데이터베이스설계 (ICE4016)

실습 7주차

MySQL Express 연동

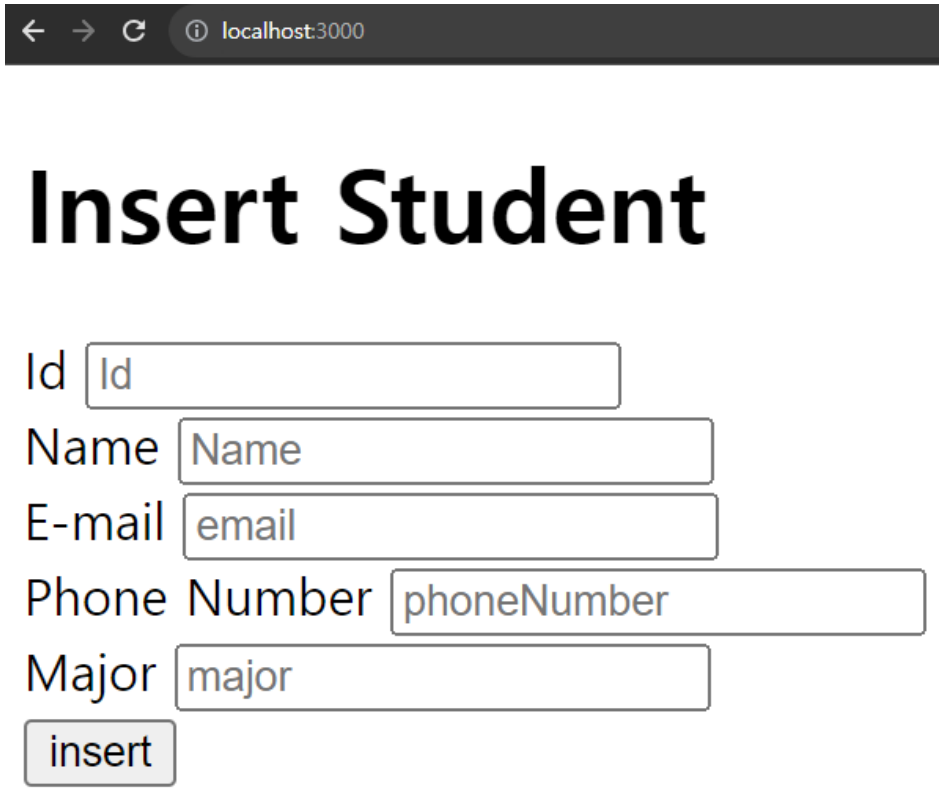
Prof. Wonik Choi

데이터 인텔리전스 연구실
Data Intelligence Laboratory

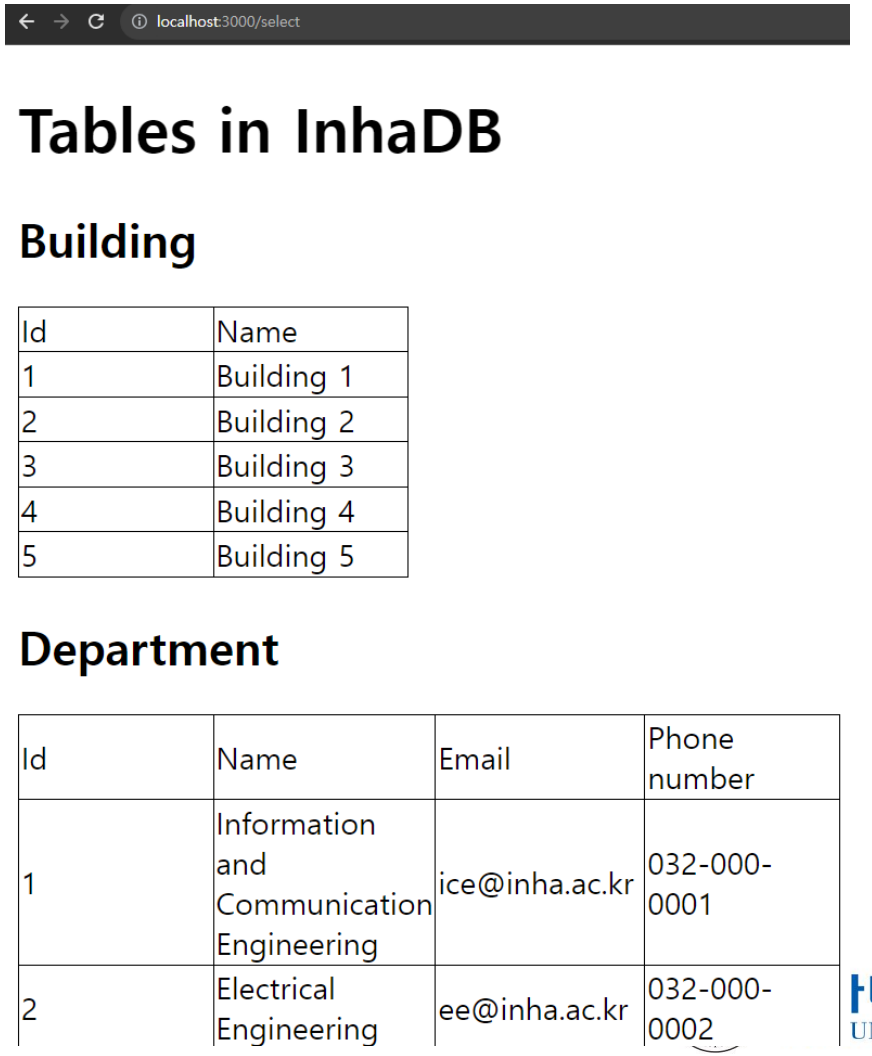
인하대학교
INHA UNIVERSITY

Week 7 practice goal

localhost:3000/
(데이터 삽입 페이지)



localhost:3000/select
(데이터 조회 페이지)



Week 7 practice goal

localhost:3000/update/student
(학생 데이터 수정 페이지)

UPDATE 'Student' table

Id	Name	Email	Phone_number	Major	Student_Id
1	LEE Sangwon	leesw9501@inha.ac.kr	010-0000-0001	Electrical and Computer En	Modify
2	Hong Gil Dong	hgd@inha.ac.kr	010-0000-0002	Electrical Engineering	Modify
3	Hansol	hsl@inha.ac.kr	010-0000-0003	Electrical Engineering, Com	Modify
4	test	undefined	010-0000-0004	undefined	Modify
5	test2	test2	010-0000-0005	test2	Modify

Week 7 practice

- 프로젝트 생성 및 환경 세팅

https://github.com/leesw9501/Database/tree/main/week6

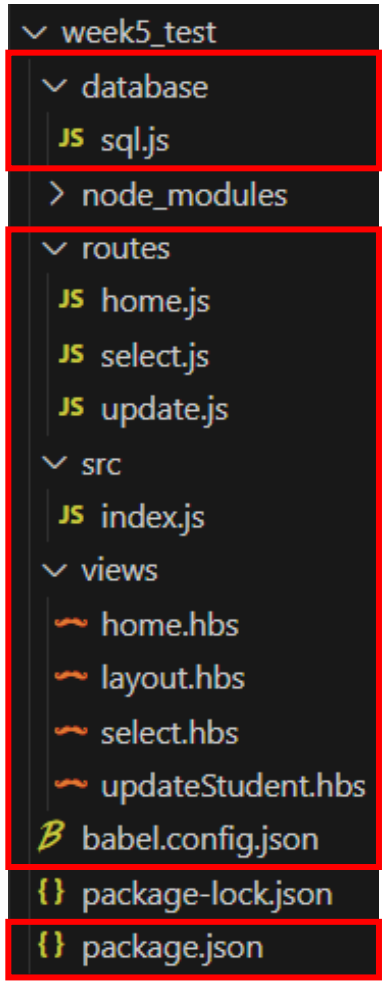
  - 환경 세팅을 위한 모듈 설치
    - npm init
    - npm install express mysql2 body-parser nodemon morgan dotenv
    - npm install @babel/node @babel/core @babel/preset-env
    - npm link hbs

```
PS C:\Users\leesw\OneDrive\바탕 화면\db\Database\week5_test> npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (week5_test)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to C:\Users\leesw\OneDrive\바탕 화면\db\Database\week5_test\package.json:

{
  "name": "week5_test",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}


Is this OK? (yes) yes
```

```
PS C:\Users\leesw\OneDrive\바탕 화면\db\Database\week5_test> npm install express mysql2 body-parser nodemon morgan dotenv

added 110 packages, and audited 111 packages in 4s

12 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\leesw\OneDrive\바탕 화면\db\Database\week5_test> npm install @babel/node @babel/core @babel/preset-env

added 247 packages, and audited 358 packages in 25s

64 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\leesw\OneDrive\바탕 화면\db\Database\week5_test> npm link hbs

added 1 package, and audited 360 packages in 1s

64 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Week 7 practice

○ 프로젝트 생성 및 환경 세팅



Database > week5 > babel.config.json > ...

```
{
    "presets": ["@babel/preset-env"]
}
```

Database > week5 > package.json > ...

