

데이터베이스 및 데이터시각화

Project 1 - Web Tutorial

Reference

<http://devlecture.tistory.com/>

Kwangwoon Univ.
Dept. of Computer Engineering
Ki-Hoon Lee



Contents

1. Node.js 설치 및 예제 실행
2. Express 설치 및 예제 실행
3. Ejs
4. 회원가입 폼 만들기 및 nodemon 사용
5. Database(MySQL) 연동
6. 게시판 구축



1. node.js 설치 및 예제 실행

- nodejs.org/ko/ 접속 후 node.js 설치



Node.js®는 Chrome V8 JavaScript 엔진으로 빌드된 JavaScript 런타임입니다.

다운로드 - Windows (x64)

LTS 버전 설치

16.17.0 LTS

안정적, 신뢰도 높음

18.8.0 현재 버전

최신 기능

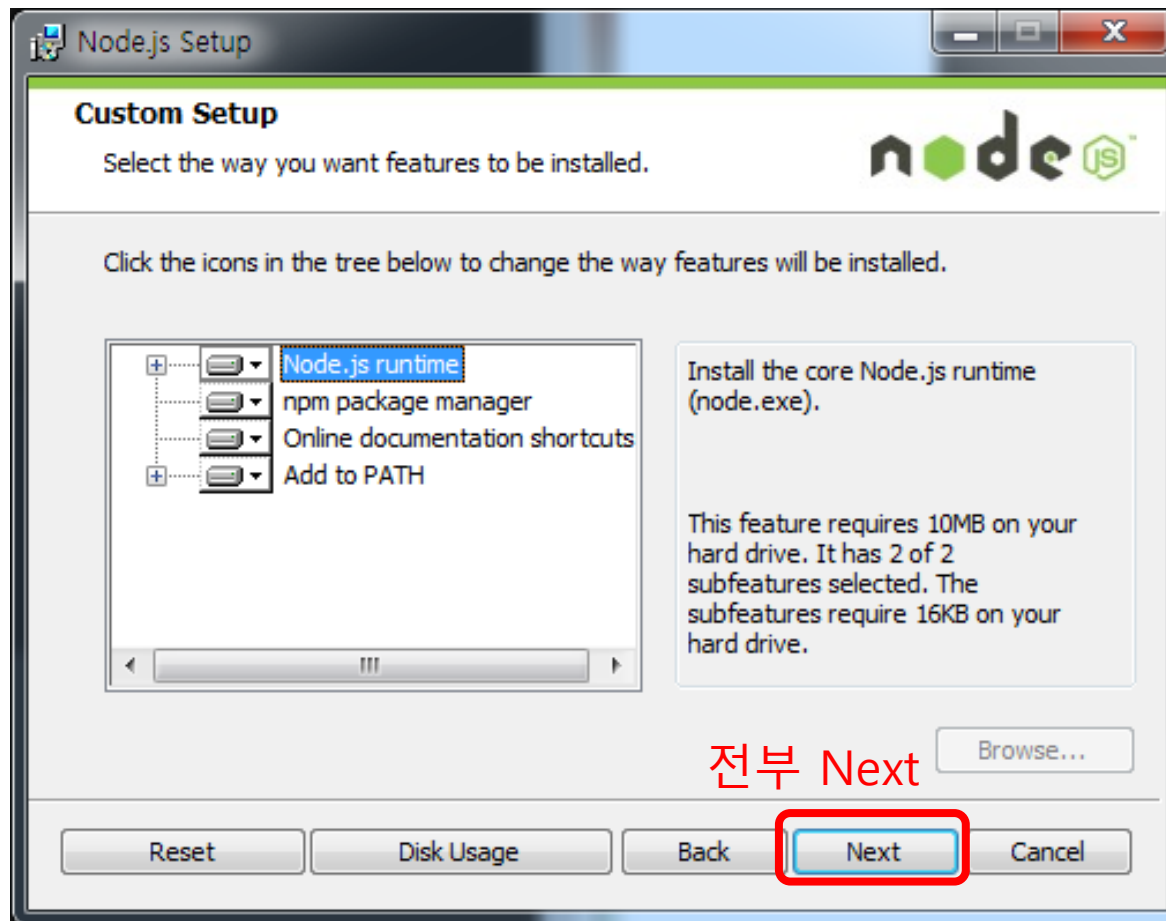
[다른 운영 체제](#) | [변경사항](#) | [API 문서](#)

[다른 운영 체제](#) | [변경사항](#) | [API 문서](#)

