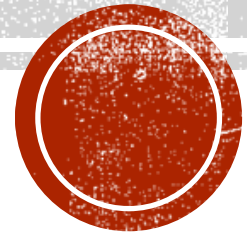


CRUD MYSQL - NODEJS

rismayana@poltekdedc.ac.id

Maret 2019



BUAT PROJECT BARU

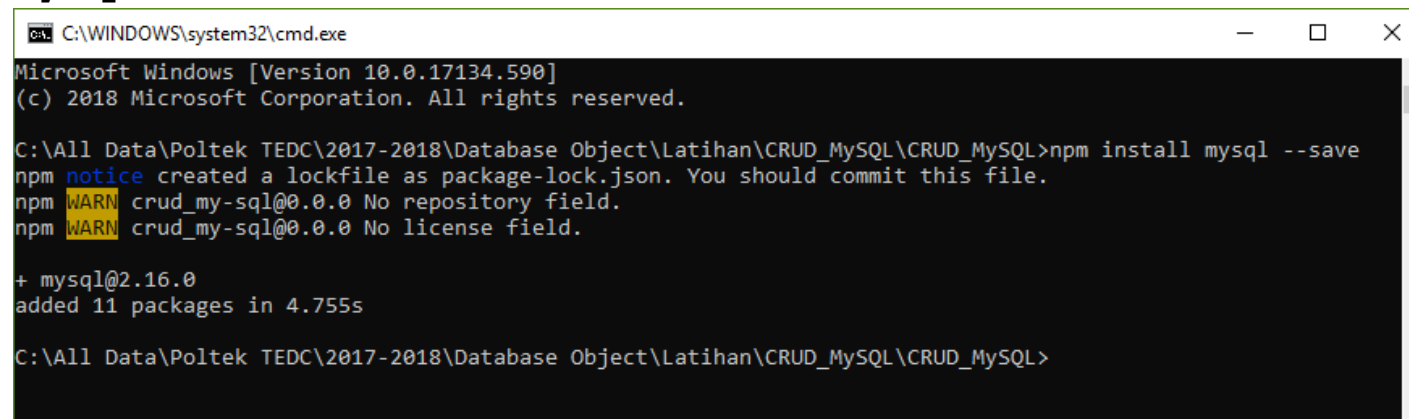
- Pada Visual Studio, buat project baru => CRUD_MySQL
- File .js yang dibuat :

```
— create_db.js  
— create_table.js  
— db_config.js  
— delete.js  
— insert.js  
— insert_multi.js  
— read.js  
— update.js
```



INSTALL MODUL MYSQL

- Kita membutuhkan modul mysql untuk menghubungkan Nodejs dengan MySQL.
- Modul ini tidak dibawa secara default oleh Nodejs. Karena itu, kita harus menginstalnya.
- Pada solution explorer, klik kanan nama projectnya, kemudian pilih “Open Command Prompt Here”
- Ketik perintah berikut untuk instal modul mysql:
 - **npm install mysql --save**



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.17134.590]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\All Data\Poltek TEDC\2017-2018\Database Object\Latihan\CRUD_MySQL\CRUD_MySQL>npm install mysql --save
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN crud_my-sql@0.0.0 No repository field.
npm WARN crud_my-sql@0.0.0 No license field.

+ mysql@2.16.0
added 11 packages in 4.755s

C:\All Data\Poltek TEDC\2017-2018\Database Object\Latihan\CRUD_MySQL\CRUD_MySQL>
```



MENGHUBUNGKAN NODEJS DENGAN MYSQL

- Buat file baru bernama db_config.js, kemudian isi dengan kode berikut:

```
1  var mysql = require('mysql');
2
3  var db = mysql.createConnection({
4      host: "localhost",
5      user: "root",
6      password: ""
7  });
8
9  db.connect(function(err) {
10     if (err) throw err;
11     console.log("Connected!");
12 });
```

- Setelah itu, coba eksekusi file db_config.js
- Bila pesan **Connected!** ditampilkan, maka itu artinya koneksi program nodejs kita dengan server mysql berhasil.



MEMBUAT DATABASE DENGAN NODEJS

- Buat file baru bernama create_db.js, kemudian isi dengan kode berikut

