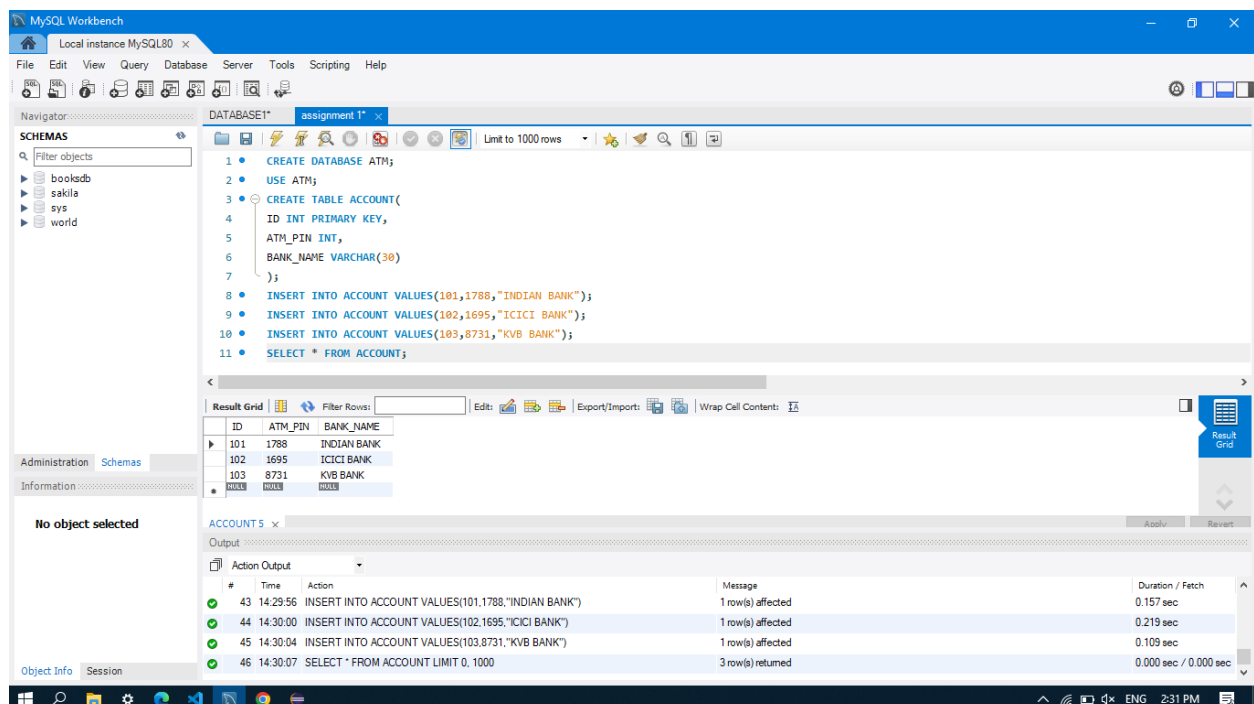


The screenshot shows the Eclipse IDE with a project named 'elango'. The main editor displays the following Java code in 'Main.java':

```
1 package elango;
2 import java.sql.*;
3 class BASE{
4     public Statement s;
5     BASE() throws ClassNotFoundException, SQLException{
6         Class.forName("com.mysql.cj.jdbc.Driver");
7         Connection con =
8             DriverManager.getConnection("jdbc:mysql://localhost:3306/ATM", "root", "Elango2003");
9         s = con.createStatement();
10    }
11    public void insert(String query) throws SQLException{
12        s.executeUpdate(query);
13    }
14    public void select(String query) throws SQLException{
15        ResultSet result = s.executeQuery(query);
16        while(result.next()){
17            System.out.println(result.getString(1) + "\t" +
18                result.getString(2) + "\t" + result.getString(3));
19        }
20    }
21 }
22 public class Main{
23     public static void main(String[] args) throws ClassNotFoundException,
24         SQLException {
25         BASE show = new BASE();
26         show.select("SELECT * FROM ACCOUNT");
27     }
28 }
```

The Package Explorer on the left shows the 'elango' project with a 'BASE' class and a 'Main' class. The Outline view on the right shows the structure of the 'Main' class, including the 'main' method.

OUTPUT



The screenshot shows the MySQL Workbench interface. The 'SQL Editor' tab contains the following SQL queries:

```
1 CREATE DATABASE ATM;
2 USE ATM;
3 CREATE TABLE ACCOUNT(
4     ID INT PRIMARY KEY,
5     ATM_PIN INT,
6     BANK_NAME VARCHAR(30)
7 );
8 INSERT INTO ACCOUNT VALUES(101,1788,"INDIAN BANK");
9 INSERT INTO ACCOUNT VALUES(102,1695,"ICICI BANK");
10 INSERT INTO ACCOUNT VALUES(103,8731,"KVB BANK");
11 SELECT * FROM ACCOUNT;
```

The 'Result Grid' shows the output of the SELECT query:

ID	ATM_PIN	BANK_NAME
101	1788	INDIAN BANK
102	1695	ICICI BANK
103	8731	KVB BANK

The 'Output' tab shows the execution results of the queries:

#	Time	Action	Message	Duration / Fetch
43	14:29:56	INSERT INTO ACCOUNT VALUES(101,1788,"INDIAN BANK")	1 row(s) affected	0.157 sec
44	14:30:00	INSERT INTO ACCOUNT VALUES(102,1695,"ICICI BANK")	1 row(s) affected	0.219 sec
45	14:30:04	INSERT INTO ACCOUNT VALUES(103,8731,"KVB BANK")	1 row(s) affected	0.109 sec
46	14:30:07	SELECT * FROM ACCOUNT LIMIT 0. 1000	3 row(s) returned	0.000 sec / 0.000 sec