LTS 일정은 [여기서](#) 확인하세요

1. node.js 설치 및 예제 실행

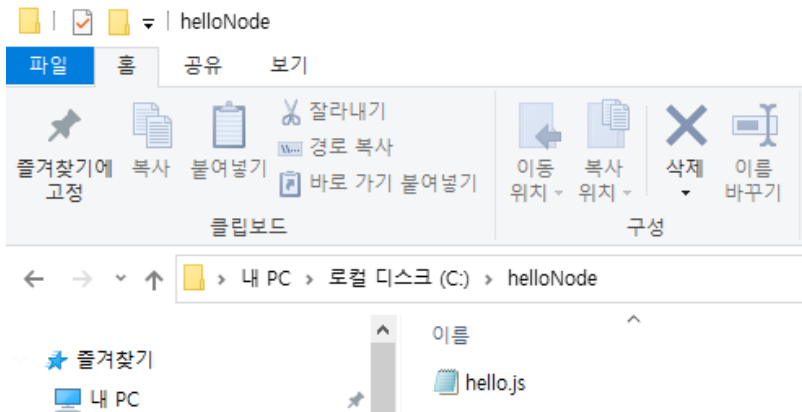
■ 설치파일 실행





1. node.js 설치 및 예제 실행

- helloNode 폴더 생성 후 폴더 안에 hello.js 파일 생성



- 예제소스 작성

```
hello.js
1  var http = require('http');
2  http.createServer(function (req, res){
3      res.writeHead(200, {'Content-Type': 'text/plain'});
4      res.end('Hello World!\n');
5  }).listen(1337, '127.0.0.1');
6  console.log('Server running at http://127.0.0.1:1337/');
```



1. node.js 설치 및 예제 실행

- cmd > node.js 파일이 있는 폴더로 이동(cd 명령어 이용)
=>node hello.js 입력

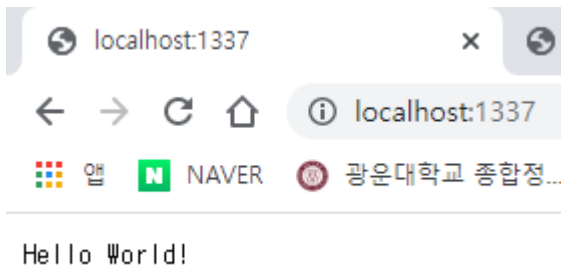
```
cmd 명령 프롬프트 - node hello.js
Microsoft Windows [Version 10.0.19041.867]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\micky>cd /

C:\>cd helloNode

C:\helloNode>node hello.js
Server running at http://127.0.0.1:1337/
```

- 웹 브라우저를 통해 확인



localhost는 Loopback host name으로
IPv4에서 127.0.0.1로 변환



1. node.js 설치 및 예제 실행

- Ctrl + C를 통해 종료

명령 프롬프트

```
C:\helloNode>node hello.js
Server running at http://127.0.0.1:1337/
^C
C:\helloNode>
```

- 새 폴더 생성 후 이동

선택 명령 프롬프트

```
C:\helloNode>cd ..
C:\>mkdir myNode
C:\>cd myNode
C:\myNode>
```



2. Express 설치 및 예제 실행

- > npm install -g express-generator 입력

```
C:\> 명령 프롬프트
C:\myNode>npm install -g express-generator
npm WARN deprecated mkdirp@0.5.1: Legacy versions of mkdirp are no longer
that the API surface has changed to use Promises in 1.x.)
C:\Users\micky\AppData\Roaming\npm\express -> C:\Users\micky\AppData\Roaming\npm\express-cli.js
+ express-generator@4.16.1
updated 2 packages in 0.806s

New major version of npm available! 6.14.12 -> 7.9.0
Changelog: https://github.com/npm/cli/releases/tag/v7.9.0
Run npm install -g npm to update!

C:\myNode>
```




2. Express 설치 및 예제 실행

■ > express example1 입력

```
cmd 명령 프롬프트
C:\myNode>express example1

warning: the default view engine will not be jade in future releases
warning: use '--view=jade' or '--help' for additional options

create : example1\
create : example1\public\
create : example1\public\javascripts\
create : example1\public\images\
create : example1\public\stylesheets\
create : example1\public\stylesheets\style.css
create : example1\routes\
create : example1\routes\index.js
create : example1\routes\users.js
create : example1\views\
create : example1\views\error.jade
create : example1\views\index.jade
create : example1\views\layout.jade
create : example1\app.js
create : example1\package.json
create : example1\bin\
create : example1\bin\www

change directory:
> cd example1

install dependencies:
> npm install

run the app:
> SET DEBUG=example1:* & npm start

C:\myNode>
```



2. Express 설치 및 예제 실행

- >cd example1 # example1로 이동
- >npm install # 모듈 설치

선택 명령 프롬프트

```
C:\myNode\example1>npm install
npm WARN deprecated jade@1.11.0: Jade has been renamed to pug, please install the latest version
npm WARN deprecated constantinople@3.0.2: Please update to at least constantinople 3.1.1
npm WARN deprecated transformers@2.1.0: Deprecated, use jstransformer
npm notice created a lockfile as package-lock.json. You should commit this file.
added 100 packages from 139 contributors and audited 101 packages in 3.436s
found 4 vulnerabilities (3 low, 1 critical)
  run `npm audit fix` to fix them, or `npm audit` for details
```