```
1  var db = require("./db_config");
2
3  db.connect(function(err) {
4      if (err) throw err;
5
6      let sql = "CREATE DATABASE latihan_nodejs";
7      db.query(sql, function (err, result) {
8          if (err) throw err;
9          console.log("Database created");
10     });
11 });
```

- Pada kode di atas, kita mengimpor file db_config.js sebagai modul.



- Berarti kita harus melakukan ekspor di dalam db_config.js agar bisa di-impor.
- Tambahkan dan ubah db_config.js menjadi seperti ini:

```
1  var mysql = require('mysql');
2
3  var db = mysql.createConnection({
4      host: "localhost",
5      user: "root",
6      password: ""
7  });
8
9  module.exports = db;
```

- Setelah itu, coba eksekusi program create_db.js.
- Setelah itu cek menggunakan phpMyAdmin untuk melihat hasilnya



- Setiap kita ingin melakukan query SQL, kita harus membuka koneksi terlebih dahulu dengan fungsi `db.connect()`.
- Lalu di dalamnya, kita bisa memanggil fungsi `db.query()`.

```
db.query(sql, function (err, result) {  
  if (err) throw err;  
  console.log("Database created");  
});
```

- Fungsi `db.query()` memiliki dua parameter yang harus diberikan.
 - parameter `sql` yang merupakan query MySQL dalam bentuk string.
 - yang kedua adalah fungsi yang akan dieksekusi saat query dilakukan.
- Fungsi ini juga memiliki dua parameter: `err` dan `result`.
 - Parameter `err` akan menjadi objek yang menyimpan `err` kalau gagal melakukan query.
 - Sedangkan `result` akan menjadi objek yang menyimpan data hasil query.



MEMBUAT TABEL MYSQL DENGAN NODEJS

- Kita sudah tahu cara mengeksekusi query sql di Nodejs. Selanjutnya, kita akan belajar cara membuat tabel.
- Caranya sama seperti membuat database. Perbedaanya pada query yang digunakan.
- Namun, sebelum itu ubah isi db_config.js menjadi seperti ini:

```
1  var mysql = require('mysql');
2
3  var db = mysql.createConnection({
4      host: "localhost",
5      user: "root",
6      password: "",
7      database: "latihan_nodejs"
8  });
9
10 module.exports = db;
```



- Setelah itu, buat file baru bernama create_table.js dengan isi sebagai berikut

```
1  var db = require("../db_config");
2
3  db.connect(function(err) {
4      if (err) throw err;
5
6      let sql = `CREATE TABLE customers
7      (
8          id int NOT NULL AUTO_INCREMENT,
9          name VARCHAR(255),
10         address VARCHAR(255),
11         PRIMARY KEY (id)
12     )`;
13
14     db.query(sql, function (err, result) {
15         if (err) throw err;
16         console.log("Table created");
17     });
18 });
```

- Pada query tersebut, kita akan membuat tabel dengan nama customers dengan kolom id, name, dan address.
- Kemudian coba kita eksekusi
- Setelah itu cek menggunakan phpMyAdmin untuk melihat hasilnya



INSERT DATA KE MYSQL DENGAN NODEJS

- Kita sudah berhasil membuat tabel, berikutnya kita akan coba isi data ke sana.
- Caranya sama, yang membedakan adalah query-nya.
- Buatlah file baru bernama insert.js, kemudian isi dengan kode berikut:

```
1  var db = require("../db_config");
2
3  db.connect(function(err) {
4      if (err) throw err;
5
6      let sql = `INSERT INTO customers (name, address)
7                VALUES ('Starbucks', 'Lombok Epicentrum Mall')`;
8
9      db.query(sql, function (err, result) {
10         if (err) throw err;
11         console.log("1 record inserted");
12     });
13 });
```



INSERT MULTI DATA

- Lalu bagaimana kalau ada banyak data yang ingin kita tambahkan?
- Ini bisa dilakukan dengan memberikan parameter [values] pada eksekusi query.
 - `db.query(sql, [values], function (err, result){ ... });`
- Buatlah file baru bernama `insert_multi.js`, kemudian isi dengan kode berikut:



```
1  var db = require("../db_config");
2
3  db.connect(function(err) {
4      if (err) throw err;
5
6      let sql = "INSERT INTO customers (name, address) VALUES ?";
7      var values = [
8          ['JS Coffee', 'Highway 71'],
9          ['3AM Coffee', 'Lowstreet 4'],
10         ['Apple Cafe', 'Apple st 652'],
11         ['Laravel Coffee', 'Mountain 21'],
12         ['Nodejs Cafe', 'Valley 345'],
13         ['PHP Hotel', 'Ocean blvd 2'],
14         ['One Cafe', 'Green Grass 1'],
15         ['Richard bar', 'Sky st 331'],
16         ['Susan Cafe', 'One way 98'],
17         ['Vicky Club', 'Yellow Garden 2'],
18         ['Ben Resto', 'Park Lane 38'],
19         ['William Company', 'Central st 954'],
20         ['Chuck Food', 'Main Road 989'],
21         ['Viola Coffee', 'Sideway 1633']
22     ];
23     db.query(sql, [values], function (err, result) {
24         if (err) throw err;
25         console.log("Number of records inserted: " + result.affectedRows);
26     });
27 });
```



- Perhatikan pada kode tersebut
- Kita menggunakan tanda tanya dalam query MySQL-nya. Tanda tanya ini akan bertugas sebagai **placeholder** untuk data.
- Nanti, tanda tanya tersebut akan mengambil nilai dari variabel `values` yang akan kita berikan sebagai parameter dalam pengesekusian query.
- Lalu di dalam fungsi callback untuk query, kita menggunakan objek `result` untuk mengetahui berapa jumlah baris atau data yang telah ditambahkan.
 - `console.log("Number of records inserted: " + result.affectedRows);`
- Sekarang coba eksekusi program tersebut.
- Dan lihat hasilnya menggunakan `phpmyadmin`



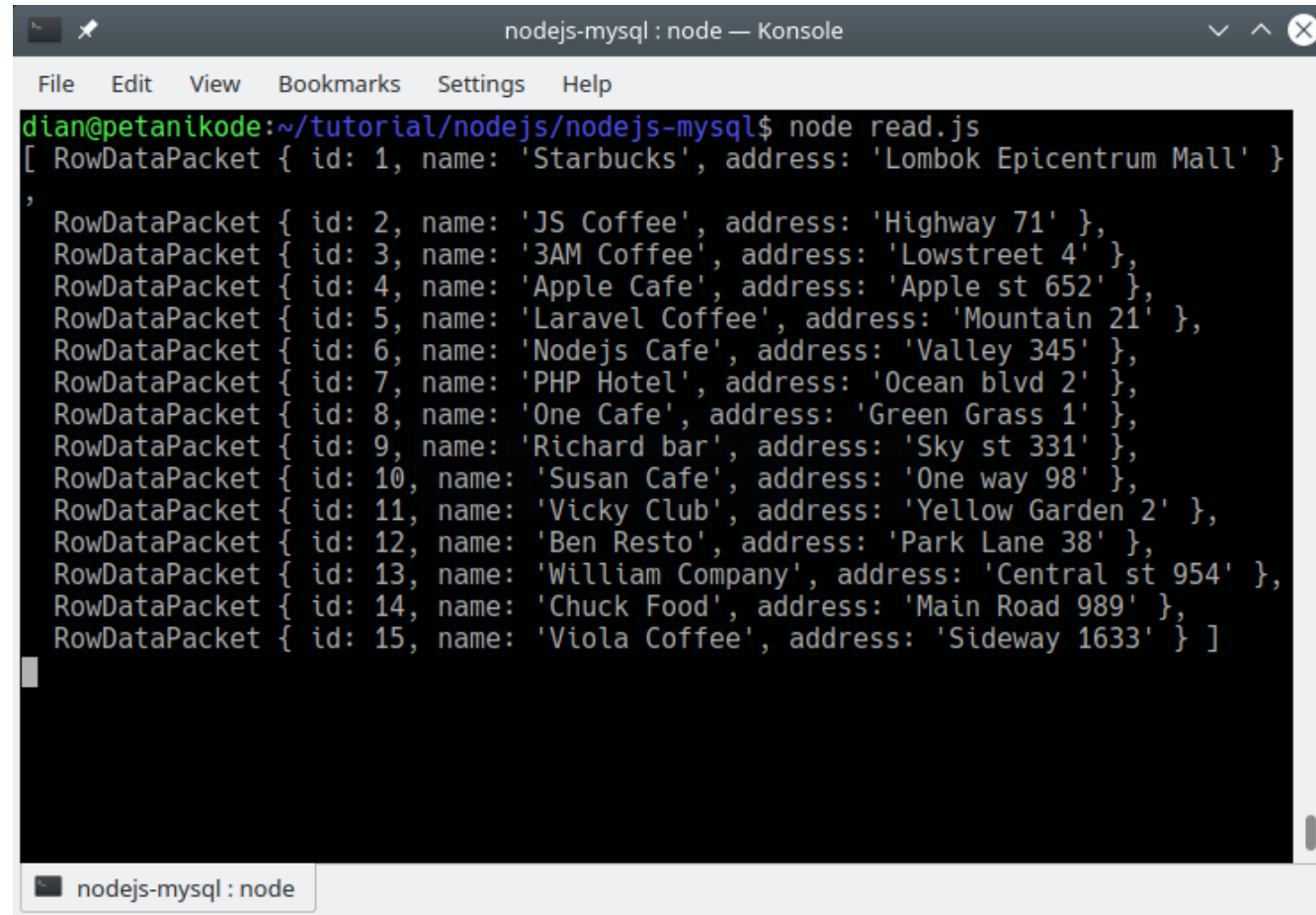
MEMBACA DATA MYSQL DI NODEJS

- Berikutnya kita akan mencoba membaca data dari MySQL dan menampilkannya di dalam program.
- Caranya sama hanya berbeda pada query saja.
- Silahkan buat file baru bernama read.js, kemudian isi dengan kode berikut

