Lab 4: JDBC and Swing

1. Write a Java program to illustrate the concept of Scrollable ResultSet.

```
public class Question1 {
public Question1() {
try {
Connection conn = DriverManager.getConnection("Jdbc:mysql://localhost/bca", "root","");
String query = "Select * from student";
PreparedStatement pst = conn.prepareStatement(query, ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);
ResultSet rs = pst.executeQuery();
Scanner sc = new Scanner(System.in);
int row = -1;
while (row != 0) {
System.out.println("Enter a row to read:");
row = sc.nextInt();
if (rs.absolute(row)) {
System.out.println("Name:" + rs.getString("name") + " phone number:" + rs.getString("phone
number"));
} else {
System.out.println("There is no data at row " + row);
}
}
} catch (Exception e) {
System.out.println("Error" + e.getMessage());
}
}
public static void main(String[] args) {
try {
Connection conn = DriverManager.getConnection("Jdbc:mysql://localhost/bca", "root",
"");
```

```
String query = "Select * from student";
PreparedStatement pst = conn.prepareStatement(query,
ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);
ResultSet rs = pst.executeQuery();
System.out.println("First Data is:");
rs.first();
System.out.println("Name:" + rs.getString("name") + " phone number:" +
rs.getString("phone number"));
System.out.println("Relative data is:");
rs.relative(3);
System.out.println("Name:" + rs.getString("name") + " phone number:" + rs.getString("phone
number"));
System.out.println("Previous Data is:");
rs.previous();
System.out.println("Name:" + rs.getString("name") + " phone number:" + rs.getString("phone
number"));
System.out.println("Last Data is:");
rs.last();
System.out.println("Name:" + rs.getString("name") + " phone number:" + rs.getString("phone
number"));
System.out.println("Relative Data is:");
rs.relative(-2);
System.out.println("Name:" + rs.getString("name") + " phone number:" + rs.getString("phone
number"));
System.out.println("Absolute Data is:");
rs.absolute(-1);
System.out.println("Name:" + rs.getString("name") + " phone number:" +
rs.getString("phone number"));
} catch (Exception e) {
System.out.println("Error" + e.getMessage());
```

```
}
new Question1();
}
}
```

2. Write a Java program to illustrate the concept of Updatable ResultSet.

```
import java.sql.*;
import java.util.*;
public class Question2 {
public static void main(String[] args) {
try {
Connection conn = DriverManager.getConnection("Jdbc:mysql://localhost/bca", "root",
"");
String query = "Select * from student";
PreparedStatement pst = conn.prepareStatement(query,
ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_UPDATABLE);
ResultSet rs = pst.executeQuery();
Scanner sc = new Scanner(System.in);
int row = -1;
while (row != 0) {
System.out.println("Enter a row to update:");
row = sc.nextInt();
if (rs.absolute(row)) {
rs.updateString("name", "Milan Acharya");
rs.updateRow();
System.out.println("Name:" + rs.getString("name") + " phone number:" +
rs.getString("phone number"));
} else {
System.out.println("There is no data at row " + row);
}
}
```

```
} catch (Exception e) {
System.out.println("Error" + e.getMessage());
}
}
}
```

3. Write a Java program to illustrate the concept of RowSet.

```
import java.sql.*;
import javax.sql.rowset.*;
public class Question3 {
public static void main(String[] args) {
try {
JdbcRowSet rowset = RowSetProvider.newFactory().createJdbcRowSet();
rowset.setUrl("Jdbc:mysql://localhost/bca");
rowset.setUsername("root");
rowset.setPassword("");
rowset.setCommand("Select * from student");
rowset.execute();
while (rowset.next()) {
System.out.println("Name:" + rowset.getString("name") + " phone number:" +
rowset.getString("phone number"));
}
} catch (SQLException e) {
System.out.println("error:" + e.getMessage());
}
}
}
```

