Desktop Programming

Data Storage

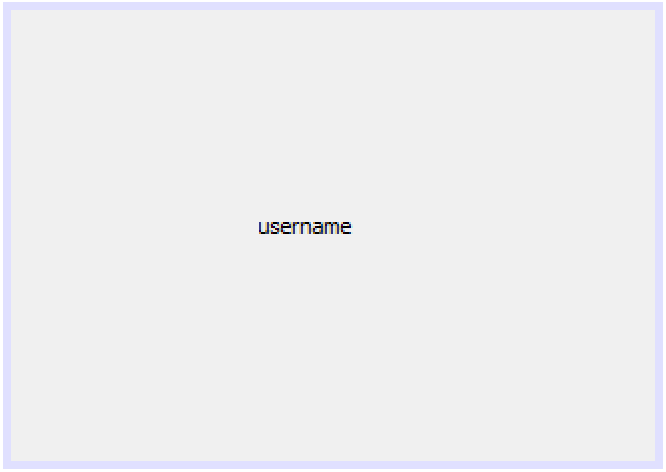
https://hermanka.github.io

Data Storage

- Configuration file (.ini,.properties)
- Standalone DB (SQLite, Access, etc.)
- DB Server (MySQL,PostgreSQL,Oracle, etc.)

Ini file

Buat class baru dengan nama “Config” dan desain antarmuka seperti dibawah ini



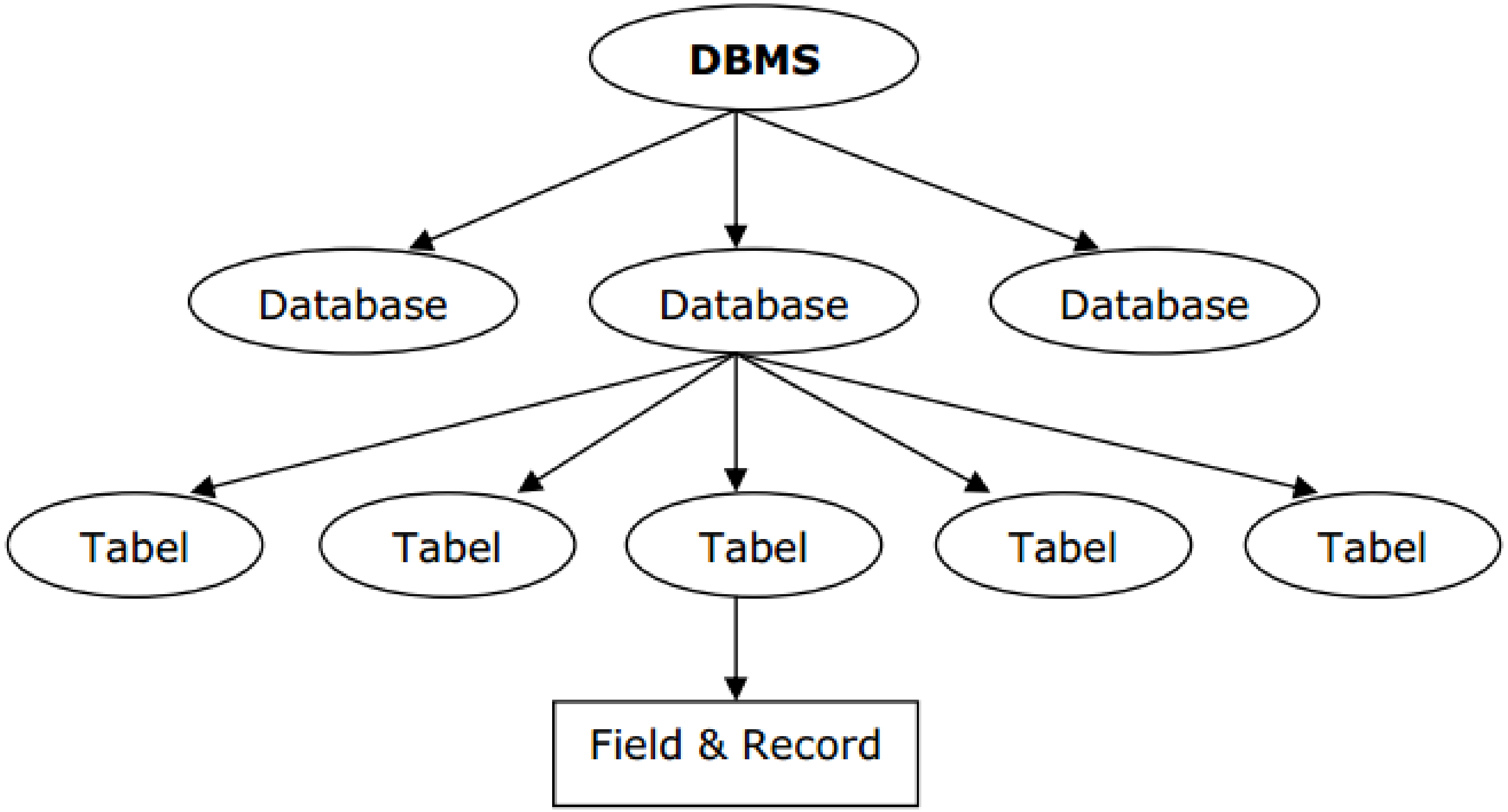
Buatlah file “conf.ini” pada root directory project

```ini
user    =   anonym
bgcolor =   red
```

conf.ini

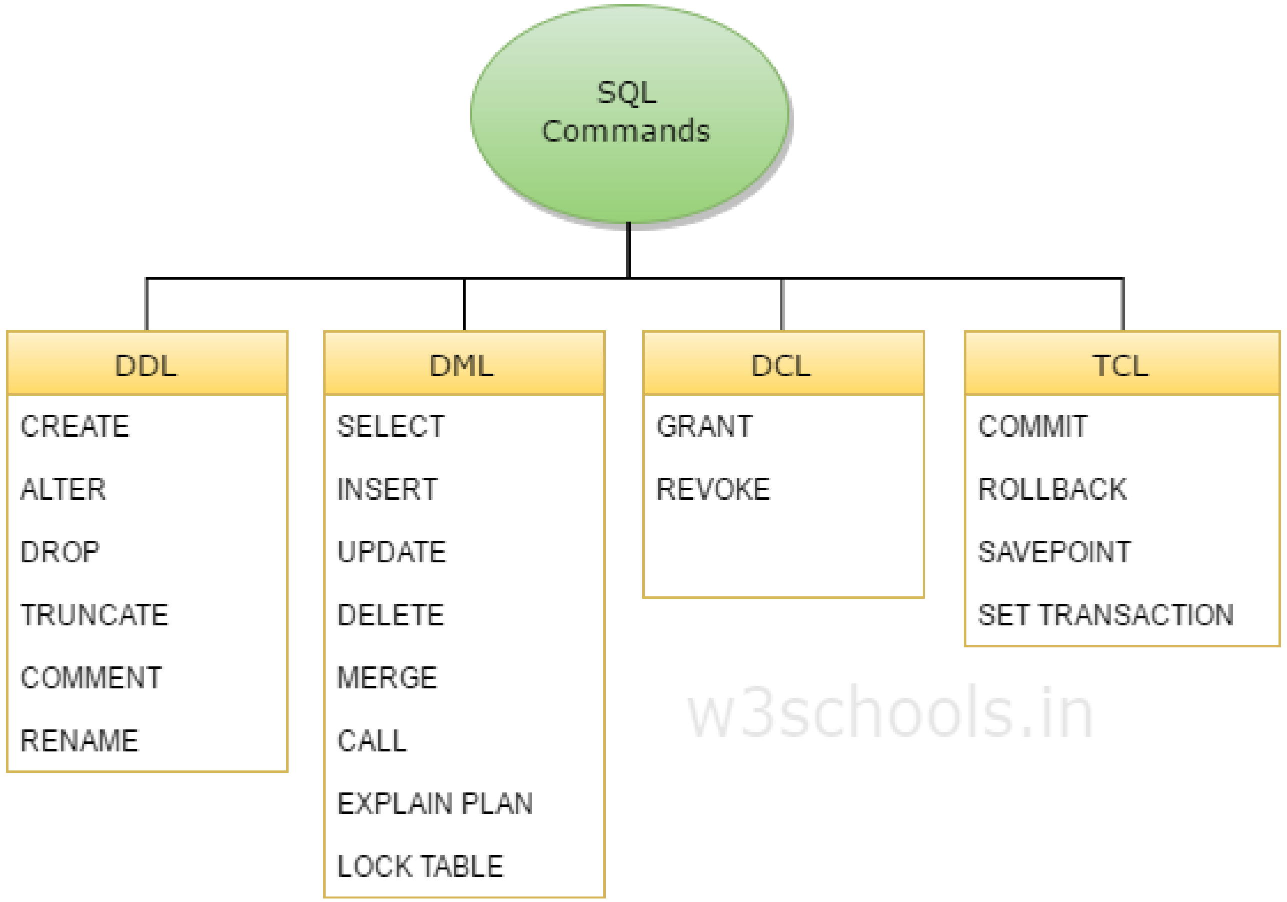
```java
    public Config() {
        initComponents();
        Properties p = new Properties();
        try {
            p.load(new FileInputStream("conf.ini"));
            lblUser.setText(p.getProperty("user"));
            this.getContentPane()
                    .setBackground((Color)Color.class.getField(p.getProperty("bgcolor")).get(null));
        } catch (Exception e) {
            System.err.println(e.getMessage());
        }
    }
```

DBMS


DBMS
Database
Database
Database
Tabel
Tabel
Tabel
Tabel
Tabel
Field & Record

SQL (*Structured Query Language*)

Merupakan suatu bahasa (*language*) yang digunakan untuk mengakses database.