```
1  var db = require("../db_config");
2
3  db.connect(function(err) {
4      if (err) throw err;
5
6      let sql = "SELECT * FROM customers";
7      db.query(sql, function (err, result) {
8          if (err) throw err;
9          console.log(result);
10     });
11 });
```



- Kemudian eksekusi



```
nodejs-mysql : node — Konsole
File Edit View Bookmarks Settings Help
dian@petanikode:~/tutorial/nodejs/nodejs-mysql$ node read.js
[ RowDataPacket { id: 1, name: 'Starbucks', address: 'Lombok Epicentrum Mall' },
  RowDataPacket { id: 2, name: 'JS Coffee', address: 'Highway 71' },
  RowDataPacket { id: 3, name: '3AM Coffee', address: 'Lowstreet 4' },
  RowDataPacket { id: 4, name: 'Apple Cafe', address: 'Apple st 652' },
  RowDataPacket { id: 5, name: 'Laravel Coffee', address: 'Mountain 21' },
  RowDataPacket { id: 6, name: 'Nodejs Cafe', address: 'Valley 345' },
  RowDataPacket { id: 7, name: 'PHP Hotel', address: 'Ocean blvd 2' },
  RowDataPacket { id: 8, name: 'One Cafe', address: 'Green Grass 1' },
  RowDataPacket { id: 9, name: 'Richard bar', address: 'Sky st 331' },
  RowDataPacket { id: 10, name: 'Susan Cafe', address: 'One way 98' },
  RowDataPacket { id: 11, name: 'Vicky Club', address: 'Yellow Garden 2' },
  RowDataPacket { id: 12, name: 'Ben Resto', address: 'Park Lane 38' },
  RowDataPacket { id: 13, name: 'William Company', address: 'Central st 954' },
  RowDataPacket { id: 14, name: 'Chuck Food', address: 'Main Road 989' },
  RowDataPacket { id: 15, name: 'Viola Coffee', address: 'Sideway 1633' } ]
```

- Dari hasil output ini, kita dapat mengetahui kalau variabel result adalah sebuah array yang berisi objek dari tiap baris atau data.



LATIHAN

- Buat aplikasi untuk Update data
- Buat aplikasi untuk Delete data
- Buat aplikasi untuk insert banyak data dalam looping (1000 data)
- Buat aplikasi untuk menampilkan data nama dan alamat saja