- >npm start 혹은 >node .\bin\www 실행

npm

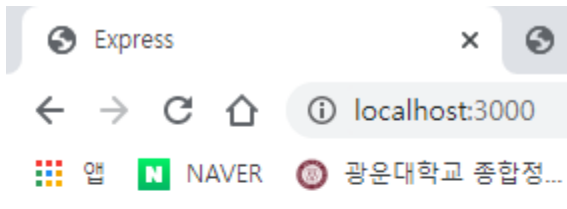
```
C:\myNode\example1>npm start

> example1@0.0.0 start C:\myNode\example1
> node ./bin/www
```



2. Express 설치 및 예제 실행

- 웹 브라우저를 통해 확인



Express

Welcome to Express



2. Express 설치 및 예제 실행

- app.js에서 url을 매핑하는 부분 확인

```
7  var indexRouter = require('./routes/index');  
8  var usersRouter = require('./routes/users');
```

```
22 app.use('/', indexRouter);  
23 app.use('/users', usersRouter);
```



3. ejs

- Embedded JavaScript(ejs) : html에 자바스크립트가 내장된 파일

ejs

```
var express = require('express');
var app = express();

app.set('view engine','ejs');
app.set('views','./views');

app.get('/',function(req,res){
    res.render('view');
});

app.listen(3000,function(){
    console.log('hello ejs');
});
```

<js 코드>

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>document.write()</title>
  </head>
  <body>
    <h2>document.write()의 예제</h2>
    <% for (var i=0; i<5; i++) { %>
      예제 실행 <%= i+1 %>번째.<br>
    <% } %>
  </body>
</html>
```

<ejs 코드>

html

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>document.write()</title>
  </head>
  <body>
    <h2>document.write()의 예제</h2>
    <script language="javascript">
      for (var i = 0; i < 5; i++) {
        document.write("예제 실행" + (i+1) + "번째<br>");
      }
    </script>
  </body>
</html>
```



4. 회원가입 폼 만들기 및 nodemon 사용

- 아이디, 비밀번호, 이름, 이메일, 전화번호, 주소, 직업, 성별, 생일을 입력받음
 - 모든 값에 공백 검사
 - 가입 버튼을 클릭 시 입력한 내용을 json 객체 형태로 표시
- 순서
 - 1) express 프로젝트 생성(ejs 템플릿엔진으로 생성)
 - 2) routes 폴더에 로직 담당하는 js 파일 생성
 - 3) app.js에 소스를 적어 2에서 생성한 js파일 연결
 - 4) views 폴더에 ejs 파일 생성
 - 5) 2에서 만든 js파일에 소스를 적어 4번에서 만든 ejs 파일 연결



4. 회원가입 폼 만들기 및 nodemon 사용

- > express --ejs joinForm # 프로젝트 생성

명령 프롬프트

```
C:\myNode\example1>cd ..  
C:\myNode>express --ejs joinForm  
  
warning: option '--ejs' has been renamed to '--view=ejs'  
  
create : joinForm\public\javascripts  
create : joinForm\public\images  
create : joinForm\public\stylesheets  
create : joinForm\public\stylesheets\style.css  
create : joinForm\routes  
create : joinForm\routes\index.js  
create : joinForm\routes\users.js  
create : joinForm\views  
create : joinForm\views\error.ejs  
create : joinForm\views\index.ejs  
create : joinForm\app.js  
create : joinForm\package.json  
create : joinForm\bin  
create : joinForm\bin\www  
  
change directory:  
  > cd joinForm  
  
install dependencies:  
  > npm install  
  
run the app:  
  > SET DEBUG=joinform:* & npm start  
  
C:\myNode>
```



4. 회원가입 폼 만들기 및 nodemon 사용

- >cd joinForm # joinForm 폴더로 이동
- >npm install # 모듈 설치
- >npm install --save multer # multer 설치

명령 프롬프트

```
C:\myNode>cd joinForm
```

```
C:\myNode\joinForm>npm install
```

```
npm notice created a lockfile as package-lock.json. You should commit this file.  
added 54 packages from 38 contributors and audited 55 packages in 1.407s  
found 0 vulnerabilities
```

```
C:\myNode\joinForm>
```

```
C:\myNode\joinForm>npm install --save multer
```

```
added 11 packages, and audited 75 packages in 3s
```




4. 회원가입 폼 만들기 및 nodemon 사용

- routes 폴더에 joinForm.js 파일 생성
- 같은 폴더에 생성된 users.js에 res.send 부분만 변경

```
joinForm.js x
1 var express = require('express');
2 var router = express.Router();
3
4 /* GET users listing. */
5 router.get('/', function(req, res, next){
6   res.send('joinForm.js check');
7 });
8
9 module.exports = router;
```