```
{
  "name": "week5",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  Debug
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon --exec babel-node ./src/index.js"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "@babel/core": "^7.22.20",
    "@babel/node": "^7.22.19",
    "@babel/preset-env": "^7.22.20",
    "body-parser": "^1.20.2",
    "dotenv": "^16.3.1",
    "express": "^4.18.2",
    "morgan": "^1.10.0",
    "mysql2": "^3.6.1",
    "nodemon": "^3.0.1"
  }
}
```

src/index.js

Database > week5_test > src > JS index.js > ...

```
1   import express from 'express';
2   import logger from 'morgan';
3   import path from 'path';
4
5   import homeRouter from '../routes/home';
6   import updateRouter from '../routes/update';
7   import selectRouter from '../routes/select';
8
9   const PORT = 3000;
10
11  const app = express();
12
13  app.use(express.static(path.join(__dirname, '/src')));
14  app.use(express.urlencoded({ extended: false }))
15  app.use(express.json());
16
17  app.set('views', path.join(__dirname, '../views'));
18  app.set('view engine', 'hbs');
19
20  app.use(logger('dev'));
21
22  app.use('/', homeRouter);
23  app.use('/update', updateRouter);
24  app.use('/select', selectRouter);
25
26  app.listen(PORT, () => {
27      console.log(`Server is running at http://localhost:${PORT}`)
28  });
```

routes/home.js & views/home.hbs

Database > week5_test > routes > JS home.js > ...

```js
import express from 'express';
import { insertSql } from '../database/sql';

const router = express.Router();

router.get('/', (req, res) => {
    res.render('home', { data: " " });
})

router.post('/', (req, res) => {
    const vars = req.body;

    const data = {
        Id: vars.id,
        Name: vars.name,
        Email: vars.email,
        PhoneNumber: vars.phoneNumber,
        Major: vars.major,
    };
    insertSql.setStudent(data);
})

module.exports = router;
```

Database > week5_test > views > home.hbs > ...

```hbs
<h1>Insert Student</h1>
<form name="student" method="post" action="/">
    <div>
        <label for="id">Id</label>
        <input id="id" name="id" type="text" required placeholder="Id" />
    </div>
    <div>
        <label for="name">Name</label>
        <input id="name" name="name" type="text" required placeholder="Name" />
    </div>
    <div>
        <label for="email">E-mail</label>
        <input id="email" name="email" type="text" placeholder="email" />
    </div>
    <div>
        <label for="phoneNumber">Phone Number</label>
        <input id="phoneNumber" name="phoneNumber" type="text"
        required placeholder="000-0000-0000" />
    </div>
    <div>
        <label for="major">Major</label>
        <input id="major" name="major" type="text" required placeholder="major" />
    </div>
    <div>
        <input type="submit" value="insert" />
    </div>
</form>
```

routes/select.js & views/select.hbs

Database > week5_test > routes > JS select.js > ...

```
import express from 'express';
import { selectSql } from '../database/sql';

const router = express.Router();

router.get('/', async (req, res) => {
    const Building = await selectSql.getBuilding();
    const Department = await selectSql.getDepartment();
    const Room = await selectSql.getRoom();
    const Student = await selectSql.getStudent();

    res.render('select', {
        main_title: "Tables in InhaDB",
        title1: "Building",
        title2: "Department",
        title3: "Room",
        title4: "Student",
        Building,
        Department,
        Room,
        Student,
    })
})

module.exports = router;
```

Database > week5_test > views > select.hbs > ...

```
<h1>{{main_title}}</h1>

<!--Building-->
<h2>{{title1}}</h2>

<table>
    <tr>
        <td>Id</td>
        <td>Name</td>
    </tr>
    {{#each Building}}
    <tr>
        <td>{{Id}}</td>
        <td>{{Name}}</td>
    </tr>
    {{/each}}
</table>

<!--Department-->
<h2>{{title2}}</h2>

<table>
    <tr>
        <td>Id</td>
        <td>Name</td>
        <td>Email</td>
        <td>Phone number</td>
    </tr>
    {{#each Department}}
    <tr>
        <td>{{Id}}</td>
        <td>{{Name}}</td>
        <td>{{Email}}</td>
        <td>{{PhoneNumber}}</td>
    </tr>
    {{/each}}
</table>
```

Database > week5_test > views > select.hbs > ...

```
<!--Room-->
<h2>{{title3}}</h2>

<table>
    <tr>
        <td>Id</td>
        <td>Name</td>
        <td>Capacity</td>
        <td>Building Id</td>
        <td>Department Id</td>
    </tr>
    {{#each Room}}
    <tr>
        <td>{{id}}</td>
        <td>{{Name}}</td>
        <td>{{Capacity}}</td>
        <td>{{building_Id}}</td>
        <td>{{Department_Id}}</td>
    </tr>
    {{/each}}
</table>

<!--Student-->
<h2>{{title4}}</h2>

<table>
    <tr>
        <td>Id</td>
        <td>Name</td>
        <td>Email</td>
        <td>Phone number</td>
        <td>Major</td>
    </tr>
    {{#each Student}}
    <tr>
        <td>{{Id}}</td>
        <td>{{Name}}</td>
        <td>{{Email}}</td>
        <td>{{PhoneNumber}}</td>
        <td>{{Major}}</td>
    </tr>
    {{/each}}
</table>
```

routes/update.js & views/ updateStudent.hbs

Database > week5_test > routes > JS update.js > ...

