

Disa Ainun  
1506689654  
APAP A

Yang dipelajari pada tutorial kali ini adalah penggunaan database dan relasi database dalam project spring boot. Di tutorial ini kita mempelajari bagaimana menerima data sesuai dengan apa yang dimau. Terdapat notasi @Result dan @Results, fungsi dari notasi @Result adalah akan mendefine sebuah result yang telah ditentukan, dan @Results akan mendefine banyak hasil yang telah ditentukan oleh masing-masing result. Notasi tersebut dapat dilihat dan dieksekusi langsung pada file mapper yang saya cantumkan.

```
@Mapper
public interface StudentMapper
{
    @Select("select npm, name, gpa from student where npm = #{npm}")
    @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"))
    })
    StudentModel selectStudent (@Param("npm") String npm);
}
```

- Method yang Anda ubah pada Latihan Merubah SelectAllStudents,




Pada langkah pertama saya mengubah method selectAllStudents pada StudentMapper yaitu dengan menambahkan selectAllStudents pada list.

```
@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 ();
```






Selain menggunakan method selectAllStudents untuk mendapatkan nama, npm, gpa. Digunakan juga method yang sebelumnya telah dibuat saat melakukan view student, yaitu method selectCourses untuk mendapatkan output mata kuliah yang diambil oleh siswa tersebut

Disa Ainun  
1506689654  
APAP A

Apabila method tersebut dijalankan akan menghasilkan



localhost:8080/student/viewall

 Apps  Poster GL-02  Kotak Masuk (284) - 

# All Students

**No. 1**  
**NPM = 123**  
**Name = Chanek**  
**GPA = 4.0**  
**Kuliah yang diambil**

- MPKT-6 sks

---

**No. 2**  
**NPM = 124**  
**Name = Chanek Jr**  
**GPA = 3.0**  
**Kuliah yang diambil**

- PSP-4 sks
- SDA-3 sks

---

Disa Ainun  
1506689654  
APAP A

- Method yang Anda buat pada Latihan Menambahkan View pada Course,

Kali ini, saya akan membuat class baru dan interface baru untuk course.

Pada langkah tutorial, telah kita buat course model,

```
1  package com.example.model;
2
3  import java.util.List;
4
5  import lombok.AllArgsConstructor;
6  import lombok.Data;
7  import lombok.NoArgsConstructor;
8
9  @Data
10 @AllArgsConstructor
11 @NoArgsConstructor
12 public class CourseModel {
13     private String id_course;
14     private String name;
15     private int credits;
16     private List<StudentModel> students;
17
18 }
19
```

Course model telah tersedia, tahap berikutnya saya membuat CourseMapper yang berisi

Disa Ainun  
1506689654  
APAP A

```
package com.example.dao;

import java.util.List;
import com.example.model.CourseModel;
import com.example.model.StudentModel;
import org.apache.ibatis.annotations.*;

@Mapper
public interface CourseMapper {

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

    @Select("select id_course, name, credits from course where id_course = #{id_course}")
    @Results(value = {
        @Result(property = "id_course", 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_course") String id_course);
}
```

Method diatas berfungsi untuk mengambil data siapa saja mahasiswa yang mengambil matakuliah tertentu.

Berikutnya saya membuat interface CourseService dan CourseServiceDatabase

```
1 package com.example.service;
2
3
4 import com.example.model.CourseModel;
5
6 public interface CourseService {
7
8     CourseModel selectCourse ((String id_course));
9
10 }
11
```

Disa Ainun  
1506689654  
APAP A

```
1 package com.example.service;
2 import com.example.dao.CourseMapper;
3 import com.example.model.CourseModel;
4 import org.springframework.beans.factory.annotation.Autowired;
5 import org.springframework.stereotype.Service;
6
7
8
9 import lombok.extern.slf4j.Slf4j;
10
11 @Slf4j
12
13 @Service
14 public class CourseServiceDatabase implements CourseService {
15
16     @Autowired
17     private CourseMapper courseMapper;
18
19     @Override
20     public CourseModel selectCourse (String id_course)
21     {
22         log.info ("select course with id_course {}", id_course);
23         return courseMapper.selectCourse (id_course);
24     }
25
26 }
27
28
```