4. Write a program using Java that has the JDBC connectivity and must be able to perform basic CRUD operations as shown below:

```
import java.awt.*;
import javax.swing.*;
```

```
import java.sql.*;
public class Question4 extends JFrame implements ActionListener {
JRadioButton r1, r2;
JComboBox<String> box;
JButton submit, update, select_for_update, Delete;
JTextField t1, t2, t3, t4;
ButtonGroup b;
JTable table;
DefaultTableModel model;
Connection conn;
int selectedUserId = -1;
public Question4() {
JFrame f = new JFrame("Form Sample");
f.setSize(1000, 500);
f.setLayout(null);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
try {
conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/BCA2077", "root",
"");
} catch (SQLException e) {
e.printStackTrace();
}
JLabel heading = new JLabel("Form Handling ");
heading.setBounds(400, 10, 200, 40);
f.add(heading);
JLabel l1 = new JLabel("First Name:");
l1.setBounds(20, 65, 100, 20);
f.add(l1);
t1 = new JTextField();
t1.setBounds(120, 65, 200, 20);
```

```
f.add(t1);
JLabel l2 = new JLabel("Last Name:");
l2.setBounds(20, 125, 100, 20);
f.add(l2);
t2 = new JTextField();
t2.setBounds(120, 125, 200, 20);
f.add(t2);
JLabel l3 = new JLabel("Choose Gender:");
l3.setBounds(20, 195, 100, 20);
f.add(l3);
r1 = new JRadioButton("Male");
r1.setBounds(120, 195, 80, 20);
r1.addActionListener(this);
r2 = new JRadioButton("Female");
r2.setBounds(200, 195, 80, 20);
r2.addActionListener(this);
b = new ButtonGroup();
b.add(r1);
b.add(r2);
f.add(r1);
f.add(r2);
JLabel l4 = new JLabel("Address:");
l4.setBounds(20, 225, 100, 20);
f.add(l4);
t3 = new JTextField();
t3.setBounds(120, 225, 200, 20);
f.add(t3);
JLabel l5 = new JLabel("Email:");
l5.setBounds(20, 275, 100, 20);
f.add(l5);
```

```
t4 = new JTextField();
t4.setBounds(120, 275, 200, 20);
f.add(t4);
JLabel l6 = new JLabel("Choose Blood Group:");
l6.setBounds(20, 350, 150, 20);
f.add(l6);
String[] blood = {"Select one", "A+", "B+", "O+", "AB+", "O-", "AB-", "A-", "B-"};
box = new JComboBox<>(blood);
box.setBounds(170, 350, 150, 20);
f.add(box);
submit = new JButton("Submit");
submit.setBounds(150, 420, 80, 20);
submit.addActionListener(this);
f.add(submit);
update = new JButton("Update");
update.setBounds(400, 420, 80, 20);
update.addActionListener(this);
f.add(update);
select_for_update = new JButton("Select For Update");
select_for_update.setBounds(600, 420, 120, 20);
select_for_update.addActionListener(this);
f.add(select_for_update);
Delete = new JButton("Delete");
Delete.setBounds(800, 420, 80, 20);
Delete.addActionListener(this);
f.add(Delete);
model = new DefaultTableModel();
model.setColumnIdentifiers(new Object[]{"ID", "First Name", "Last Name", "Gender",
"Address", "Email", "Blood Group"});
table = new JTable(model);
```

```
JScrollPane pane = new JScrollPane(table);
pane.setBounds(350, 50, 630, 350);
f.add(pane);
loadTableData();
f.setVisible(true);
}
public static void main(String[] args) {
new Question4();
}
@Override
public void actionPerformed(ActionEvent e) {
if (e.getSource() == submit) {
try {
String gender = r1.isSelected() ? "Male" : "Female";
String iquery = "INSERT INTO users(first_name, last_name, gender, address, email,
blood_group) VALUES (?, ?, ?, ?, ?, ?)";
PreparedStatement pst = conn.prepareStatement(iquery);
pst.setString(1, t1.getText());
pst.setString(2, t2.getText());
pst.setString(3, gender);
pst.setString(4, t3.getText());
pst.setString(5, t4.getText());
pst.setString(6, (String) box.getSelectedItem());
pst.executeUpdate();
loadTableData();
clearForm();
} catch (SQLException ex) {
ex.printStackTrace();
}
} else if (e.getSource() == update) {
```

```
if (selectedUserId != -1) {
try {
String gender = r1.isSelected() ? "Male" : "Female";
String uquery = "UPDATE users SET first_name = ?, last_name = ?, gender = ?, address
= ?, email = ?, blood_group = ? WHERE id = ?";
PreparedStatement pst = conn.prepareStatement(uquery);
pst.setString(1, t1.getText());
pst.setString(2, t2.getText());
pst.setString(3, gender);
pst.setString(4, t3.getText());
pst.setString(5, t4.getText());
pst.setString(6, (String) box.getSelectedItem());
pst.setInt(7, selectedUserId);
pst.executeUpdate();
loadTableData();
clearForm();
selectedUserId = -1;
} catch (SQLException ex) {
ex.printStackTrace();
}
} else {
JOptionPane.showMessageDialog(null, "Please select a user to update");
}
} else if (e.getSource() == select_for_update) {
int row = table.getSelectedRow();
if (row >= 0) {
selectedUserId = (int) model.getValueAt(row, 0);
t1.setText(model.getValueAt(row, 1).toString());
t2.setText(model.getValueAt(row, 2).toString());
String gender = model.getValueAt(row, 3).toString();
```

```
if (gender.equals("Male")) {
r1.setSelected(true);
} else {
r2.setSelected(true);
}
t3.setText(model.getValueAt(row, 4).toString());
t4.setText(model.getValueAt(row, 5).toString());
box.setSelectedItem(model.getValueAt(row, 6).toString());
} else {
JOptionPane.showMessageDialog(null, "Please select a user from the table");
}
} else if (e.getSource() == Delete) {
int row = table.getSelectedRow();
if (row >= 0) {
int userId = (int) model.getValueAt(row, 0);
try {
String dquery = "DELETE FROM users WHERE id = ?";
PreparedStatement pst = conn.prepareStatement(dquery);
pst.setInt(1, userId);
pst.executeUpdate();
loadTableData();
} catch (SQLException ex) {
ex.printStackTrace();
}
} else {
JOptionPane.showMessageDialog(null, "Please select a user to delete");
}
}
}
private void loadTableData() {
```

```
try {
model.setRowCount(0); // Clear previous data
String squery = "SELECT * FROM users";
PreparedStatement pst = conn.prepareStatement(squery);
ResultSet rs = pst.executeQuery();
while (rs.next()) {
model.addRow(new Object[]{
rs.getInt("id"),
rs.getString("first_name"),
rs.getString("last_name"),
rs.getString("gender"),
rs.getString("address"),
rs.getString("email"),
rs.getString("blood_group")
});
}
} catch (SQLException e) {
e.printStackTrace();
}
}
private void clearForm() {
t1.setText("");
t2.setText("");
b.clearSelection();
t3.setText("");
t4.setText("");
box.setSelectedIndex(0);
}
}
```