DINA FEBRIANTI KRISNA
1506757421

TUTORIAL 5
KELAS B

Tutorial kali ini menjelaskan tentang cara menghubunkan relasi-relasi yang memiliki *relationship* dalam *database*.

Latihan Mengubah *Method selectAllStudents*

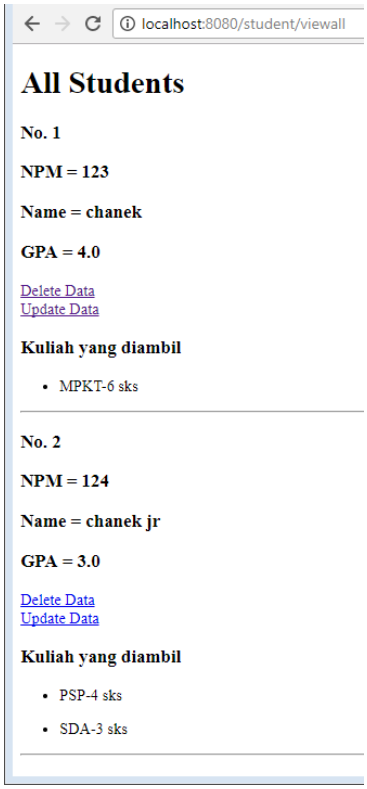
1. Kode

```
@Select("select npm, name, gpa from student")
@Results(value = {
      @Result(property="npm", column="npm"),
      @Result(property="name", column="name"),
      @Result(property="gpa", column="gpa"),
      @Result(property="courses", column="npm",
                  javaType = List.class,
                  many=@Many(select="selectCourses"))
})
List<StudentModel> selectAllStudents ();
```

2. Penjelasan

Method diubah dengan menambahkan anotasi `@Results` dan `@Results`. Anotasi *Results* digunakan untuk memetakan hasil dari *query select* ke *class model.* Setiap *property* dari *class StudentModel* akan diisi dengan nilai dari kolom-kolom tabel *student* hasil dari *query select.* Khusus untuk *property courses* yang dimiliki oleh *class StudentModel*, hasilnya diambil dari *method selectCourses* yang telah dibuat pada tutorial sebelumnya. Kolom npm hasil dari *query* dijadikan sebagai *input* untuk parameter *method selectCourses*.

3. Tampilan



DINA FEBRIANTI KRISNA
1506757421

Latihan Menambahkan *View* pada Course

1. **Kode**

```
@Select("select id_course, name, credits from course where id_course
= #{id}")
@Results(value = {
      @Result(property="idCourse", column="id_course"),
      @Result(property="name", column="name"),
      @Result(property="credits", column="credits"),
      @Result(property="students", column="id_course",
                  javaType = List.class,
                  many=@Many(select="selectStudents"))
})
CourseModel selectCourse (@Param("id") String id);

@Select("select student.npm, name " +
      "from studentcourse join student " +
      "on studentcourse.npm = student.npm " +
      "where studentcourse.id_course = #{id}")
List<StudentModel> selectStudents (@Param("id") String id);
```

2. **Penjelasan**

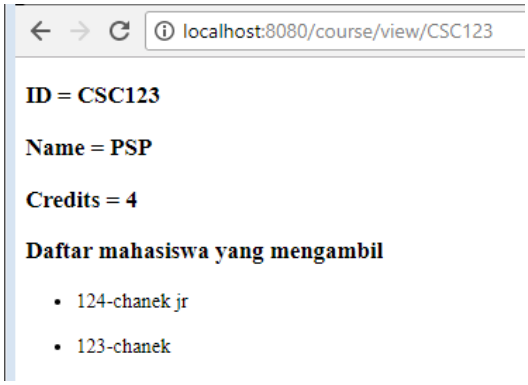
Kedua *method* tersebut prinsipnya sama dengan kode untuk melakukan *view Student* beserta *Courses* yang diambil. Pertama, perlu dibuat *method selectStudents* yang menerima parameter ID dari *course* tertentu. Method ini akan mengembalikan *list of StudentModel*, yang berisi *students* yang mengambil *course* dengan ID sesuai input parameter.

Kemudian perlu ditambahkan method *selectCourse* yang menerima parameter ID dari *course.* Setelah itu dilakukan operasi *select* dari tabel course berdasarkan ID yang diinput. Hasil dari *query* akan dipetakan ke *class CourseModel* melalui anotasi *Results.* Kolom id_course pada hasil *query* dipetakan ke *property* idCourse pada *CourseModel*. Kolom name pada hasil *query* dipetakan ke *property* name pada *CourseModel.* Kolom credits pada hasil *query* dipetakan ke *property* credits pada *CourseModel.* Sedangkan untuk *property students* yang dimiliki oleh *class CourseModel*, hasilnya diambil dari *method selectStudents* yang telah dibuat sebelumnya. Kolom id_course pada hasil *query* dijadikan sebagai *input* untuk parameter *method selectStudents*.

Agar kedua *methods* tersebut dapat berjalan, saya juga menambahkan *classes* dan *interfaces* yang dibutuhkan. *Interfaces* baru yang saya buat diantaranya adalah *CourseMapper* yang berisi kedua *methods* di atas dan *CourseService. Classes* baru yang saya buat diantaranya adalah *CourseController* dan *CourseServiceDatabase* yang *implements interface CourseService*.

Selain itu, saya juga membuat dua halaman html baru untuk menangani tampilan *course*. Kedua halaman tersebut adalah *view-course* dan *not-found-course*. Halaman *view-course* akan menampilkan *course* dengan ID yang di-*input* melalui URL. Halaman *not-found-course* akan menampilkan pesan apabila ID yang di-*input* tidak ditemukan di dalam *database.*

3. **Tampilan**


localhost:8080/course/view/CSC123
ID = CSC123
Name = PSP
Credits = 4
Daftar mahasiswa yang mengambil
124-chanek jr
123-chanek