4. 회원가입 폼 만들기 및 nodemon 사용

- app.js에 소스를 추가하여 joinForm.js 파일 연결

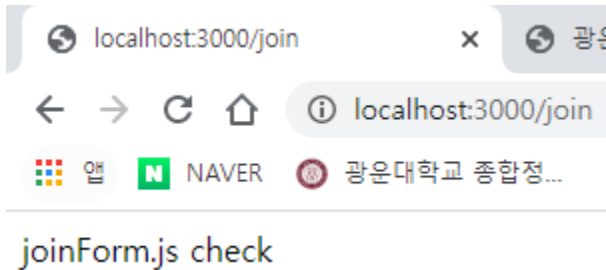
```
joinForm.js x app.js x
1  var createError = require('http-errors');
2  var express = require('express');
3  var path = require('path');
4  var cookieParser = require('cookie-parser');
5  var logger = require('morgan');
6
7  var indexRouter = require('./routes/index');
8  var usersRouter = require('./routes/users');
9  var join = require('./routes/joinForm');
10
11 var app = express();
12
13 // view engine setup
14 app.set('views', path.join(__dirname, 'views'));
15 app.set('view engine', 'ejs');
16
17 app.use(logger('dev'));
18 app.use(express.json());
19 app.use(express.urlencoded({ extended: false }));
20 app.use(cookieParser());
21 app.use(express.static(path.join(__dirname, 'public')));
22
23 app.use('/', indexRouter);
24 app.use('/users', usersRouter);
25 app.use('/join', join);
```



4. 회원가입 폼 만들기 및 nodemon 사용

- 서버를 시작하여 웹 브라우저를 통해 동작 확인

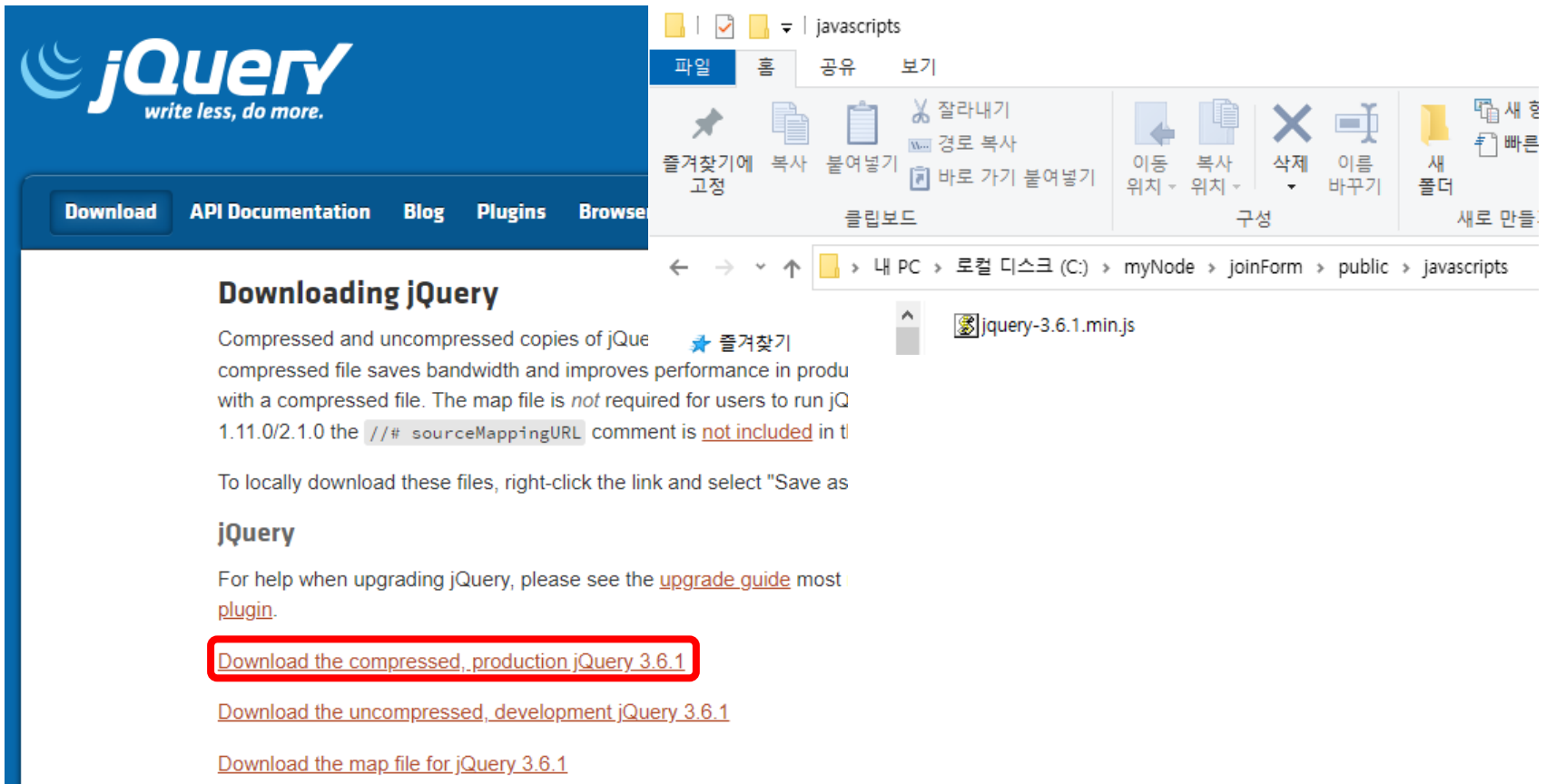
```
C:\> npm
C:\myNode\joinForm> npm start
> joinform@0.0.0 start C:\myNode\joinForm
> node ./bin/www
```





