

2024

# PBW

## Praktikum 6 – Accessing DB

Silakan buat halaman web sesuai petunjuk. Gunakan template yang telah disediakan.

(Adopted and edited from Modul 9 - PHP MySQL Access (2017) by Maria Veronica)



## Intro

Pada praktikum ini anda akan dipelajari cara mengakses *database* melalui Spring Boot. Pada pengerjaan tugas ini, sudah disediakan template yang berisi seluruh dependencies yang dibutuhkan untuk pengerjaan tugas ini. Rename folder **TEMPLATE** menjadi T06xyyy. Lakukan build. Pada tugas ini anda diminta melakukan koneksi dengan cara JdbcTemplate dan Repository.

Halaman html sudah dibuat secukupnya. Jika dirasa kurang, silahkan melakukan perubahan secara mandiri. Silahkan pelajari terlebih dahulu kode apa yang sudah dibuat. Akan tetapi fokus dari praktikum ini adalah koneksi dengan basis data, sehingga untuk kerapihan tampilan, mungkin dapat dikesampingkan terlebih dahulu.

## Data dan Connection

Pertama-tama, bukalah server postgresQL kalian dan buatlah basis data dengan nama **ide**. Selanjutnya lakukan import dengan sql yang diberikan di folder **DATA**. Sebagai informasi, berikut adalah keterangan dari tabel users.

Nama Field	Tipe Data	Sifat
<b>ID_U</b>	int	Auto increment, Primary key
<b>userID</b>	varchar(10)	Unique
<b>name</b>	varchar(50)	-
<b>username</b>	varchar(9)	Unique
<b>pass</b>	varchar(50)	-
<b>ID_UG</b>	int	Foreign Key (ke tabel usergroups)

Tabel usergroups cukup sederhana hanya terdapat id dan nama (dapat dilihat pada schema). Selain itu terawatt VIEW database dengan nama **user\_data** yang dibuat untuk menggabungkan kedua table (query dapat dicek).

Lakukan pengatur application.properties agar sesuai dengan environment anda. Kemudian buatlah Bean DataSource untuk membuat koneksi ke postgresQL. Untuk kelas model, dapat memanfaatkan library Lombok.

## SELECT

Pertama-tama kita akan melihat halaman utama (menggunakan tampilan **users/index.html**). Halaman ini akan menampilkan tabel data users seperti dapat dilihat pada Gambar 1.

User ID	Name	Role
20160065	Maria Veronica Claudia	lecturer
20130053	Husnul Hakim	lecturer
2011730053	MARIA VERONICA	student
2011730103	JOHANES MARIO ADRIANO	student
2012730027	NICHOLAS MARTIN PRIBADI	student
2012730078	RIZQI PUTRA PRATAMA	student
2012730093	YOHAN STURE ENANDER	student
2013730001	ALVIN IRAWAN	student
2013730002	CHERIA	student
2013730003	FADHIL AHSAN	student
2013730004	CTARA	student

Gambar 1 – Tampilan pada url '/users/'

Query SQL yang digunakan untuk menampilkan data tersebut adalah sebagai berikut.

```
SELECT * FROM user_data
```

Intruksi pengerjaan adalah sebagai berikut:

1. Implementasikan controller **UserController** untuk membuka halaman tersebut dan mengeksekusi query diatas dengan bantuan UserRepository. Kemudian data yang dihasilkan dikirim ke sisi klien dan ditampilkan satu per satu.
2. Lakukan pula hal yang serupa untuk UserGroupController untuk menampilkan halaman seperti Gambar 2 (menggunakan tampilan di **usergroups/index.html**).

Role
lecturer
student
admin

---

Add New Role

Gambar 2 – Tampilan pada url '/userGroups/'

## SELECT ... WHERE ...

Misalkan Anda ingin mencari user yang namanya mengandung huruf-huruf tertentu. Anda hanya perlu menambahkan sintaks WHERE dalam kode SQL. Untuk mencari teks string maka digunakan keyword ILIKE. Pada halaman user terdapat form untuk mencari berdasarkan nama. Jika dimasukan teks pada input lalu ditekan tombol SEARCH, maka halaman akan tetap di halaman ini, namun isi dari tabel sudah berubah sesuai dengan hanya menampilkan data yang memiliki potongan teks di namanya. Contoh jika diinput keyword “vi” maka akan dimunculkan 4 buah data. Jika menggunakan filter, text pencariannya tetap muncul pada textbox search.

Notes: ILIKE pada PostgreSQL berfungsi seperti LIKE pada umumnya, tetapi tidak melihat huruf besar atau huruf kecil (case insensitive).

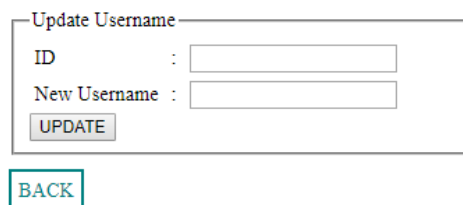
## INSERT

Salah satu operasi dasar dalam pemrograman basis data adalah *insert*. Operasi ini dilakukan untuk menambahkan *record* baru ke dalam tabel pada basis data. Pada modul ini, anda akan mencoba menambahkan role baru sesuai input dari pengguna dengan mengisi form yang disediakan. Pada halaman `userGroup`, jika tombol ADD ditekan, maka dilakukan penambahan row baru pada tabel `usergroups`. Setelah itu halaman akan kembali ke halaman itu namun data sudah bertambah.