SQL CHEAT SHEET http://www.sqltutorial.org

MANAGING TABLES

```
CREATE TABLE t (
    id INT PRIMARY KEY,
    name VARCHAR NOT NULL,
    price INT DEFAULT 0
);
```
Create a new table with three columns

```
DROP TABLE t ;
```
Delete the table from the database

```
ALTER TABLE t ADD column;
```
Add a new column to the table

```
ALTER TABLE t DROP COLUMN c ;
```
Drop column c from the table

```
ALTER TABLE t ADD constraint;
```
Add a constraint

```
ALTER TABLE t DROP constraint;
```
Drop a constraint

```
ALTER TABLE t1 RENAME TO t2;
```
Rename a table from t1 to t2

```
ALTER TABLE t1 RENAME c1 TO c2 ;
```
Rename column c1 to c2

```
TRUNCATE TABLE t;
```
Remove all data in a table

USING SQL CONSTRAINTS

```
CREATE TABLE t(
   c1 INT, c2 INT, c3 VARCHAR,
   PRIMARY KEY (c1,c2)
);
```
Set c1 and c2 as a primary key

```
CREATE TABLE t1(
   c1 INT PRIMARY KEY,
   c2 INT,
   FOREIGN KEY (c2) REFERENCES t2(c2)
);
```
Set c2 column as a foreign key

```
CREATE TABLE t(
   c1 INT, c1 INT,
   UNIQUE(c2,c3)
);
```
Make the values in c1 and c2 unique

```
CREATE TABLE t(
  c1 INT, c2 INT,
  CHECK(c1> 0 AND c1 >= c2)
);
```
Ensure c1 > 0 and values in c1 >= c2

```
CREATE TABLE t(
    c1 INT PRIMARY KEY,
    c2 VARCHAR NOT NULL
);
```
Set values in c2 column not NULL

MODIFYING DATA

```
INSERT INTO t(column_list)
VALUES(value_list);
```
Insert one row into a table

```
INSERT INTO t(column_list)
VALUES (value_list),
          (value_list), ....;
```
Insert multiple rows into a table

```
INSERT INTO t1(column_list)
SELECT column_list
FROM t2;
```
Insert rows from t2 into t1

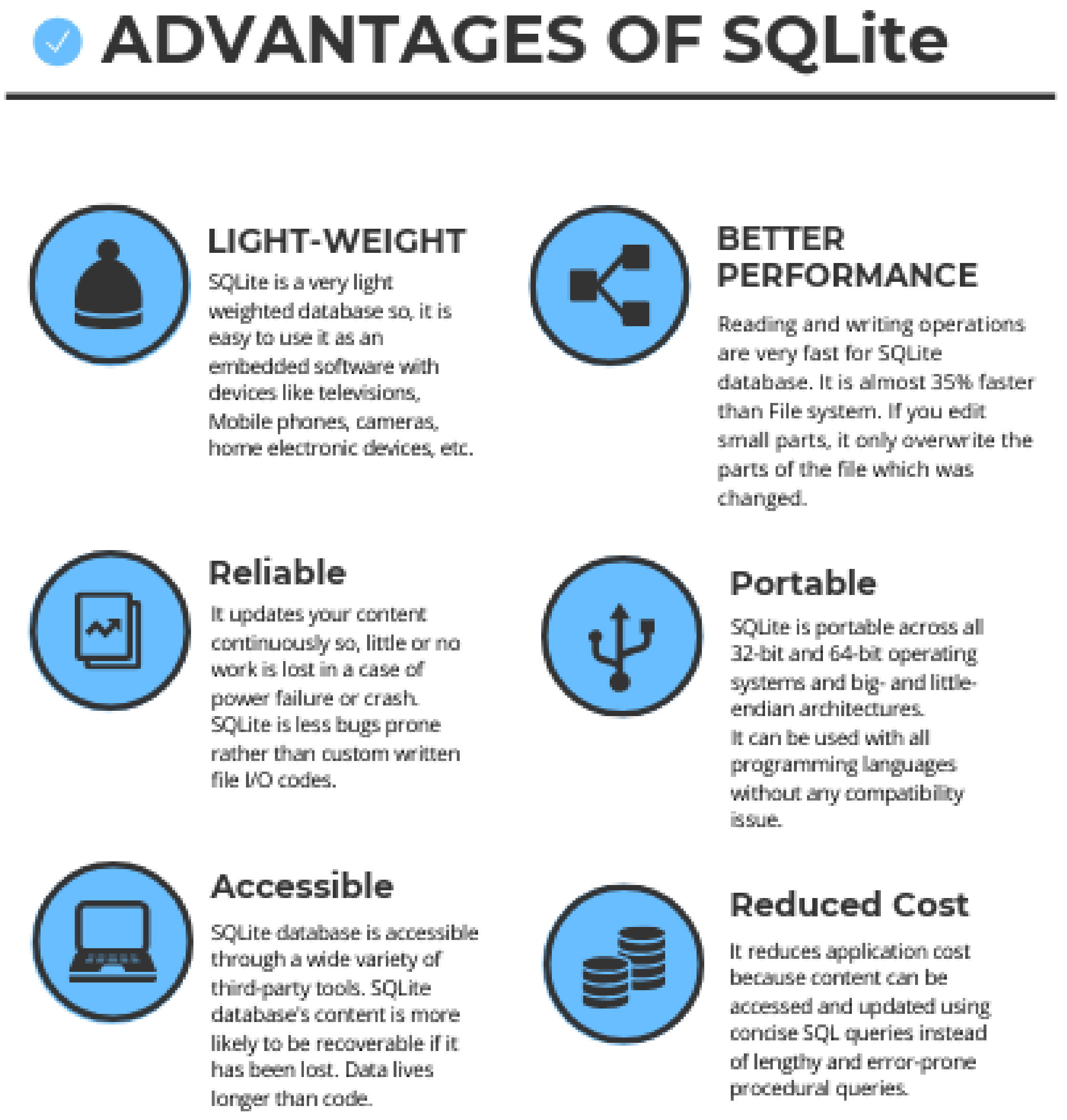
```
UPDATE t
SET c1 = new_value;
```
Update new value in the column c1 for all rows

```
UPDATE t
SET c1 = new_value,
      c2 = new_value
WHERE condition;
```
Update values in the column c1, c2 that match the condition

```
DELETE FROM t;
```
Delete all data in a table

```
DELETE FROM t
WHERE condition;
```
Delete subset of rows in a table

SQlite



ADVANTAGES OF SQLite

LIGHT-WEIGHT

SQLite is a very light weighted database so, it is easy to use it as an embedded software with devices like televisions, Mobile phones, cameras, home electronic devices, etc.

BETTER PERFORMANCE

Reading and writing operations are very fast for SQLite database. It is almost 35% faster than File system. If you edit small parts, it only overwrite the parts of the file which was changed.

Reliable

It updates your content continuously so, little or no work is lost in a case of power failure or crash. SQLite is less bugs prone rather than custom written file I/O codes.

Portable

SQLite is portable across all 32-bit and 64-bit operating systems and big- and little-endian architectures.
It can be used with all programming languages without any compatibility issue.

Accessible

SQLite database is accessible through a wide variety of third-party tools. SQLite database's content is more likely to be recoverable if it has been lost. Data lives longer than code.