4. 회원가입 폼 만들기 및 nodemon 사용

- jquery.com/download/ jQuery 다운로드
- public\javascripts 폴더에 저장



The screenshot shows the jQuery download page on the left and a Windows File Explorer window on the right. The File Explorer window is open to the 'public\javascripts' folder, showing the file 'jquery-3.6.1.min.js' which has just been downloaded.

jQuery Download Page:

- jQuery logo: write less, do more.
- Buttons: Download, API Documentation, Blog, Plugins, Browse
- Section: **Downloading jQuery**
- Text: Compressed and uncompressed copies of jQuery are available. The compressed file saves bandwidth and improves performance in production with a compressed file. The map file is *not* required for users to run jQuery 1.11.0/2.1.0 the `//# sourceMappingURL` comment is **not included** in the compressed file.
- Text: To locally download these files, right-click the link and select "Save as".
- Section: **jQuery**
- Text: For help when upgrading jQuery, please see the [upgrade guide](#) most [plugin](#).
- Link: [Download the compressed, production jQuery 3.6.1](#) (highlighted with a red box)
- Link: [Download the uncompressed, development jQuery 3.6.1](#)
- Link: [Download the map file for jQuery 3.6.1](#)

File Explorer:

- Path: 내 PC > 로컬 디스크 (C:) > myNode > joinForm > public > javascripts
- File: jquery-3.6.1.min.js



4. 회원가입 폼 만들기 및 nodemon 사용

- 다음 소스코드를 따라서 작성하여 views 폴더에 joinForm.ejs 템플릿 생성

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title><%= title %></title>
5      <meta charset='utf-8'>
6      <link rel='stylesheet' href='/stylesheets/style.css' />
7      <script src="/javascripts/jquery-3.6.1.min.js"></script>
8    <script>
9      $(function () {
10
11        $("#btnJoin").click(
12          function () {
13            if ($("#id").val() == "") {
14              alert("아이디를 꼭 입력 하세요");
15              $("#id").focus();
16              return;
17            }
18            if ($("#passwd").val() == "") {
19              alert("비밀번호를 꼭 입력 하세요");
20              $("#passwd").focus();
21              return;
22            }
23            if ($("#name").val() == "") {
24              alert("이름을 꼭 입력 하세요");
25              $("#name").focus();
26              return;
27            }
28            if ($("#email").val() == "") {
29              alert("이메일을 꼭 입력 하세요");
30              $("#email").focus();
31              return;
32            }

```



4. 회원가입 폼 만들기 및 nodemon 사용

```
38     if($("#tel2").val() == "")
39     {
40         alert("전화번호 두번째 자리를 확인해 주세요");
41         $("#tel2").focus();
42         return;
43     }
44     if($("#tel3").val() == "")
45     {
46         alert("전화번호 세번째 자리를 확인해 주세요");
47         $("#tel3").focus();
48         return;
49     }
50
51     var tel= $("#tel1").val() + '-' + $("#tel2").val() + '-' + $("#tel3").val();
52     $("#tel").val(tel);
53
54
55     if($("#address").val() == ""){
56         alert("주소를 꼭 입력하세요");
57         $("#address").focus();
58         return;
59     }
60
61     if($("#birth").val() == ""){
62         alert("생일을 꼭 입력하세요");
63         $("#birth").focus();
64         return;
65     }
66
67     if($("#birth").val().length != 8){
68         alert("생일은 8글자로 입력하세요");
69         $("#birth").focus();
70         return;
71     }
72     if(isNaN($("#birth").val())){
73         alert("생일은 숫자만 입력하세요");
74         $("#birthb").focus();
75         return;
76     }
77     $("#joinForm").submit();
78
79     });
80
81     </script>
82 </head>
```



4. 회원가입 폼 만들기 및 nodemon 사용

