

```

import java.sql.*;
import javax.swing.JOptionPane;

public class CustomerUpdateDB{
    private static Connection connection;
    private static Statement scrollStatement;
    private static ResultSet customerDBTable;

    public static void connect() throws ClassNotFoundException, SQLException{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        String url = "jdbc:odbc:DentonMunicipal";
        String user = "Admin";
        String password = "";
        connection = DriverManager.getConnection(url, user, password);
    }

    public static void open() throws SQLException{
        scrollStatement = connection.createStatement(
            ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_UPDATABLE);
        String query = "SELECT CustomerNumber, LastName, FirstName, MiddleName, "
            + "Address, City, State, Zipcode, CustomerSinceMo, CustomerSinceDay, "
            + "CustomerSinceYear, CustomerType, ServiceType, LastReadingMo,
LastReadingDay, "
            + "LastReadingYear, LastReading, CurrentReadingMo,
CurrentReadingDay, "
            + "CurrentReadingYear, CurrentReading, OutstandingBalance,
CurrentBalance, "
            + "LastPaymentMo, LastPaymentDay, LastPaymentYear "
            + "FROM customer ORDER BY CustomerNumber ASC";
        customerDBTable = scrollStatement.executeQuery(query);
    }

    public static void close(){
        try{
            customerDBTable.close();
            scrollStatement.close();
        }
        catch(SQLException sqle){
            System.err.println(sqle.getMessage());
        }
    }

    public static CustomerUpdate moveFirst() throws SQLException{
        customerDBTable.first();
        CustomerUpdate firstCustomer = new CustomerUpdate(
            customerDBTable.getString("CustomerNumber"),
            customerDBTable.getString("LastName"),
            customerDBTable.getString("FirstName"),
            customerDBTable.getString("MiddleName"),
            customerDBTable.getString("Address"),
            customerDBTable.getString("City"),
            customerDBTable.getString("State"),
            customerDBTable.getString("Zipcode"),
            customerDBTable.getInt("CustomerSinceMo"),
            customerDBTable.getInt("CustomerSinceDay"),
            customerDBTable.getInt("CustomerSinceYear"),
            customerDBTable.getString("CustomerType"),

```

```

        customerDBTable.getInt("ServiceType"),
        customerDBTable.getInt("LastReadingMo"),
        customerDBTable.getInt("LastReadingDay"),
        customerDBTable.getInt("LastReadingYear"),
        customerDBTable.getInt("LastReading"),
        customerDBTable.getInt("CurrentReadingMo"),
        customerDBTable.getInt("CurrentReadingDay"),
        customerDBTable.getInt("CurrentReadingYear"),
        customerDBTable.getInt("CurrentReading"),
        customerDBTable.getDouble("OutstandingBalance"),
        customerDBTable.getDouble("CurrentBalance"),
        customerDBTable.getInt("LastPaymentMo"),
        customerDBTable.getInt("LastPaymentDay"),
        customerDBTable.getInt("LastPaymentYear"));
    return firstCustomer;
}

public static CustomerUpdate movePrevious() throws SQLException{
    if (customerDBTable.isFirst() == false)
        customerDBTable.previous();
    else
        customerDBTable.first();
    CustomerUpdate previousCustomer = new CustomerUpdate(
        customerDBTable.getString(1), customerDBTable.getString(2),
        customerDBTable.getString(3), customerDBTable.getString(4),
        customerDBTable.getString(5), customerDBTable.getString(6),
        customerDBTable.getString(7), customerDBTable.getString(8),
        customerDBTable.getInt(9), customerDBTable.getInt(10),
        customerDBTable.getInt(11), customerDBTable.getString(12),
        customerDBTable.getInt(13), customerDBTable.getInt(14),
        customerDBTable.getInt(15), customerDBTable.getInt(16),
        customerDBTable.getInt(17), customerDBTable.getInt(18),
        customerDBTable.getInt(19), customerDBTable.getInt(20),
        customerDBTable.getInt(21), customerDBTable.getDouble(22),
        customerDBTable.getDouble(23), customerDBTable.getInt(24),
        customerDBTable.getInt(25), customerDBTable.getInt(26));
    return previousCustomer;
}

public static CustomerUpdate moveNext() throws SQLException{
    if (customerDBTable.isLast() == false)
        customerDBTable.next();
    else
        customerDBTable.last();
    CustomerUpdate nextCustomer = new CustomerUpdate(
        customerDBTable.getString(1), customerDBTable.getString(2),
        customerDBTable.getString(3), customerDBTable.getString(4),
        customerDBTable.getString(5), customerDBTable.getString(6),
        customerDBTable.getString(7), customerDBTable.getString(8),
        customerDBTable.getInt(9), customerDBTable.getInt(10),
        customerDBTable.getInt(11), customerDBTable.getString(12),
        customerDBTable.getInt(13), customerDBTable.getInt(14),
        customerDBTable.getInt(15), customerDBTable.getInt(16),
        customerDBTable.getInt(17), customerDBTable.getInt(18),
        customerDBTable.getInt(19), customerDBTable.getInt(20),
        customerDBTable.getInt(21), customerDBTable.getDouble(22),
        customerDBTable.getDouble(23), customerDBTable.getInt(24),