Reduced Cost

It reduces application cost because content can be accessed and updated using concise SQL queries instead of lengthy and error-prone procedural queries.

javatpoint.com

SQLite Disadvantages

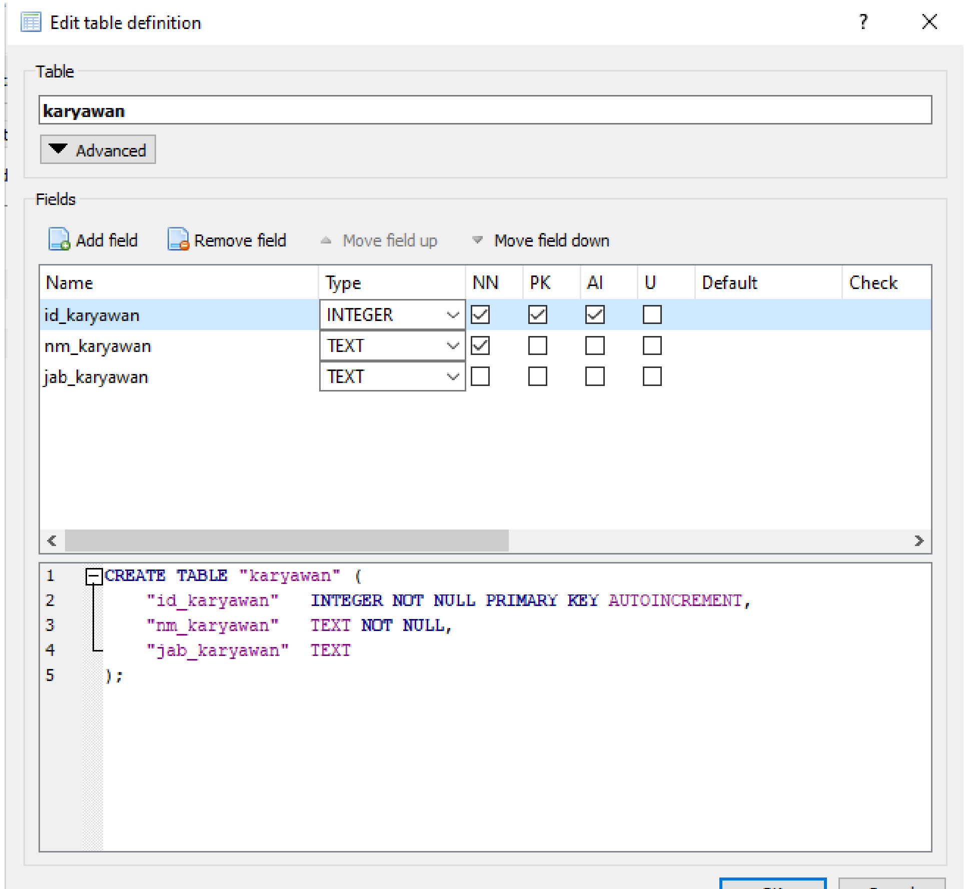
- SQLite is used to handle low to medium traffic HTTP requests.
- Database size is restricted to 2GB in most cases.

Requirements

- SQLite Manager
- SQLite JDBC
  - https://bitbucket.org/xerial/sqlite-jdbc/downloads/

SQLite

1. Buka SQL Editor Anda
2. Buat database > simpan di directory project dengan ekstensi *.db
3. Buat Tabel dengan nama “karyawan”



Edit table definition

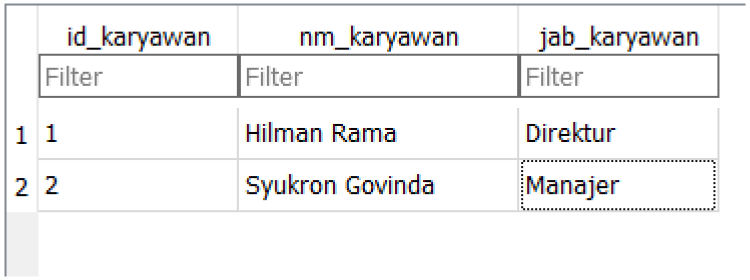
Table: karyawan

Fields

Name	Type	NN	PK	AI	U	Default	Check
id_karyawan	INTEGER	✓	✓	✓	☐		
nm_karyawan	TEXT	✓	☐	☐	☐		
jab_karyawan	TEXT	☐	☐	☐	☐		

```
CREATE TABLE "karyawan" (
    "id_karyawan"   INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
    "nm_karyawan"   TEXT NOT NULL,
    "jab_karyawan"  TEXT
);
```

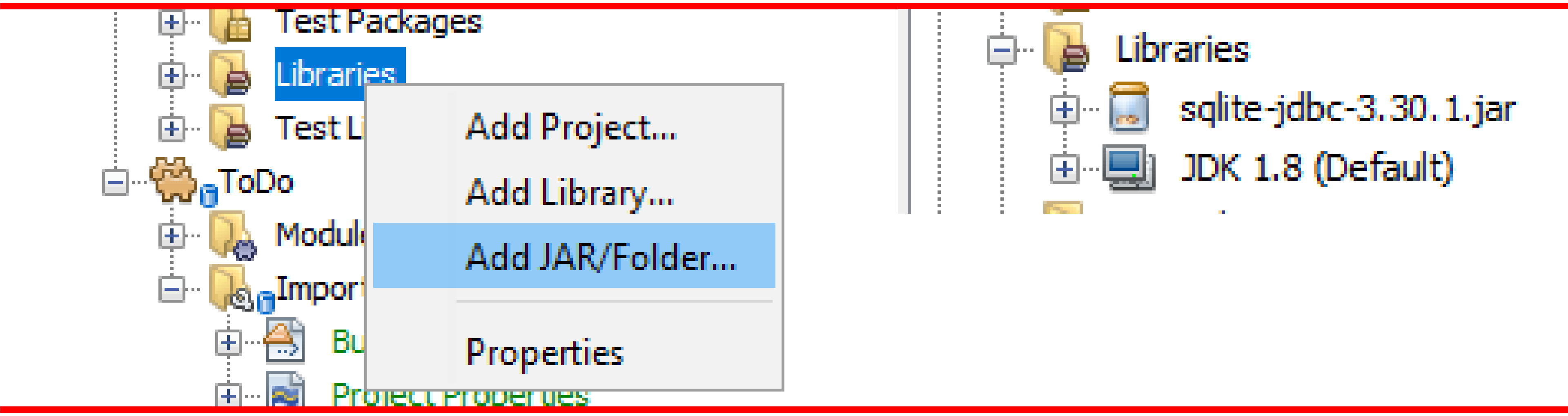
4. Tambahkan beberapa record ke table karyawan



	id_karyawan	nm_karyawan	jab_karyawan
	Filter	Filter	Filter
1	1	Hilman Rama	Direktur
2	2	Syukron Govinda	Manajer

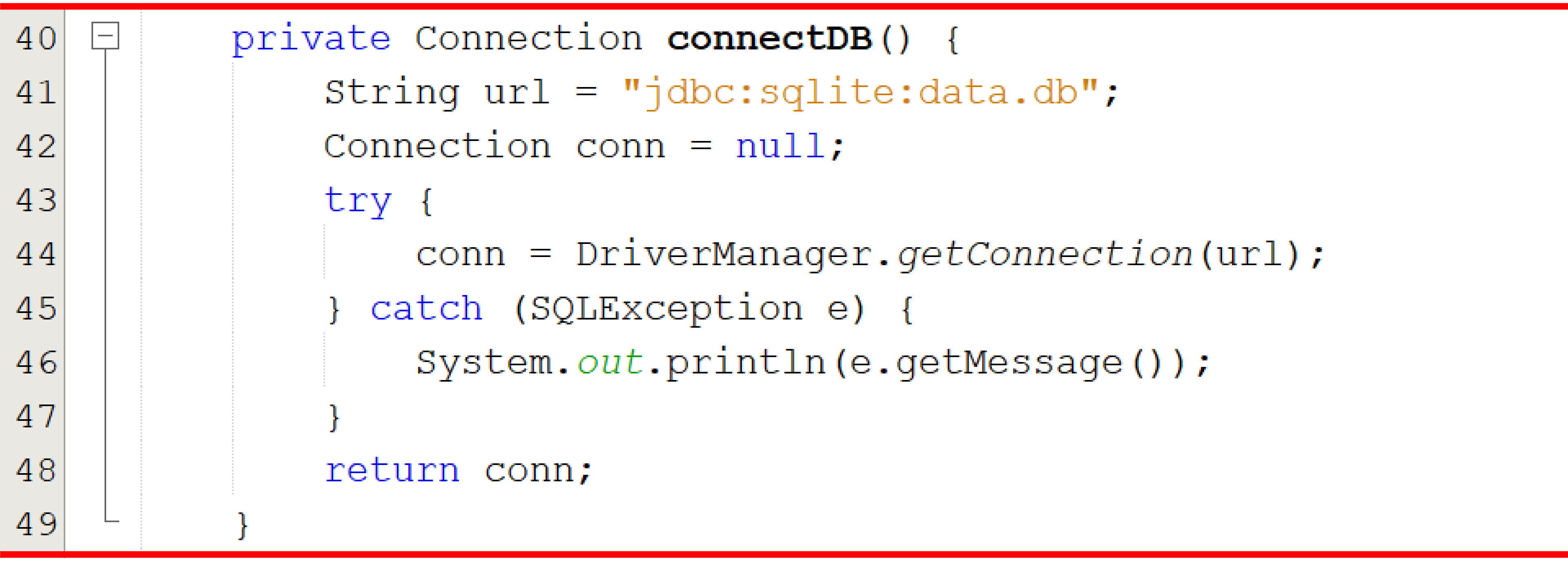
SQLite

Import library SQLite JDBC ke project (link download ada pada slide sebelumnya)



