



# ICE4016 데이터베이스 설계

## <제목>

Web, Login

### 보고서 작성 서약서

1. 나는 타학생의 보고서를 베끼거나 여러 보고서의 내용을 짜집기하지 않겠습니다.
2. 나는 보고서의 주요 내용을 인터넷사이트 등을 통해 얻지 않겠습니다.
3. 나는 보고서의 내용을 조작하지 않겠습니다.
4. 나는 보고서 작성에 참고한 문헌의 출처를 밝히겠습니다.
5. 나는 나의 보고서를 제출 전에 타학생에게 보여주지 않겠습니다.

나는 보고서 작성시 윤리에 어긋난 행동을 하지 않고 정보통신공학인으로서 나의 명예를 지킬 것을 맹세합니다.

2022년 11 월 8일

학부 정보통신공학

학년 3

성명 김민겸

학번 12201863

## 1. 개요

### ○ STEP 1 : Inha DB에 로그인 기능 구현 적용

- 학생의 학번 및 비밀번호로 로그인 하는 기능 구현할 것
  - 학생은 학생 정보를 조회할 수 있는 페이지로 이동

### ○ STEP 2 : Class 테이블에 데이터 삽입하여 delete 결과 확인

- 학생으로 로그인 한 경우에만 수강하는 수업(class)을 수강취소(삭제) 할 수 있도록 구현

### ○ 보고서에 반드시 포함될 내용 (hwp/doc화일)

- 소스코드 및 보고서는 하나의 폴더에 압축하여 업로드(.zip / .tar / .gz 등)

## 2. 상세 설계 내용

### Step 1. Inha DB에 로그인 기능 구현

#### (1) DB 개선

```
mysql> desc student;
```

Field	Type	Null	Key	Default	Extra
Id	int	NO	PRI	NULL	auto_increment
Name	varchar(10)	NO		NULL	
Email	varchar(45)	YES	UNI	NULL	
Phone_number	varchar(45)	YES	UNI	NULL	
Student_id	int	NO		NULL	
Major	int	NO	PRI	NULL	

6 rows in set (0.01 sec)

현재 학생의 학번 (Student\_id)가 PK로 지정되어있지 않다. Id 필드를 지우고 Student\_id를 PK로 지정하며, Password 필드를 추가해 사용자가 비밀번호를 입력할 수 있도록 한다.

```
mysql> desc student;
```

Field	Type	Null	Key	Default	Extra
Student_id	int	NO	PRI	NULL	
Name	varchar(10)	NO		NULL	
Password	varchar(45)	NO		NULL	
Email	varchar(45)	YES	UNI	NULL	
Phone_number	varchar(45)	YES	UNI	NULL	
Major	int	NO	MUL	NULL	

6 rows in set (0.00 sec)

개선한 DB는 다음과 같다.

## (2) 로그인 기능 구현

- A. index.js에서 loginRouter를 설정해 사용자가 로그인하는 화면으로 접근할 수 있도록 한다.

```
import loginRouter from "../routes/login";
```

```
app.use("/",loginRouter);
```

index.js에 loginRouter를 ./routes/login 폴더에 연결해주고 '/' root URL에 접근하였을 때 이 파일로 이동하도록 하였다.

- B. sql.js에서는 inha DB를 연결한다. login 구현 과정에서 모든 Student의 정보를 불러오기 위해 모든 학생들의 정보를 불러오는 select query를 작성한다.

```
const pool = mysql.createPool(  
  process.env.JAWSDB_URL ?? {  
    host: "localhost",  
    user: "root",  
    database: "inha",  
    password: "f... ..",  
    waitForConnections: true,  
    connectionLimit: 10,  
    queueLimit: 0,  
  });
```

```
export const selectSql = {  
  getStudent: async () => {  
    const [rows] = await promisePool.query(`select * from student`);  
  
    return rows;  
  }  
}
```

로컬 MySQL의 inha DB에 연결해주고, selectSql이라는 이름으로 작성한 배열에 getStudent 함수를 정의한다. getStudent는 모든 학생들의 정보를 가져와 login시 입력한 정보에 해당하는 학생을 찾는 역할을 한다.