```js
import express from 'express';
import { selectSql, updateSql } from '../database/sql';

const router = express.Router();

router.get('/student', async (_req, res) => {
    const student_res = await selectSql.getStudent();
    console.log(student_res)
    res.render('updateStudent', {
        main_title: "UPDATE 'Student' table",
        student_res,
    });
});
router.post('/student', async (req, res) => {
    const vars = req.body;
    const data = {
        Id: vars.id,
        Name: vars.name,
        Email: vars.email,
        PhoneNumber: vars.phonenumber,
        Major: vars.major,
    }
    await updateSql.updateStudent(data);

    res.redirect('/update');
})

module.exports = router;
```

Database > week5_test > views > updateStudent.hbs > ...

```hbs
<h1>{{main_title}}</h1>
<table>
    <tr>
        <td>Id</td>
        <td>Name</td>
        <td>Email</td>
        <td>Phone_number</td>
        <td>Major</td>
        <td>Student_Id</td>
    </tr>
    {{#each student_res}}
    <form method="post">
        <tr>
            <td>{{Id}}</td>
            <td style="display: none"><input type="text" name="id" value={{Id}}></td>
            <td><input type="text" name="name" value="{{Name}}"></td>
            <td><input type="text" name="email" value={{Email}}></td>
            <td><input type="text" name="phonenumber" value={{PhoneNumber}}></td>
            <td><input type="text" name="major" value="{{Major}}"></td>
            <td><input type="submit" value="Modify" formaction="/update/student"></td>
        </tr>
    </form>
    {{/each}}
</table>
```

database/sql.js

Database > week5_test > database > JS sql.js > ...

```
1   import mysql from 'mysql2';
2
3   require("dotenv").config();
4
5   const pool = mysql.createPool({
6       host: 'localhost',
7       port: 3306,
8       user: 'root',
9       password: '',
10      database: 'inha_week5',
11  });
12
13  const promisePool = pool.promise();
14
15  // select query
16  export const selectSql = {
17      getBuilding: async () => {
18          const sql = `select * from building`;
19          const [result] = await promisePool.query(sql);
20          return result;
21      },
22      getDepartment: async () => {
23          const sql = `select * from department`;
24          const [result] = await promisePool.query(sql);
25          return result;
26      },
27      getRoom: async () => {
28          const sql = `select * from room`;
29          const [result] = await promisePool.query(sql);
30          return result;
31      },
32      getStudent: async () => {
33          const sql = `select * from Student`;
34          const [result] = await promisePool.query(sql);
35          return result;
36      },
37  }
```

Database > week5_test > database > JS sql.js > ...

```
39  // insert query
40  export const insertSql = {
41      setStudent: async (data) => {
42          const sql = `insert into student values (
43              "${data.Id}", "${data.Name}", "${data.Email}",
44              "${data.PhoneNumber}", "${data.Major}"
45          )`
46          console.log(data);
47          await promisePool.query(sql);
48      },
49  };
50
51  // update query
52  export const updateSql = {
53      updateStudent: async (data) => {
54          console.log(data);
55          const sql = `
56              UPDATE Student
57              SET Id = ${data.Id}, Name = "${data.Name}",
58                  Email = "${data.Email}", PhoneNumber = "${data.PhoneNumber}",
59                  Major = "${data.Major}"
60              WHERE Id = ${data.Id}`;
61          console.log(sql);
62          await promisePool.query(sql);
63      },
64  };
```

views/layout.hbs

Database > week5_test > views > layout.hbs > ...

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <title>{{title}}</title>
6       <style type="text/css">
7           table {
8               border-collapse: collapse;
9           }
10
11          th, td {
12              border: 1px solid black;
13              width: 100px;
14              height: 20px;
15          }
16      </style>
17  </head>
18  <body>
19      {{{body}}}
20  </body>
21  </html>
```

Week 7 Assignment

- STEP 1 : localhost:3000/select 페이지에 class table 추가
- STEP 2 : localhost:3000/update/department 페이지 만들기
- 두 개 스탭을 진행하고, 보고서 작성
  - 기능 동작을 확인할 수 있는 사진들을 포함 할 것
  - 코드 제출 필요 없음