SQLite

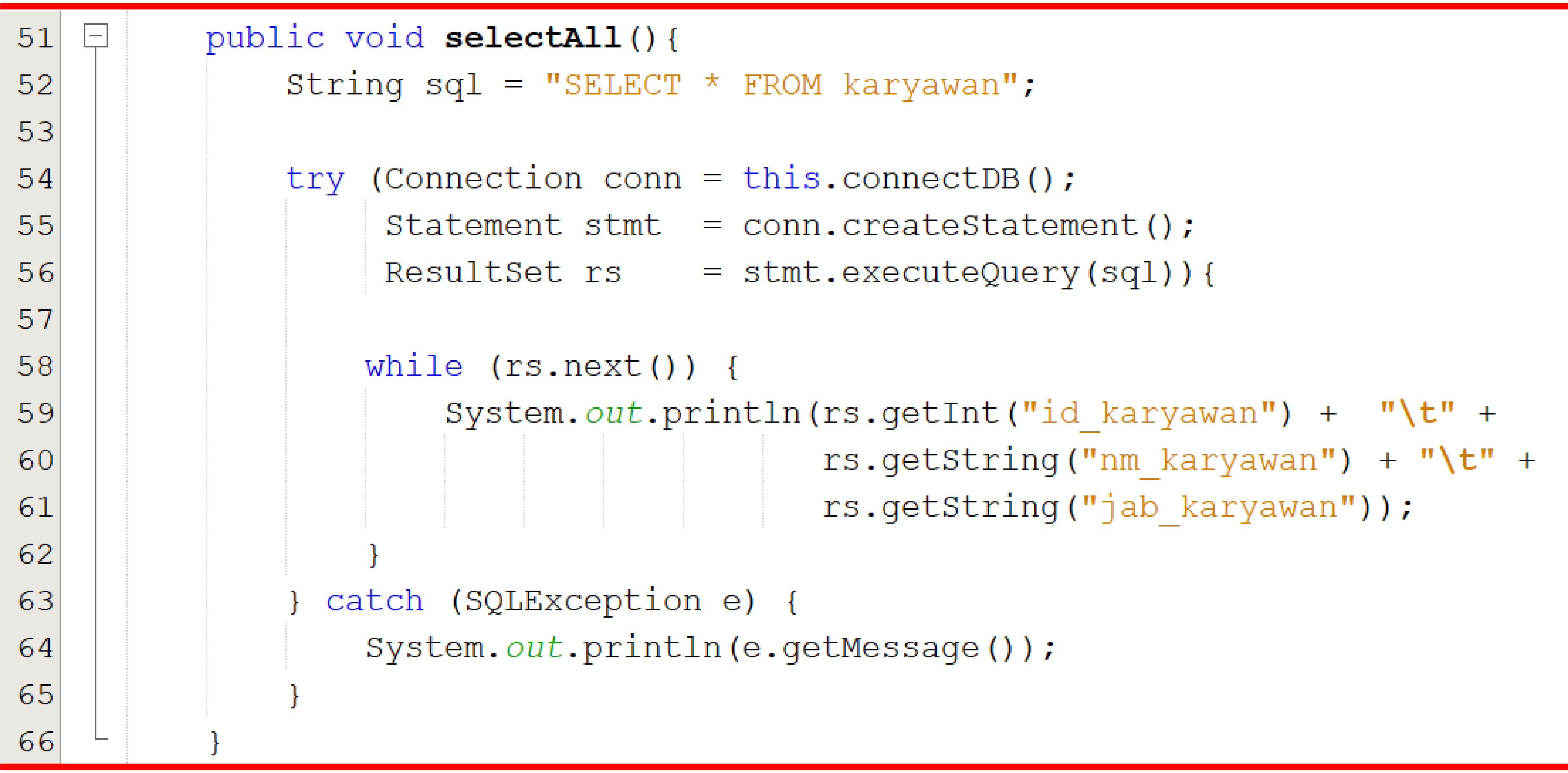
1. Buat Class baru dengan nama “Karyawan”
2. Kemudian buat method connetDB untuk koneksi database



```java
    private Connection connectDB() {
        String url = "jdbc:sqlite:data.db";
        Connection conn = null;
        try {
            conn = DriverManager.getConnection(url);
        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
        return conn;
    }
```

SQLite

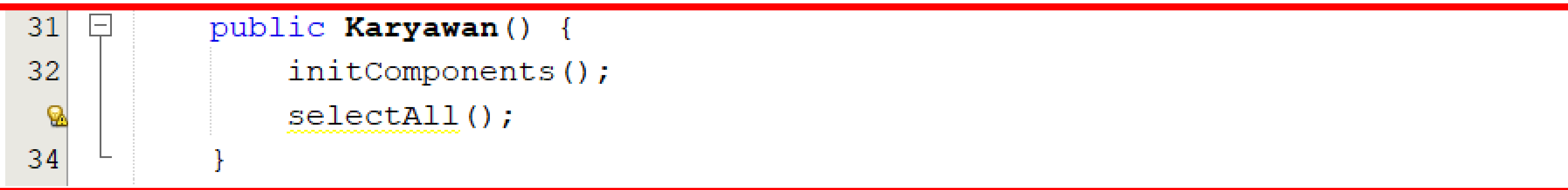
Buatlah method “selectAll” untuk membaca table “karyawan” dan menampilkannya di console



```java
    public void selectAll(){
        String sql = "SELECT * FROM karyawan";

        try (Connection conn = this.connectDB();
             Statement stmt  = conn.createStatement();
             ResultSet rs    = stmt.executeQuery(sql)){

            while (rs.next()) {
                System.out.println(rs.getInt("id_karyawan") +  "\t" +
                                   rs.getString("nm_karyawan") + "\t" +
                                   rs.getString("jab_karyawan"));
            }
        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
    }
```

SQLite

Untuk menjalankan method “selectAll”, panggil method pada constructor class.

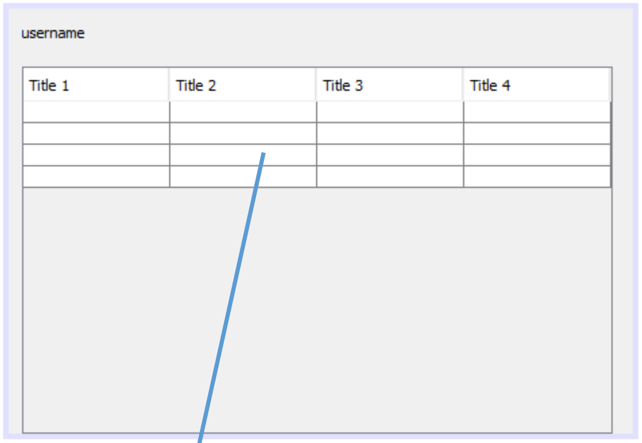


```
    public Karyawan() {
        initComponents();
        selectAll();
    }
```

SQLite

Untuk menampilkan record dalam bentuk table, buatlah method "loadTabelKaryawan" seperti dibawah ini.

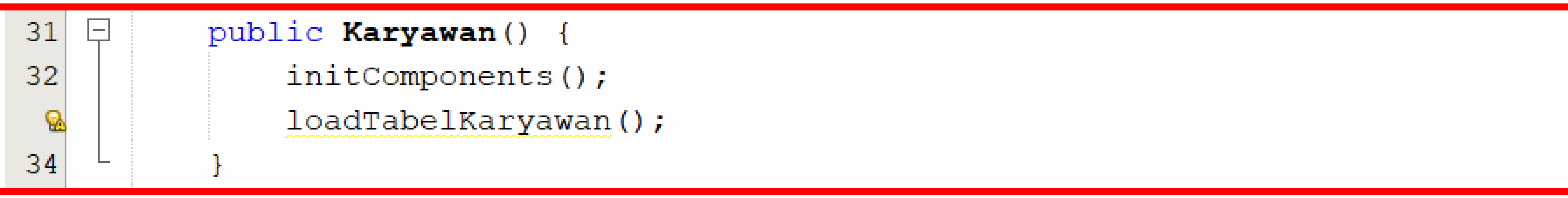
```java
    Connection conn = this.connectDB();
    public void loadTabelKaryawan(){
        String sql = "SELECT * FROM karyawan";
        Object[] kolom = { "ID", "Nama", "Jabatan" };
        DefaultTableModel dataModel = new DefaultTableModel(null, kolom);
        tbKaryawan.setModel(dataModel);
        tbKaryawan.getColumnModel().getColumn(0).setMaxWidth(30);

        try {
            Statement stmt  = conn.createStatement();
            ResultSet rs    = stmt.executeQuery(sql);

            while(rs.next()){
                int id = rs.getInt("id_karyawan");
                String nama = rs.getString("nm_karyawan");
                String jabatan = rs.getString("jab_karyawan");

                Object[] data={id, nama, jabatan};
                dataModel.addRow(data);
            }

        } catch (SQLException e) {
            System.out.println(e.getMessage());
        }
    }
```



tbKaryawan

SQLite

Untuk menjalankan method “loadTabelKaryawan”, panggil method pada constructor class.

