

**Laporan Database Sql Connection
Java Maven**

Dosen pengajar:
RUDHI WAHYUDI FEBRIANTO



Disusun Oleh:

Nama : Ade Hikmat Pauji Ridwan
Kelas : 222K
NPM : 22552011130

**PROGRAM STUDI TEKNIK INFORMATIKA
UNIVERSITAS TEKNOLOGI BANDUNG
2024**

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dengan perubahan zaman yang membawa segala sesuatu menjadi global, teknologi juga mengalami perkembangan yang pesat, dan pemrograman komputer menjadi salah satu aspek yang sangat krusial dalam era digital ini. Salah satu aspek penting dari Java adalah kemampuannya untuk mengelola koneksi ke berbagai basis data, sistem, dan layanan melalui konsep Java Connection.

Java Connection mengacu pada berbagai mekanisme dan API (Application Programming Interface) yang disediakan oleh Java untuk menghubungkan aplikasi dengan sumber daya eksternal seperti database, layanan web, dan sistem lainnya. Salah satu API paling terkenal dalam konteks ini adalah JDBC (Java Database Connectivity), yang memungkinkan aplikasi Java untuk berkomunikasi dengan basis data relasional dengan cara yang efisien dan terstruktur.

Java dikembangkan oleh James Gosling pada tahun 1995 sebagai bagian dari Sun Microsystems Java Platform. Java sangat dipengaruhi oleh sintaksis bahasa C dan C++, tetapi dengan penyederhanaan dan keketatan yang lebih tinggi, serta akses yang lebih terbatas ke sistem operasi. Hal ini menjadikan Java sebagai bahasa pemrograman yang mudah dipelajari dan dipahami. Sebagai bahasa pemrograman tingkat tinggi (High Level Language), Java dirancang tidak hanya untuk dikompilasi oleh mesin, tetapi juga agar dapat dimengerti oleh manusia.

Dalam penggunaannya, Java Connection melibatkan berbagai konsep dan operator pemrograman seperti Class, Object, Constructor, Inheritance, dan lain-lain. Konsep-konsep ini memungkinkan pengembang untuk membuat kode yang lebih terstruktur dan modular, sehingga memudahkan dalam pengembangan dan pemeliharaan aplikasi. Melalui API seperti JDBC, Hibernate, dan Java EE, pengembang dapat mengimplementasikan koneksi yang aman, handal, dan efisien ke berbagai basis data dan layanan web.

Hingga saat ini, mayoritas produk yang diperkenalkan ke industri teknologi informasi dan telekomunikasi modern menggunakan bahasa pemrograman Java dan Java Connection sebagai komponen utama dari arsitektur mereka, menunjukkan betapa pentingnya peran Java dalam perkembangan teknologi informasi global. Dengan kemampuannya untuk mengelola

koneksi dengan berbagai sumber daya eksternal, Java terus menjadi pilihan utama bagi pengembang yang ingin membangun aplikasi yang scalable dan robust.

1.2. Tujuan

Adapun tujuan dari praktikum ini adalah:

1. Mampu memahami konsep JDBC.
2. Mampu membuat program yang bisa terhubung ke database.

BAB II DASAR TEORI

2.1 JDBC

Java Database Connectivity (JDBC) adalah API Java yang digunakan untuk menghubungkan aplikasi Java dengan berbagai jenis database. JDBC menyediakan antarmuka standar yang memungkinkan pengembang untuk membuat, mengelola, dan memanipulasi database secara efisien. Konsep dasar JDBC meliputi beberapa komponen penting:

1. **Driver Manager:** Mengelola daftar driver database. Menggunakan metode `DriverManager.getConnection()`, aplikasi Java dapat membuat koneksi ke database yang ditentukan.
2. **Driver:** Komponen ini menyediakan koneksi antara aplikasi Java dan database tertentu. Driver JDBC harus diimpor ke dalam proyek Java agar dapat digunakan.
3. **Connection:** Objek yang merepresentasikan koneksi ke database. Melalui objek ini, aplikasi dapat berkomunikasi dengan database untuk menjalankan perintah SQL.
4. **Statement:** Antarmuka untuk mengirimkan perintah SQL ke database. Terdapat tiga jenis utama: `Statement`, `PreparedStatement`, dan `CallableStatement`.
 - `Statement`: Digunakan untuk menjalankan perintah SQL sederhana tanpa parameter.
 - `PreparedStatement`: Digunakan untuk menjalankan perintah SQL yang memiliki parameter. Lebih aman dan efisien dibandingkan dengan `Statement`.
 - `CallableStatement`: Digunakan untuk menjalankan prosedur tersimpan (stored procedures) di dalam database.
5. **ResultSet:** Objek yang digunakan untuk menyimpan hasil query SQL. `ResultSet` memungkinkan iterasi melalui hasil query dan pengambilan data dari kolom tertentu.