```
83     <body>
84         <h1><%= title %></h1>
85         <form id="joinForm" action="/join" method="post">
86             <table width="400" border="1">
87                 <tr>
88                     <th>아이디</th>
89                     <td><input type="text" name="id" id="id" size="12" maxlength="12"></td>
90                 </tr>
91                 <tr>
92                     <th>비밀번호</th>
93                     <td><input type="password" name="passwd" id="passwd" size="12" maxlength="12"></td>
94                 </tr>
95                 <tr>
96                     <th>이름</th>
97                     <td><input type="text" name="name" id="name" size="10" maxlength="10"></td>
98                 </tr>
99                 <tr>
100                     <th>이메일</th>
101                     <td><input type="email" name="email" id="email" size="30" maxlength="50"></td>
102                 </tr>
103                 <tr>
104                     <th>전화번호</th>
105                     <td>
106                         <input type="hidden" name="tel" id="tel">
107                         <select name="tel1" id="tel1">
108                             <option value="010">010</option>
109                             <option value="011">011</option>
110                             <option value="016">016</option>
111                             <option value="017">017</option>
112                             <option value="018">018</option>
113                             <option value="019">019</option>
114                         </select>
115                         <input type="text" name="tel2" id="tel2" size="4" maxlength="4">
116                         <input type="text" name="tel3" id="tel3" size="4" maxlength="4">
117                     </td>
118                 </tr>
```

< 소스코드 3/4 >



4. 회원가입 폼 만들기 및 nodemon 사용

```
119     <tr>
120         <th>주소</th>
121         <td><input type="text" id="address" size="30" maxlength="50"></td>
122     </tr>
123     <tr>
124         <th>직업</th>
125         <td>
126             <select name="job" id="job">
127                 <option value="학생">학생</option>
128                 <option value="직장인">직장인</option>
129                 <option value="주부">주부</option>
130             </select>
131         </td>
132     </tr>
133     <tr>
134         <th>성별</th>
135         <td>
136             <input type="radio" name="gender" value="남" checked="checked">남
137             <input type="radio" name="gender" value="여">여
138         </td>
139     </tr>
140     <tr>
141         <th>생일</th>
142         <td><input type="text" name="birth" id="birth" size="8" maxlength="8" value="*YYYYMMDD"></td>
143     </tr>
144     <tr>
145         <td colspan="2">
146             <input type="button" id="btnJoin" name="join" value="가입">
147             <input type="reset" name="cancel" value="취소">
148         </td>
149     </tr>
150 </table>
151 </form>
152 </body>
153 </html>
```




4. 회원가입 폼 만들기 및 nodemon 사용

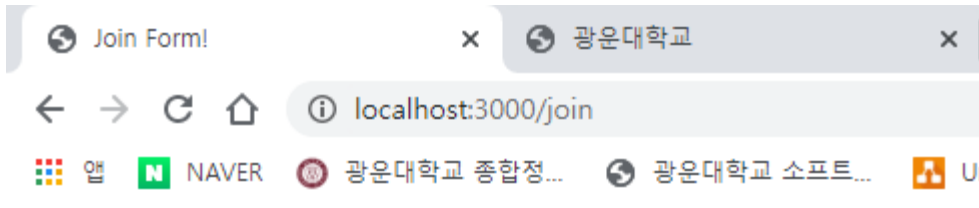
- Routes폴더의 joinForm.js의 res.send 부분을 다음과 같이 수정

```
joinForm.js  x  app.js  x  joinForm.ejs  x
1  var express = require('express');
2  var router = express.Router();
3
4  /* GET users listing. */
5  router.get('/', function(req, res, next){
6    res.render('joinForm', {title: 'Join Form!'});
7  });
8
9  module.exports = router;
```



4. 회원가입 폼 만들기 및 nodemon 사용

- 서버 재시작 후 웹 브라우저를 통해 결과 확인



Join Form!

아이디	<input type="text"/>		
비밀번호	<input type="password"/>		
이름	<input type="text"/>		
이메일	<input type="text"/>		
전화번호	010 ▾	<input type="text"/>	<input type="text"/>
주소	<input type="text"/>		
직업	학생 ▾		
성별	<input checked="" type="radio"/> 남 <input type="radio"/> 여		
생일	<input type="text"/> *YYYYMMDD		
<input type="button" value="가입"/> <input type="button" value="취소"/>			



4. 회원가입 폼 만들기 및 nodemon 사용

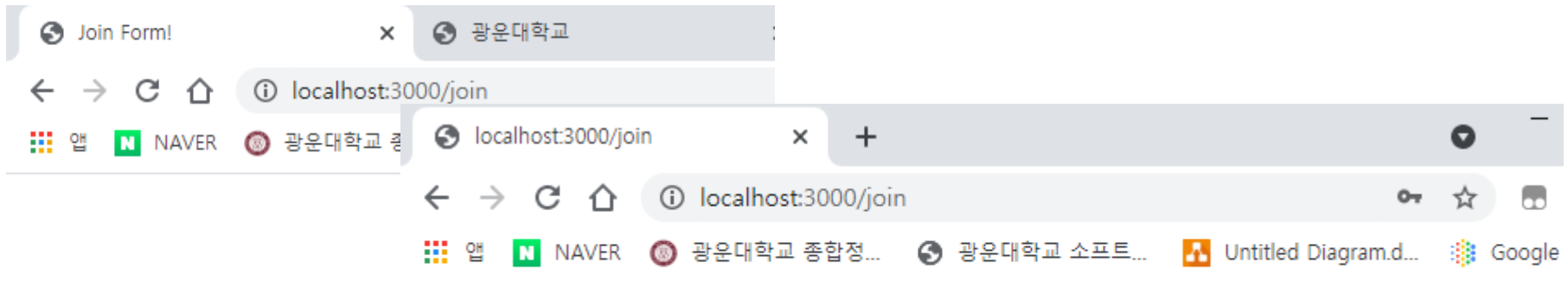
- 사용자 입력값을 JSON으로 출력하기 위해 joinForm.js에 다음 소스코드 추가