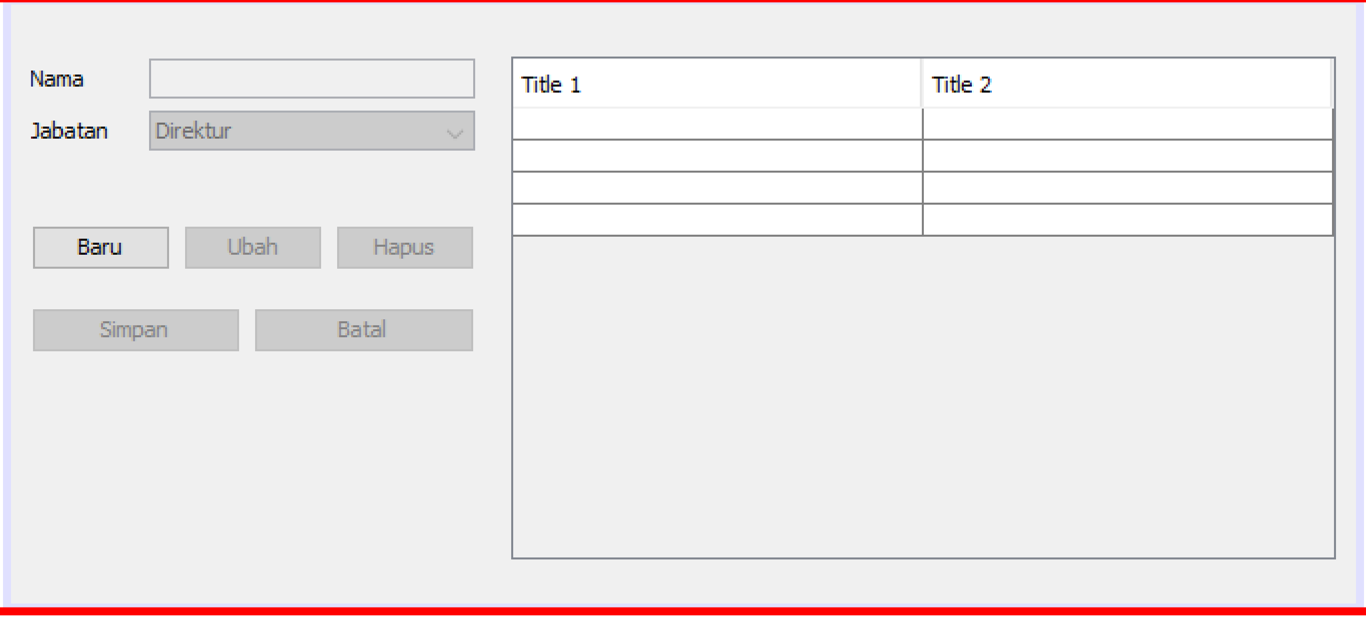


```
    public Karyawan() {
        initComponents();
        loadTabelKaryawan();
    }
```

SQLite: CRUD

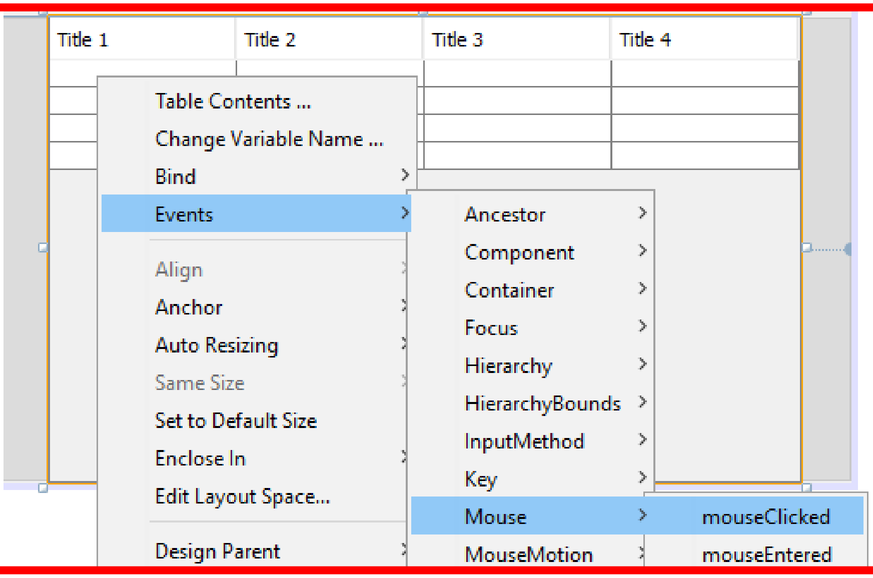
Buatlah antarmuka seperti dibawah ini

txtNama	(editable=false)
cbJabatan	(disable)
btnSimpan	(disable)
btnBatal	(disable)
btnHapus	(disable)
btnBaru	
btnUbah	(disable)