- C. login.js에서 아래 두 가지 get, post 함수를 작성해준다. 이때 id는 사용자의 학번(Student\_id)이고 비밀번호는 Password로 지정한다.

```
router.get('/',(req,res) => {  
  res.render('login');  
});
```

get method로 root URL에 접근하면 login.hbs가 출력되도록 설정하였다.

post method로 root URL에 접근하면 아래 함수가 실행된다.

```

router.post('/', async(req, res) => {
  const vars = req.body;
  const students = await selectSql.getStudent();
  let checkLogin = false;

  students.map((student) => {
    console.log(student.id);
    if (vars.id === student.Id && vars.password === student.Password) {
      console.log('login success!');
      checkLogin = true;
    }
  })

  if (checkLogin) {
    res.redirect('/select');
  } else {
    console.log('login failed!');
    res.send("<script>alert('로그인에 실패했습니다.');" + location.href = '/');
  }
})

module.exports = router;

```

selectSql로부터 받아온 rows를 students에 담고 화면에서 입력한 body와 일치하는 지점을 찾아 일치하는 Id와 password가 동시에 동일하면 checkLogin을 true로 변경한다.

탐색이 종료된 이후 checkLogin이 true이면 select로 redirect하여 학생 정보를 조회하는 페이지로 이동한다. false인 경우에는 login failed 로그를 찍고 alert창이 뜨게 한다.

### (3) 학생 정보 조회 페이지로 이동

A. 로그인에 성공한 경우 'select' 페이지로 이동하기 위해 먼저 index.js에 selectRouter를 설정

```

import selectRouter from "./routes/select";

app.use("/select", selectRouter);

```

select 구문을 통해 selectRouter과 '/select' URL을 연결해주었다.

B. login.js에 로그인에 성공한 경우에 '/select'로 이동하도록 설정

```

if (checkLogin) {
  res.redirect('/select');
} else {

```

C. select.js에 학생 정보 조회 페이지로 이동시킴

```
router.get('/', async function(req,res){
  const student = await selectSql.getStudent();

  res.render('select',{
    title: '학생 정보',
    student
  });
});
```

select.hbs에서 학생 정보를 출력할 수 있도록 파라미터를 넘겨준다.

D. select.hbs 작성하기

```
<h1>{{ title }}</h1>
<table>
<tr>
<td>Student_Id</td>
<td>Name</td>
<td>Email</td>
<td>Phone_number</td>
<td>Major</td>
</tr>
{{#each student}}
<tr>
<td>{{Student_Id}}</td>
<td>{{Name}}</td>
<td>{{Email}}</td>
<td>{{Phone_number}}</td>
<td>{{Major}}</td>
</tr>
{{/each}}
</table>
```

select.hbs는 위와 같이 작성해준다.

## Step2. Class 테이블에 데이터 삽입하여 delete 결과 확인

학생으로 로그인한 경우에만 수강하는 수업을 수강취소 할 수 있다.

DB 개선 없이 바로 프로그램 구현 과정으로 들어간다.

(1) index.js에서 deleteRouter 추가하기

```
import deleteRouter from "../routes/delete";
```

```
app.use("/select",selectRouter);
app.use("/delete",deleteRouter);
```

## (2) session 정보 추가하기

express-session 라이브러리를 사용해서 login시 login userId를 저장하고 delete 화면에서 사용하려고 한다.

➤ npm install express-session

명령어를 이용해서 라이브러리를 설치하고 아래와 같이 index.js에 입력해준다.

```
/* 세션 설정 */
const session = require('express-session');
const MemoryStore = require('memorystore')(session);

const maxAge = 1000 * 60 * 5
const sessionObj = {
  secret: 'asdf1234',
  resave: false,
  saveUninitialized: true,
  store : new MemoryStore({checkPeriod: maxAge}),
  cookie: {
    maxAge,
  },
};

app.use(session(sessionObj));
```

그리고 login.js에서 로그인 시 userId를 세션에 저장한다.

```
if(checkLogin) { // login succeed
  req.session.userid = vars.id;
  console.log(vars.id);
  console.log(req.session);
  res.redirect(['/select']);
}
```

콘솔을 찍어보면 아래와 같이 세션에 userId가 잘 기록되는 것을 볼 수 있다.

```
GET /select 304 65.904 ms - -
Session {
  cookie: {
    path: '/',
    _expires: 2022-11-09T06:35:19.479Z,
    originalMaxAge: 299999,
    httpOnly: true
  },
  userid: '12201863'
```

(3) delete.js에서 deleteSql과 연결해서 수업을 취소하는 get, post 기능을 제작한다.

get 메서드에서는 delete.hbs로 연결할 수 있도록 하고 로그인한 사용자에게 대한 class 정보를 가져와서 현재 수강 중인 수업의 리스트를 출력한다.

```
router.get('/', async (req, res) => {
  console.log(req.session);
  const classes = await selectSql.getClasses(req.session.userid);
  res.render('delete', {
    title: "수강 취소 기능",
    classes
  });
});
```