Dilanjutkan dengan membuat CourseController dan view-course.html

Disa Ainun  
1506689654  
APAP A

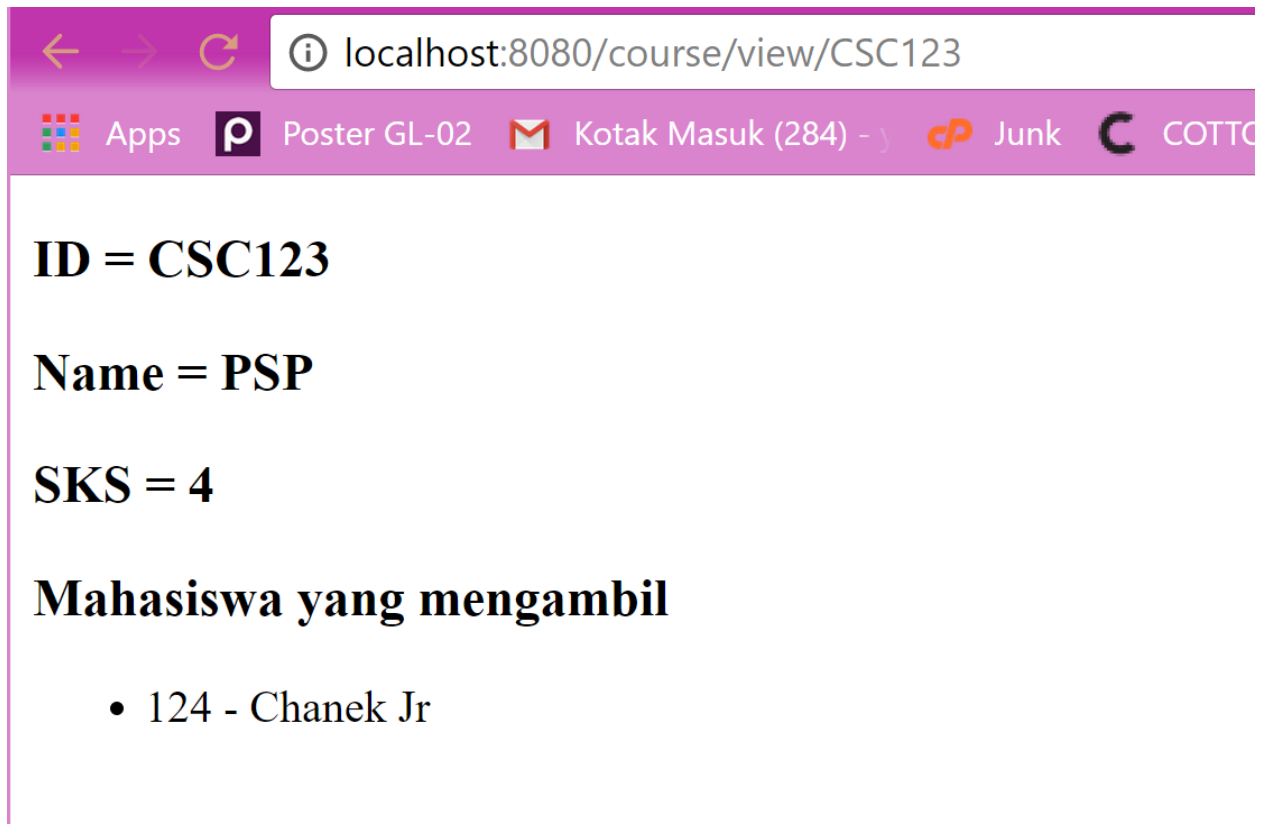
```
CourseController.java ^ StudentModel.java ^ StudentMapper.java ^ CourseServiceDatabase.java ^ StudentSe

1 package com.example.controller;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.beans.factory.annotation.Autowired;
5 import org.springframework.ui.Model;
6 import org.springframework.web.bind.annotation.PathVariable;
7 import org.springframework.web.bind.annotation.RequestMapping;
8
9 import com.example.model.CourseModel;
10 import com.example.service.CourseService;
11
12 @Controller
13 public class CourseController {
14
15     @Autowired
16     CourseService courseDAO;
17
18
19     @RequestMapping("/course/view/{id_course}")
20     public String viewCourse (Model model,
21                             @PathVariable(value = "id_course") String id_course)
22     {
23         CourseModel course = courseDAO.selectCourse(id_course);
24         if (course != null) {
25             model.addAttribute ("course", course);
26             return "view-course";
27         } else {
28             model.addAttribute ("course", course);
29             return "course-not-found";
30         }
31     }
32 }

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8"/>
5     <title>View Course by ID</title>
6 </head>
7 <body>
8     <h3 th:text="'ID = ' + ${course.id_course}">Course ID</h3>
9     <h3 th:text="'Name = ' + ${course.name}">Course Name</h3>
10    <h3 th:text="'SKS = ' + ${course.credits}">Course Credit</h3>
11
12    <h3>Mahasiswa yang mengambil </h3>
13    <ul th:each="student, iterationStatus: ${course.students}">
14        <li th:text="${student.npm} + ' - ' + ${student.name}">
15            NPM - NAMA STUDENT
16        </li>
17    </ul>
18 </body>
19 </html>
```

Apabila method tersebut dijalankan akan menghasilkan

Disa Ainun  
1506689654  
APAP A



The screenshot shows a web browser interface with a purple header bar. The address bar displays "localhost:8080/course/view/CSC123". Below the address bar, there is a navigation bar with several icons and labels: "Apps", "Poster GL-02", "Kotak Masuk (284)", "Junk", and "COTTO". The main content area is white and contains the following text:

**ID = CSC123**

**Name = PSP**

**SKS = 4**

**Mahasiswa yang mengambil**

- 124 - Chanek Jr