```

```

        customerDBTable.getInt(25), customerDBTable.getInt(26));
return nextCustomer;
}

public static CustomerUpdate moveLast() throws SQLException{
    customerDBTable.last();
    CustomerUpdate lastCustomer = new CustomerUpdate(
        customerDBTable.getString(1), customerDBTable.getString(2),
        customerDBTable.getString(3), customerDBTable.getString(4),
        customerDBTable.getString(5), customerDBTable.getString(6),
        customerDBTable.getString(7), customerDBTable.getString(8),
        customerDBTable.getInt(9), customerDBTable.getInt(10),
        customerDBTable.getInt(11), customerDBTable.getString(12),
        customerDBTable.getInt(13), customerDBTable.getInt(14),
        customerDBTable.getInt(15), customerDBTable.getInt(16),
        customerDBTable.getInt(17), customerDBTable.getInt(18),
        customerDBTable.getInt(19), customerDBTable.getInt(20),
        customerDBTable.getInt(21), customerDBTable.getDouble(22),
        customerDBTable.getDouble(23), customerDBTable.getInt(24),
        customerDBTable.getInt(25), customerDBTable.getInt(26));
    return lastCustomer;
}

public static void addRecord(CustomerUpdate customer) throws SQLException{
    String query = "INSERT INTO customer( " +
        "CustomerNumber, LastName, FirstName, MiddleName, Address,
City, State, Zipcode, " +
        "CustomerSinceMo, CustomerSinceDay, CustomerSinceYear,
CustomerType, ServiceType, " +
        "LastReadingMo, LastReadingDay, LastReadingYear, LastReading,
" +
        "CurrentReadingMo, CurrentReadingDay, CurrentReadingYear,
CurrentReading, " +
        "OutstandingBalance, CurrentBalance, LastPaymentMo,
LastPaymentDay, LastPaymentYear )" +
        "VALUES ('" + customer.getCustomerNumber() + "', " +
        "'" + customer.getLastName() + "', " +
        "'" + customer.getFirstName() + "', " +
        "'" + customer.getMiddleName() + "', " +
        "'" + customer.getAddress() + "', " +
        "'" + customer.getCity() + "', " +
        "'" + customer.getState() + "', " +
        "'" + customer.getZip() + "', " +
        "'" + customer.getCustomerSinceMo() + "', " +
        "'" + customer.getCustomerSinceDay() + "', " +
        "'" + customer.getCustomerSinceYear() + "', " +
        "'" + customer.getCustomerType() + "', " +
        "'" + customer.getServiceType() + "', " +
        "'" + customer.getLastReadingMo() + "', " +
        "'" + customer.getLastReadingDay() + "', " +
        "'" + customer.getLastReadingYear() + "', " +
        "'" + customer.getLastReading() + "', " +
        "'" + customer.getCurrentReadingMo() + "', " +
        "'" + customer.getCurrentReadingDay() + "', " +
        "'" + customer.getCurrentReadingYear() + "', " +
        "'" + customer.getCurrentReading() + "', " +
        "'" + customer.getOutstandingBalance() + "', " +

```

```

        "" + customer.getCurrentBalance() + "', " +
        "" + customer.getLastPaymentMo() + "', " +
        "" + customer.getLastPaymentDay() + "', " +
        "" + customer.getLastPaymentYear() + "')";
Statement statement = connection.createStatement();
statement.executeUpdate(query);
statement.close();
close();
open();
}

public static void updateRecord(CustomerUpdate customer) throws SQLException{
String query = "UPDATE customer SET " +
    "CustomerNumber = '" + customer.getCustomerNumber() + "', " +
    "LastName = '" + customer.getLastName() + "', " +
    "FirstName = '" + customer.getFirstName() + "', " +
    "MiddleName = '" + customer.getMiddleName() + "', " +
    "Address = '" + customer.getAddress() + "', " +
    "City = '" + customer.getCity() + "', " +
    "State = '" + customer.getState() + "', " +
    "Zipcode = '" + customer.getZip() + "', " +
    "CustomerSinceMo = '" + customer.getCustomerSinceMo() + "', "
+
    "CustomerSinceDay = '" + customer.getCustomerSinceDay() + "',
" +
    "CustomerSinceYear = '" + customer.getCustomerSinceYear() +
"', " +
    "CustomerType = '" + customer.getCustomerType() + "', " +
    "ServiceType = '" + customer.getServiceType() + "', " +
    "LastReadingMo = '" + customer.getLastReadingMo() + "', " +
    "LastReadingDay = '" + customer.getLastReadingDay() + "', " +
    "LastReadingYear = '" + customer.getLastReadingYear() + "', "
+
    "LastReading = '" + customer.getLastReading() + "', " +
    "CurrentReadingMo = '" + customer.getCurrentReadingMo() + "',
" +
    "CurrentReadingDay = '" + customer.getCurrentReadingDay() +
"', " +
    "CurrentReadingYear = '" + customer.getCurrentReadingYear() +
"', " +
    "CurrentReading = '" + customer.getCurrentReading() + "', " +
    "OutstandingBalance = '" + customer.getOutstandingBalance() +
"', " +
    "CurrentBalance = '" + customer.getCurrentBalance() + "', " +
    "LastPaymentMo = '" + customer.getLastPaymentMo() + "', " +
    "LastPaymentDay = '" + customer.getLastPaymentDay() + "', " +
    "LastPaymentYear = '" + customer.getLastPaymentYear() + "' " +
    "WHERE CustomerNumber = " + customer.getCustomerNumber();
Statement statement = connection.createStatement();
statement.executeUpdate(query);
statement.close();
}

public static void deleteRecord(String customerNumber) throws SQLException{
String query = "DELETE FROM customer " +
    "WHERE CustomerNumber = " + customerNumber;
Statement statement = connection.createStatement();
}

```

```
statement.executeUpdate(query);
statement.close();
close();
open();
}
}
```