그림 1 delete.js

delete.js에서는 login.js에서 로그인한 값이 필요하다. 사용자의 아이디를 통해서 사용자가 수강하는 수업의 목록을 추가해야하기 때문이다. getClass 함수는 아래와 같이 class와 class\_has\_student를 조인해서 현재 로그인한 사용자의 userid를 기준으로 select하도록 하였다. delete.js에서는 req.session.userid를 입력해 로그인 시 저장한 userid 정보를 넘겨주었다.

```
getClasses : async (userid) => { // 현재 사용자에게 대한 수강 강의 목록을 가져오는 함수
  const [rows] = await promisePool.query(`select c.id as Id, Name, Number_Of_Participants, Professor
    from class as c, class_has_student as chs
    where chs.student_id = ${userid} and c.id=chs.class_id`);
  return rows;
}
```

class 테이블과 class\_has\_student 테이블의 id가 겹치는 문제가 있어 위와 같이 수정하였다.

그림 2 sql.js

```
button type="button" onclick="location.href='/delete'">수강취소하러 가기</button>
```

그림 3 select.hbs

select.hbs에 위의 구문을 추가해서 session 정보가 손실되지 않고 수강취소 화면으로 넘어가도록 하였다.

```
</tr>
  {{#each classes}}
  <form method="post">
  <tr>
    <td>{{Id}}</td>
    <td>{{Name}}</td>
    <td>{{Number_Of_Participants}}</td>
    <td>{{Professor}}</td>
    <td style="border: none; margin-left: 10px;">
    <button
      style="margin-left: 10px;"
      name='delBtn'
      type="submit"
      value="{{Id}}"
      formaction="/delete">삭제</button>
    </td>
  </tr>
```

delete.hbs를 위와 같이 수정하였다. 여기서 value를 기준으로 id를 가져와 js에서 사용할 수 있고 이때 req.body.delBtn으로 접근하면 해당 class의 id를 가져올 수 있다.

(4) sql.js에서 delete query를 작성한다. 이때 현재 학생의 정보에서 선택한 강좌를 취소하는 query문을 설계한다.

```
router.post('/', async (req, res) => {
  const data = {
    classId: req.body.id,
    studentId: req.session.userid,
  };

  await deletesql.deleteClass(data);

  res.redirect('/delete');
})
```

그림 4 delete.js

delete.js에서 위와 같이 post메서드에 classId와 studentId를 data에 담아 sql.js에 전송하였다.

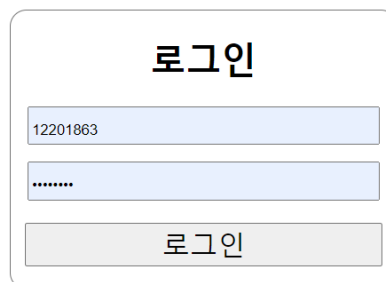
```
export const deletesql = {
  deleteClass: async (data) => {
    const sql = `delete from class_has_student where class_id="${data.classId}" and Student_id="${data.studentId}"`;

    await promisePool.query(sql);
  }
}
```

그림 5 sql.js

sql.js에서 delete문을 위와 같이 작성하여 수강 정보를 삭제해주었다.

### (3) 실행 화면



The image shows a login form titled '로그인' (Login). It contains two input fields: the first for a user ID, which has the value '12201863' entered, and the second for a password, which is masked with '\*\*\*\*\*'. Below these fields is a button labeled '로그인' (Login).

그림 6 로그인 화면



## 학생 정보

Student_Id	Name	Email	Phone_number	Major
12200000	이슬	angel@naver.com	010-1234-1234	3
12201234	김다영	ekdud@naver.com	010-0000-0000	2
12201235	조한나	whgkssk@naver.com	010-1234-0101	4
12201236	원민재	jaejae@naver.com	010-1010-1010	5
12201863	김민겸	mingyum119@naver.com	010-9479-0373	1

[수강취소하러 가기](#)

그림 7 로그인 성공 시 redirect되는 학생 정보 테이블 ('/select')

## 수강 취소 기능

Id	Name	Number of Participants	Professor	
ICE4008	컴퓨터 네트워크	50	0	<a href="#">삭제</a>
ICE4010	이동통신	45	0	<a href="#">삭제</a>
ICE4020	정보 보호론	30	0	<a href="#">삭제</a>

그림 8 현재 사용자의 수강 정보 ('/delete')

## 수강 취소 기능

Id	Name	Number of Participants	Professor	
ICE4010	이동통신	45	0	<a href="#">삭제</a>
ICE4020	정보 보호론	30	0	<a href="#">삭제</a>

그림 9 delete 요청을 보낸 후 현재 사용자의 수강 정보 ('/delete')