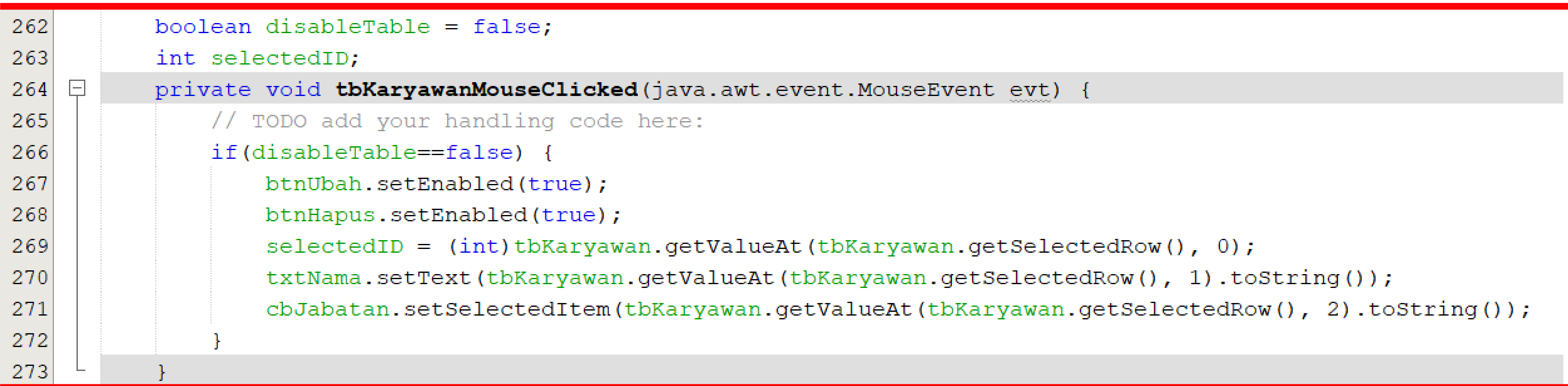


SQLite: CRUD

Tambahkan event "mouseClicked" pada komponen table



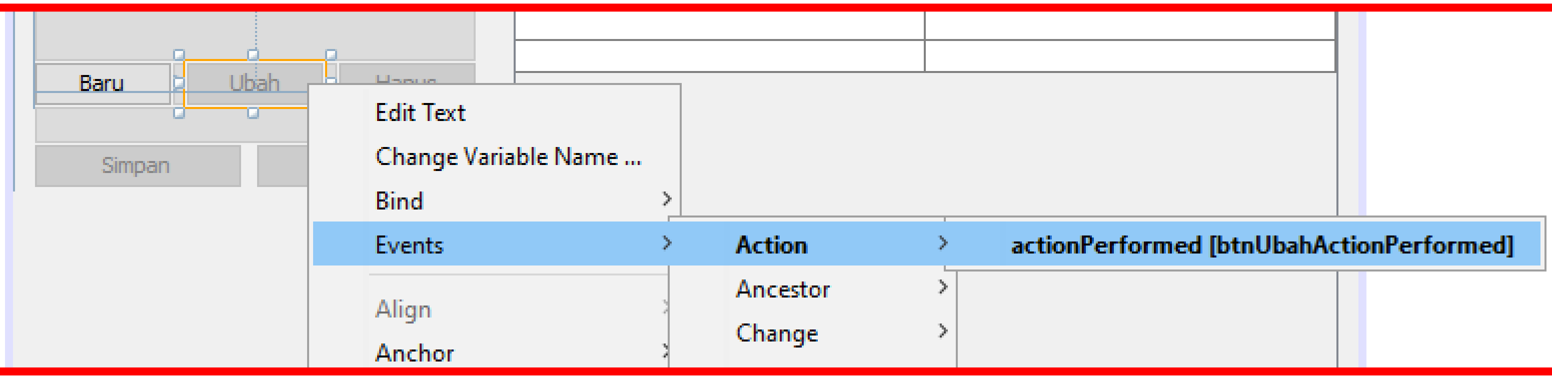
Kode untuk event mouseClicked

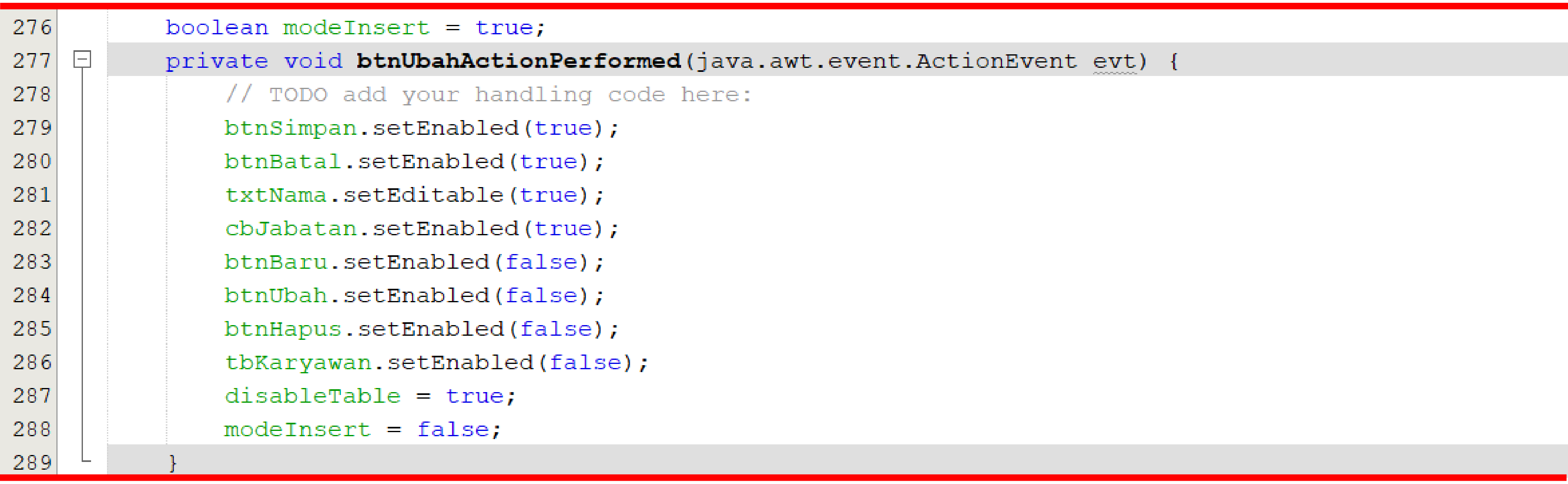


```
    boolean disableTable = false;
    int selectedID;
    private void tbKaryawanMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        if(disableTable==false) {
            btnUbah.setEnabled(true);
            btnHapus.setEnabled(true);
            selectedID = (int)tbKaryawan.getValueAt(tbKaryawan.getSelectedRow(), 0);
            txtNama.setText(tbKaryawan.getValueAt(tbKaryawan.getSelectedRow(), 1).toString());
            cbJabatan.setSelectedItem(tbKaryawan.getValueAt(tbKaryawan.getSelectedRow(), 2).toString());
        }
    }
```

SQLite: CRUD

Event btnUbah

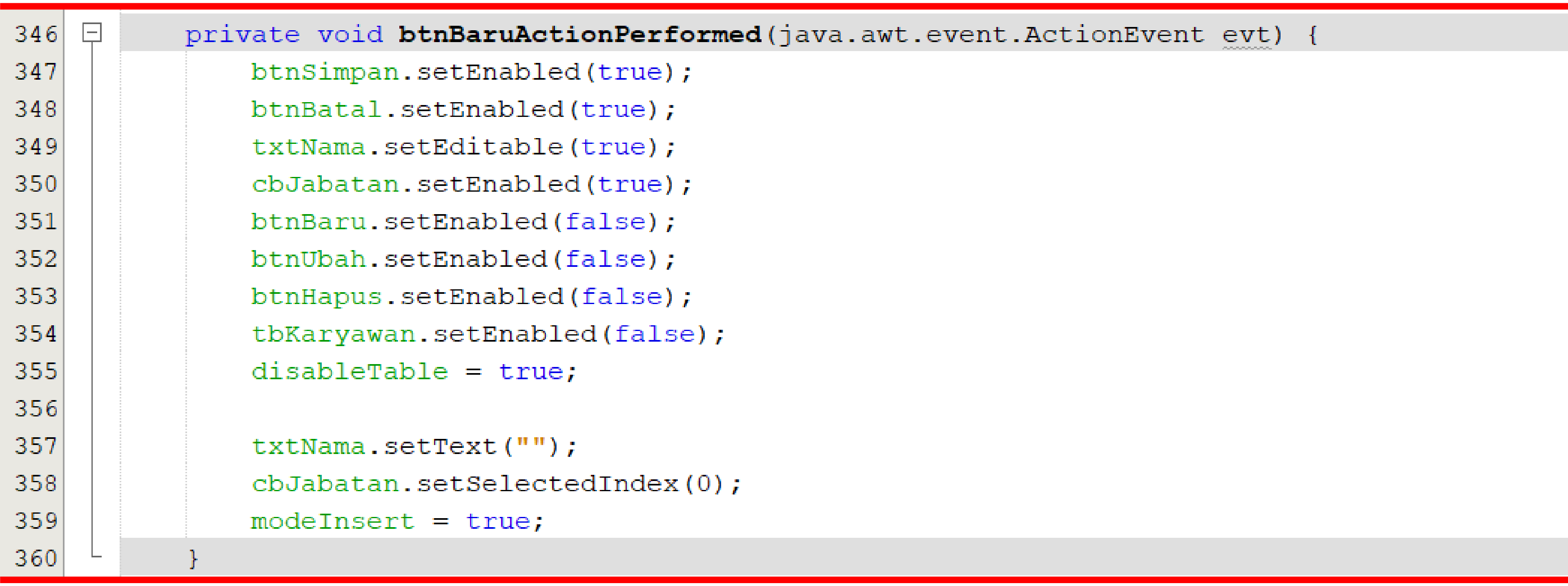




```
    boolean modeInsert = true;
    private void btnUbahActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        btnSimpan.setEnabled(true);
        btnBatal.setEnabled(true);
        txtNama.setEditable(true);
        cbJabatan.setEnabled(true);
        btnBaru.setEnabled(false);
        btnUbah.setEnabled(false);
        btnHapus.setEnabled(false);
        tbKaryawan.setEnabled(false);
        disableTable = true;
        modeInsert = false;
    }
```

SQLite: CRUD

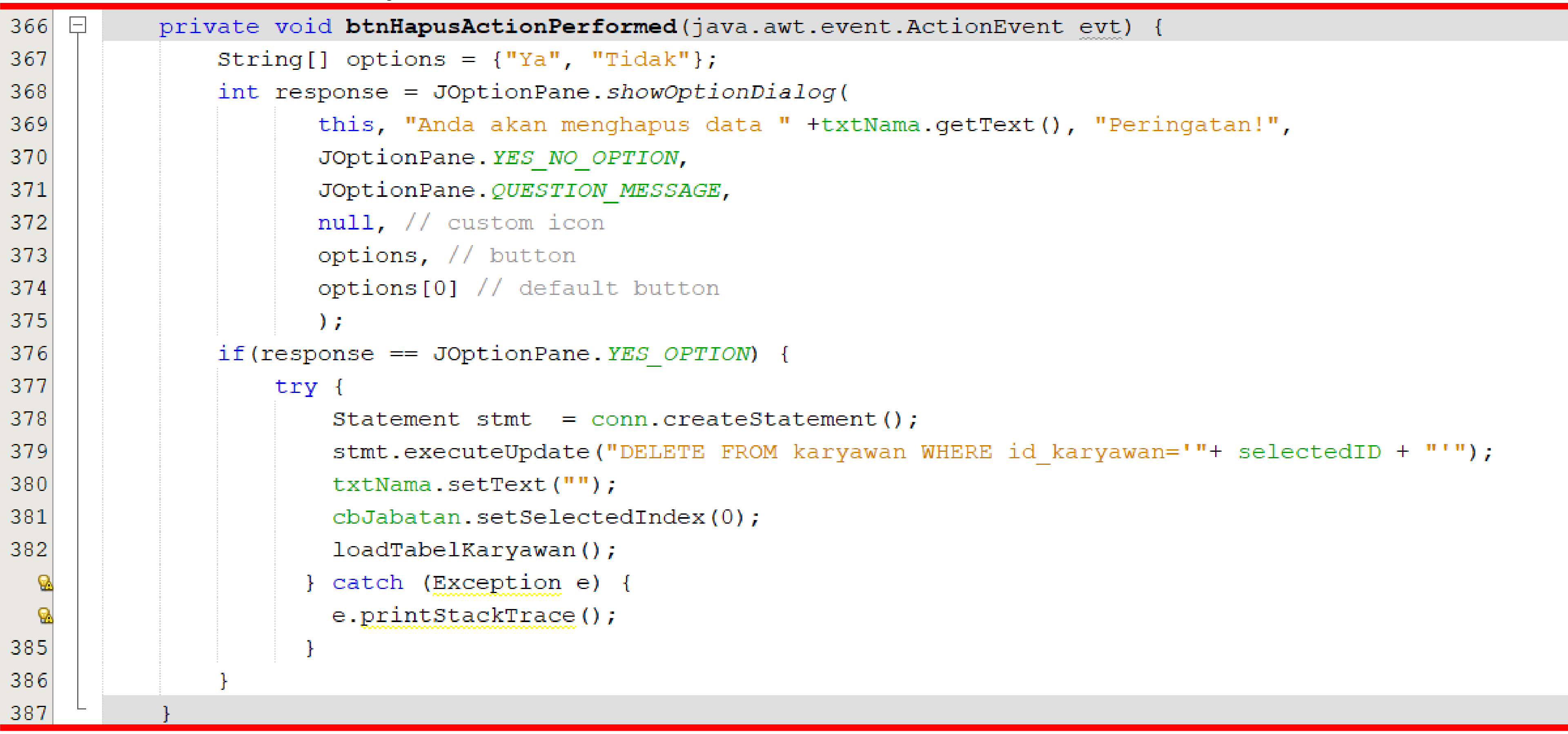
Event btnBaru



```
private void btnBaruActionPerformed(java.awt.event.ActionEvent evt) {
    btnSimpan.setEnabled(true);
    btnBatal.setEnabled(true);
    txtNama.setEditable(true);
    cbJabatan.setEnabled(true);
    btnBaru.setEnabled(false);
    btnUbah.setEnabled(false);
    btnHapus.setEnabled(false);
    tbKaryawan.setEnabled(false);
    disableTable = true;

    txtNama.setText("");
    cbJabatan.setSelectedIndex(0);
    modeInsert = true;
}
```