```
joinForm.js  x  app.js  x  joinForm.ejs  x
1  var express = require('express');
2  var router = express.Router();
3
4  /* GET users listing. */
5  router.get('/', function(req, res, next){
6      res.render('joinForm', {title: 'Join Form!'});
7  });
8
9  router.post('/', function(req, res, next){
10     console.log('req.body: ' + JSON.stringify(req.body));
11     res.json(req.body);
12 })
13
14 module.exports = router;
```



4. 회원가입 폼 만들기 및 nodemon 사용

- 서버 재시작 후 웹 브라우저를 통해 결과 확인



Join Form!

```
{"id":"SE","passwd":"1234","name":"DSL","email":"micky1996@naver.com","tel":"010-1234-5678","tel1":"010","tel2":"1234","tel3":"5678","job":"학생","gender":"남","birth":"19990101"}
```

아이디	SE
비밀번호
이름	DSL
이메일	micky1996@naver.com
전화번호	010 ▼ 1234 5678
주소	광운대학교
직업	학생 ▼
성별	<input checked="" type="radio"/> 남 <input type="radio"/> 여
생일	19990101 *YYYYMMDD
<input type="button" value="가입"/> <input type="button" value="취소"/>	



4. 회원가입 폼 만들기 및 nodemon 사용

- nodemon
 - 소스코드를 수정할 때 마다 서버를 재시작하지 않아도 됨
 - 수정된 소스가 콘솔창에 나타남
- > npm install -g nodemon

```
명령 프롬프트
C:\myNode\joinForm>npm install -g nodemon
C:\Users\micky\AppData\Roaming\npm\nodemon -> C:\Users\micky\AppData\Roaming\npm\node_modules\nodemon\bin\nodemon.js

> nodemon@2.0.7 postinstall C:\Users\micky\AppData\Roaming\npm\node_modules\nodemon
> node bin/postinstall || exit 0

Love nodemon? You can now support the project via the open collective:
> https://opencollective.com/nodemon/donate

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@~2.3.1 (node_modules\nodemon\node_modules\chokidar\node_modules\
fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.3.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

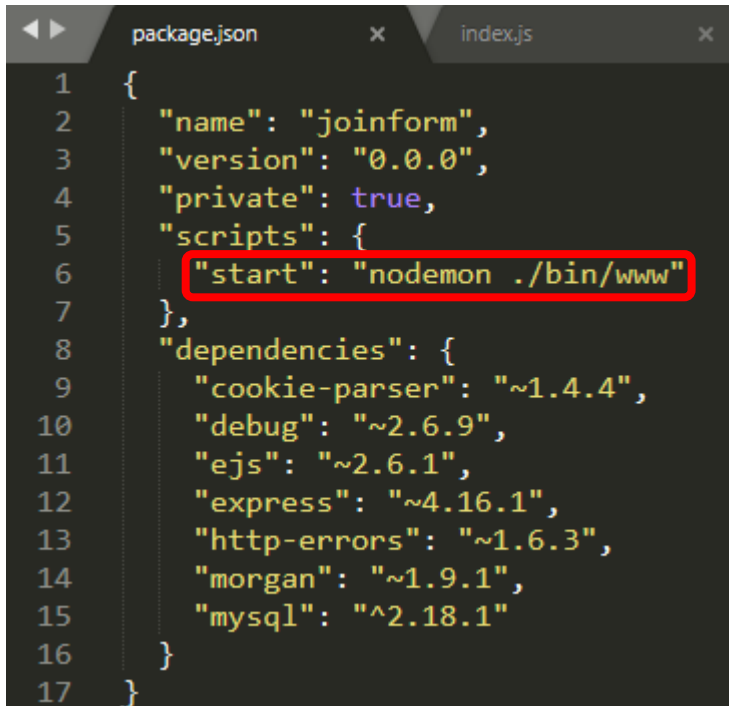
+ nodemon@2.0.7
removed 1 package and updated 23 packages in 4.408s