Hint: gunakan perintah `redirect`.

## UPDATE

Bukalah halaman `users/update.html` yang akan menampilkan halaman pada Gambar 3 berikut.



Update Username

ID :

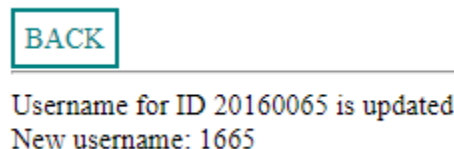
New Username :

UPDATE

BACK

Gambar 3 – Halaman update.

Halaman ini berguna untuk mengupdate username yang ada di table user dengan ID tertentu. Jika tombol UPDATE ditekan, maka isi dari form akan dikirim ke server. Implementasikan kode yang menerima input tersebut, lalu melakukan update pada basis data sesuai dengan yang dikirimkan. Setelah update dilakukan, halaman kembali ke halaman update. Jika berhasil, maka akan ditampilkan keterangan di bawah tombol BACK sesuai dengan Gambar 4.



Gambar 4 – Halaman update jika berhasil melakukan perubahan.

Jika gagal, cek di basis data anda, apakah data sudah berubah. Jika sudah berubah, mungkin ada sedikit kesalahan saat melakukan SELECT sehingga informasi di atas tidak muncul. Jika *database* juga belum *update*, periksa kembali seluruh kode anda untuk bagian update ini.

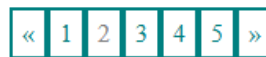
## LIMIT

Saat data yang ditampilkan sangat banyak, biasanya tampilan akan dibagi menjadi beberapa halaman. Sintaks SQL untuk membatasi jumlah *record* yang ditampilkan adalah LIMIT dan OFFSET (pada postgres). Bagian pagination ini menggunakan kode di `layout/pagination.html`.

1. Ubahlah agar pada halaman users (halaman utama) menampilkan maksimal 8 data per halamannya dan sisanya harus dilakukan dengan berganti halaman.

Syntax SQL = SELECT ... FROM ... WHERE ... LIMIT start OFFSET show.

2. Start menampilkan halaman dimulai dari baris berapa. Show menunjukkan berapa banyak yang ditampilkan di satu layar. Sebagai contoh pada modul ini akan terbagi menjadi 5 halaman dengan keterangan sebagai berikut.
  - a. Halaman pertama menampilkan data dari baris 0-7
  - b. Halaman kedua menampilkan data dari baris 8-15
  - c. Halaman ketiga menampilkan data dari baris 16-23
  - d. dst.
3. Untuk membedakan halaman users yang menggunakan pagination atau tidak, dibedakan dengan menggunakan parameter di url dengan nama page:
  - a. /users/ akan menampilkan table tanpa pagination.
  - b. /users/?page=2 akan menampilkan table dengan pagination untuk halaman kedua.
4. Tampilan users dengan pagination dapat dilihat pada Gambar 5.



User ID	Name	Role
2013730002	CHERIA	student
2013730003	FADHIL AHSAN	student
2013730004	CLARA	student
2013730005	ALDY MARCELLINO CHRISTIAN	student
2013730006	ANTONIUS	student
2013730008	ENRICO FINDLEY	student
2013730009	ROMMY KURNIAWAN WIJAYA	student
2013730010	YOSUA YUUTA BIMA PRAMANA	student

Gambar 5 – Tampilan untuk bagian pagination.

5. Untuk mengimplementasikan pagination terdapat 2 variable yang digunakan yaitu:
  - a. currentPage: nilai diambil langsung dari url page. Halaman selalu dimulai dari angka 1.
  - b. pageCount: digunakan untuk menampilkan berapa banyak link pagination yang ditampilkan, dapat digunakan perhitungan:
$$\text{pageCount} = (\text{int}) \text{Math.ceil}((\text{double}) \text{rowCount} / \text{show});$$
6. Setiap link pagination kecuali di halaman sendiri, dapat ditekan dan mengarah ke halaman yang sesuai. Selain itu, jika berada di halamn pertama tidak bisa menekan tombol Prev dan jika di halaman terakhir, tidak bisa menekan tombol Next.
7. Challenge (Opsional): Ubahlah program halaman ini dengan mengkombinasikan where dengan limit. Contoh tampilkan daftar mahasiswa yang mengandung huruf i pada halaman ketiga.
  - a. Jika sedang di mode pagination, ketika klik tombol search maka page mengikuti sebelumnya.

- b. Jika sedang di mode search dan pagination, ketika klik ganti halaman, maka filter akan tetap ikut.

Kumpulkan semua pekerjaan dalam bentuk zip dengan nama T06xyyy.zip

**= SELAMAT MENGERJAKAN =**