SQLite: CRUD

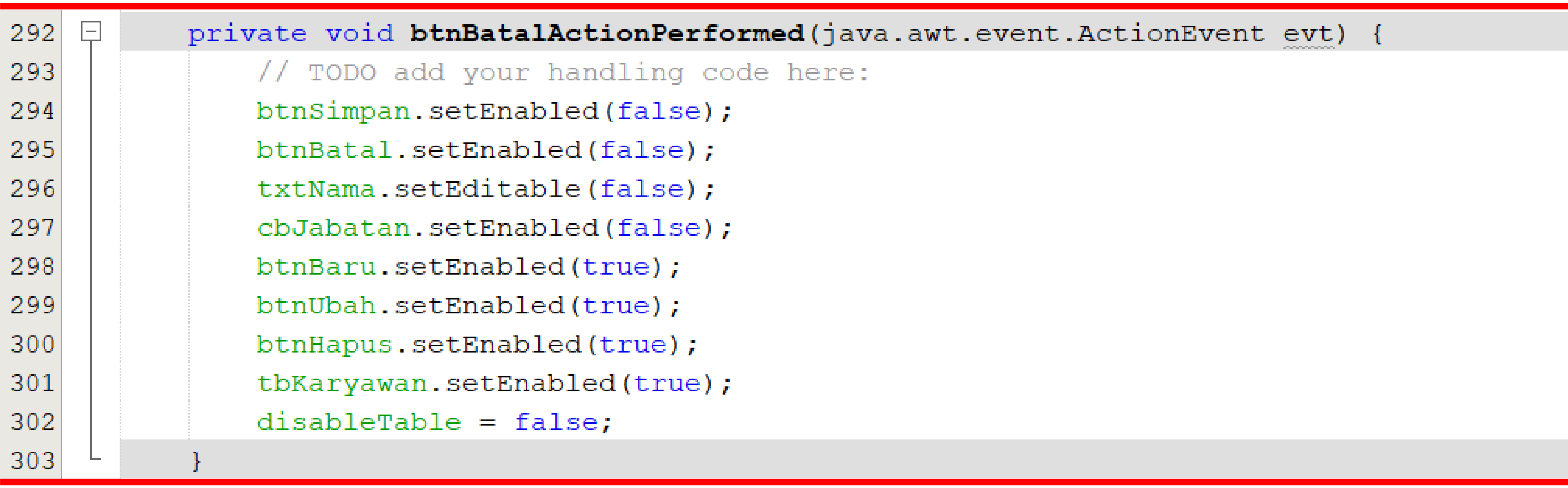
Event btnHapus



```
    private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {
        String[] options = {"Ya", "Tidak"};
        int response = JOptionPane.showOptionDialog(
                this, "Anda akan menghapus data " +txtNama.getText(), "Peringatan!",
                JOptionPane.YES_NO_OPTION,
                JOptionPane.QUESTION_MESSAGE,
                null, // custom icon
                options, // button
                options[0] // default button
                );
        if(response == JOptionPane.YES_OPTION) {
            try {
                Statement stmt  = conn.createStatement();
                stmt.executeUpdate("DELETE FROM karyawan WHERE id_karyawan='"+ selectedID + "'");
                txtNama.setText("");
                cbJabatan.setSelectedIndex(0);
                loadTabelKaryawan();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
```

SQLite: CRUD

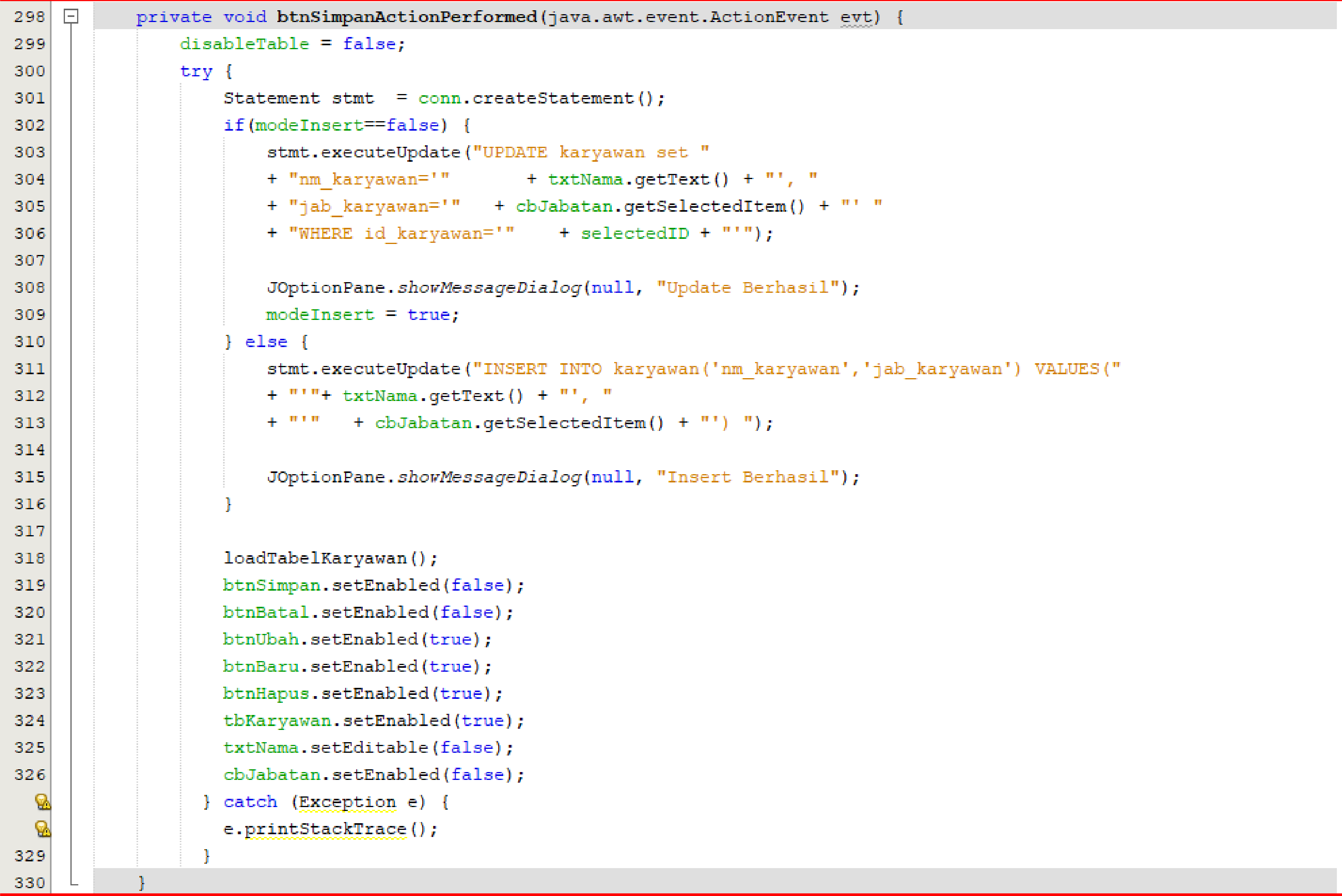
Event btnBatal



```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    btnSimpan.setEnabled(false);
    btnBatal.setEnabled(false);
    txtNama.setEditable(false);
    cbJabatan.setEnabled(false);
    btnBaru.setEnabled(true);
    btnUbah.setEnabled(true);
    btnHapus.setEnabled(true);
    tbKaryawan.setEnabled(true);
    disableTable = false;
}
```