BAB III

METOLOGI PERCOBAAN

3.1 Alat dan Bahan

1. PC / Laptop
2. Netbeans
3. Java Maven

3.2 Algoritma

3.2.1 Algoritma Connect Database

1. Mulai.
2. Pastikan database sql telah terinstall.
3. Pastikan anda telah menginstall driver db sql, anda bisa menambahkan di pom.xml lalu update devendency anda.
4. Siapkan username dan juga password beserta db untuk konfigurasi connection.
5. Panggil instance dari class `java.sql.DriverManager`.
6. Jika config sudah sesuai maka koneksi ada akan berhasil.
7. Selesai.

BAB IV

HASIL DAN ANALISA

4.1 Hasil Percobaan

4.1.1 Membuat interface untuk lib sql db agar bisa di gunakan walau beda db

```
package com.mycompany.tkai.Lib.SqlDbManager;

import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 * @author macbook
 */
public interface DbManager {
    void connect() throws SQLException;
    void disconnect() throws SQLException;
    ResultSet executeQuery(String query) throws SQLException;
    int executeUpdate(String query) throws SQLException;
}
```

4.2.2 buat implementasi dari interface

```
package com.mycompany.tkai.Lib.SqlDbManager;

/**
 *
 * @author macbook
 */
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class SqlDbManager implements DbManager {

    private Connection connection;
    private final String url;
    private final String username;
    private final String password;

    public SqlDbManager(String url, String username, String password) {
        this.url = url;
        this.username = username;
        this.password = password;
    }

    @Override
    public void connect() throws SQLException {
        if (connection == null || connection.isClosed()) {
            if (!"".equals(username) && !"".equals(password)) {
                connection = DriverManager.getConnection(url, username, password);
            } else {
                connection = DriverManager.getConnection(url);
            }
        }
    }

    @Override
    public void disconnect() throws SQLException {
        if (connection != null && !connection.isClosed()) {
            connection.close();
        }
    }

    @Override
```

```

public ResultSet executeQuery(String query) throws SQLException {
    if (connection == null || connection.isClosed()) {
        throw new SQLException("Not connected to the database.");
    }
    Statement statement = connection.createStatement();
    return statement.executeQuery(query);
}

@Override
public int executeUpdate(String query) throws SQLException {
    if (connection == null || connection.isClosed()) {
        throw new SQLException("Not connected to the database.");
    }
    Statement statement = connection.createStatement();
    return statement.executeUpdate(query);
}
}

```

4.2.2 lakukan percobaan membuat instance dari SqlDbManager

```

public class TKAI {

    static void testDB() {
        // Ganti dengan detail koneksi DB Anda yang sebenarnya
        String url = "jdbc:postgresql://localhost:5432/sales_channel";
        String username = "postgres";
        String password = "postgres";

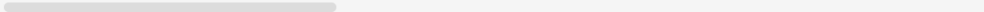
        DbManager dbManager = new SqlDbManager(url, username, password);

        try {
            dbManager.connect();
            System.out.println("db connected");
            dbManager.disconnect();
            System.out.println("db disconnected");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```


Hasil tidak ada error exception maka connection db berhasil

```
| --- compiler:3.11.0:compile (default-compile) @ TKAI ---  
Changes detected - recompiling the module! :source  
Compiling 11 source files with javac [debug target 17] to target/classes  
  
| --- exec:3.1.0:exec (default-cli) @ TKAI ---  
db connected  
db disconnected  
-----  
BUILD SUCCESS  
-----  
Total time: 2.078 s  
Finished at: 2024-06-24T19:22:34+07:00  
-----
```



Untuk merubah connection db menjadi db lain kita bisa merubah username, password dan url selagi dbnya sql dan juga driver nya tersedia maka tidak perlu ada perubahan yang signifikan untuk koneksi db ini.

contoh connection untuk ms access:

```
String url = "jdbc:ucanaccess://path/to/your/accessdb.accdb";  
untuk password dan juga username ini bisa di kosongkan.
```

BAB V

KESIMPULAN

Dalam praktikum ini, kami telah mempelajari konsep dasar Java Database Connectivity (JDBC) dan langkah-langkah untuk membuat program Java yang terhubung ke database. Dengan memahami komponen-komponen utama JDBC seperti Driver Manager, Driver, Connection, Statement, dan ResultSet, kami dapat mengembangkan aplikasi yang dapat berinteraksi dengan database secara efektif dan efisien.

Melalui implementasi praktis, kami berhasil memuat driver database, membuat koneksi ke database, menjalankan perintah SQL, dan mengolah hasil query. Langkah-langkah tersebut merupakan fondasi penting dalam pengembangan aplikasi berbasis data yang andal dan scalable.

Praktikum ini tidak hanya memperkuat pemahaman teoritis tentang JDBC, tetapi juga memberikan pengalaman praktis dalam mengelola koneksi database dalam aplikasi Java, yang merupakan keterampilan penting dalam dunia pemrograman dan pengembangan perangkat lunak modern.