SQLite: CRUD

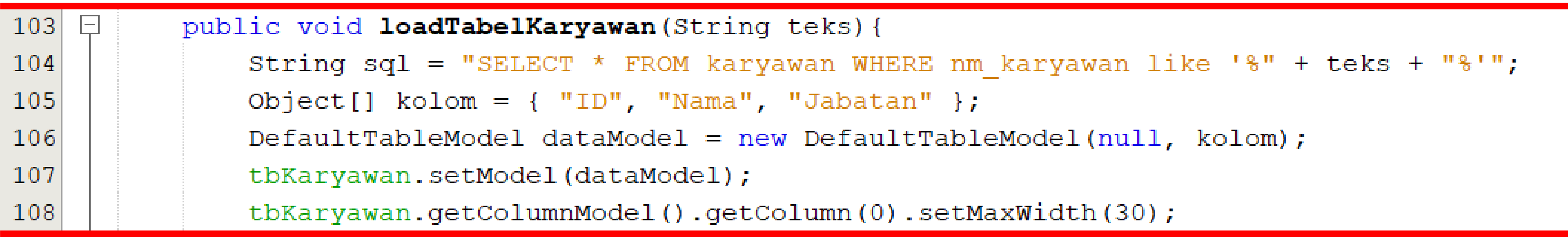
Event btnSimpan



```java
    private void btnSimpanActionPerformed(java.awt.event.ActionEvent evt) {
        disableTable = false;
        try {
            Statement stmt  = conn.createStatement();
            if(modeInsert==false) {
                stmt.executeUpdate("UPDATE karyawan set "
                + "nm_karyawan='"        + txtNama.getText() + "', "
                + "jab_karyawan='"    + cbJabatan.getSelectedItem() + "' "
                + "WHERE id_karyawan='"     + selectedID + "'");

                JOptionPane.showMessageDialog(null, "Update Berhasil");
                modeInsert = true;
            } else {
                stmt.executeUpdate("INSERT INTO karyawan('nm_karyawan','jab_karyawan') VALUES("
                + "'"+ txtNama.getText() + "', "
                + "'"   + cbJabatan.getSelectedItem() + "') ");

                JOptionPane.showMessageDialog(null, "Insert Berhasil");
            }

            loadTabelKaryawan();
            btnSimpan.setEnabled(false);
            btnBatal.setEnabled(false);
            btnUbah.setEnabled(true);
            btnBaru.setEnabled(true);
            btnHapus.setEnabled(true);
            tbKaryawan.setEnabled(true);
            txtNama.setEditable(false);
            cbJabatan.setEnabled(false);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
```

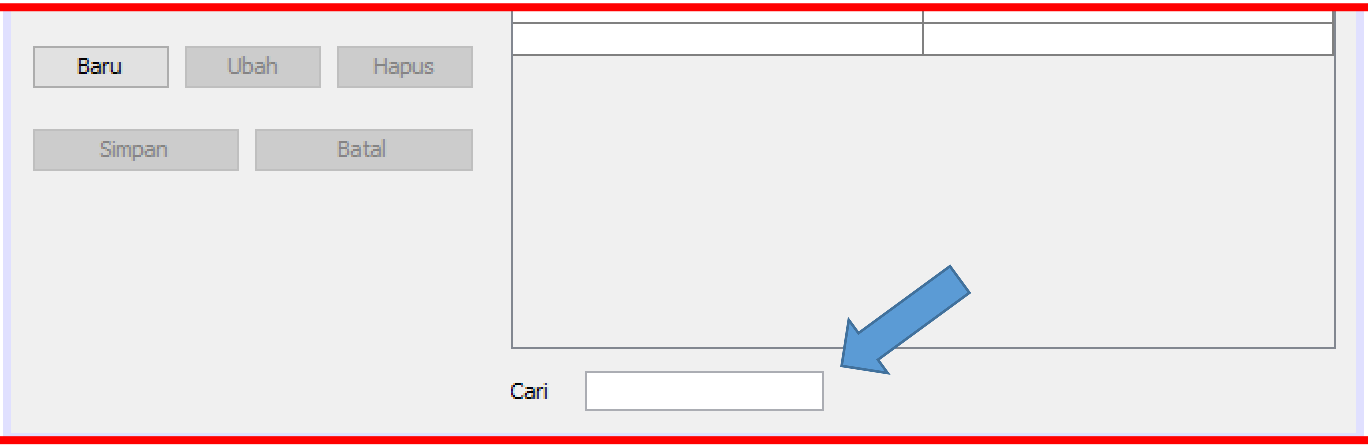
SQLite: Pencarian

Overload method loadTabelKaryawan (*copy-paste* method loadTabelKaryawan, kemudian bedakan pada dua baris pertama)



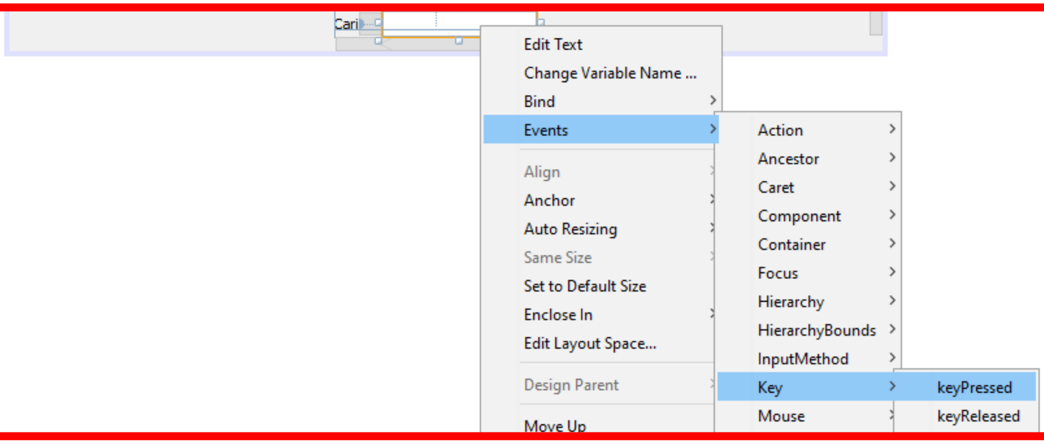
```
    public void loadTabelKaryawan(String teks){
        String sql = "SELECT * FROM karyawan WHERE nm_karyawan like '%" + teks + "%'";
        Object[] kolom = { "ID", "Nama", "Jabatan" };
        DefaultTableModel dataModel = new DefaultTableModel(null, kolom);
        tbKaryawan.setModel(dataModel);
        tbKaryawan.getColumnModel().getColumn(0).setMaxWidth(30);
```

Tambahkan komponen label dan TextField “txtCari” pada di bawah tabel

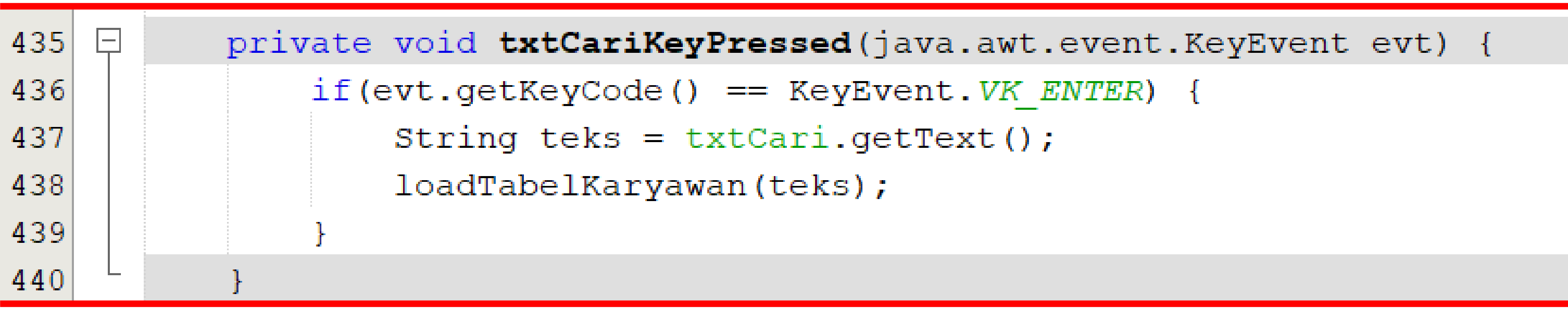


SQLite: Pencarian

Tambahkan event keyPressed pada txtCari



Kode event ketika tombol enter ditekan pada kolompencarian txtCari



```
    private void txtCariKeyPressed(java.awt.event.KeyEvent evt) {
        if(evt.getKeyCode() == KeyEvent.VK_ENTER) {
            String teks = txtCari.getText();
            loadTabelKaryawan(teks);
        }
    }
```

Latihan

1. Tambahkan beberapa field data pada tabel karyawan seperti jenis kelamin, alamat, nomor telepon, dll, lalu sesuaikan di antarmuka.
2. Tambahkan kemampuan pencarian hingga tidak hanya dapat mencari nama saja namun dapat mencari semua field pada satu kolom pencarian txtCari.
3. Tambahkan library “MySQL JDBC Driver” lalu ubah database yang digunakan pada aplikasi ke database MySQL.