C:\myNode\joinForm>
```



4. 회원가입 폼 만들기 및 nodemon 사용

■ package.json 수정

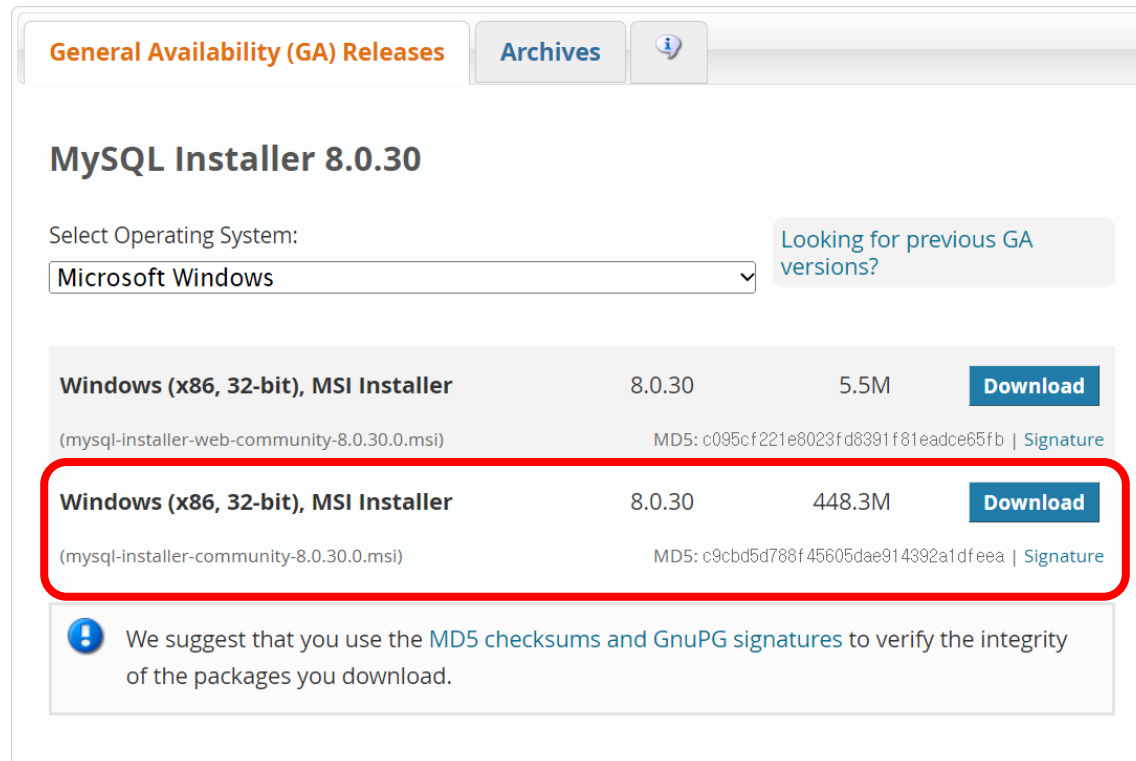


```
package.json x index.js x
1 {
2   "name": "joinform",
3   "version": "0.0.0",
4   "private": true,
5   "scripts": {
6     "start": "nodemon ./bin/www"
7   },
8   "dependencies": {
9     "cookie-parser": "~1.4.4",
10    "debug": "~2.6.9",
11    "ejs": "~2.6.1",
12    "express": "~4.16.1",
13    "http-errors": "~1.6.3",
14    "morgan": "~1.9.1",
15    "mysql": "^2.18.1"
16  }
17 }
```



5. Database(MySQL)

- MySQL 설치
 - 다음 URL을 웹 브라우저에 입력함
 - <https://dev.mysql.com/downloads/installer>



General Availability (GA) Releases Archives ⓘ

MySQL Installer 8.0.30

Select Operating System:
Microsoft Windows

Looking for previous GA versions?

Windows (x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.30.0.msi)	8.0.30	5.5M	Download
Windows (x86, 32-bit), MSI Installer (mysql-installer-community-8.0.30.0.msi)	8.0.30	448.3M	Download

MD5: c095cf221e8023fd8391f81eadce65fb | [Signature](#)

MD5: c9cbd5d788f45605dae914392a1dfeea | [Signature](#)

! We suggest that you use the [MD5 checksums](#) and [GnuPG signatures](#) to verify the integrity of the packages you download.



5. Database(MySQL)

Login Now or Sign Up for a free account.

An Oracle Web Account provides you with the following advantages:

- Fast access to MySQL software downloads
- Download technical White Papers and Presentations
- Post messages in the MySQL Discussion Forums
- Report and track bugs in the MySQL bug system

Login »

using my Oracle Web account

Sign Up »

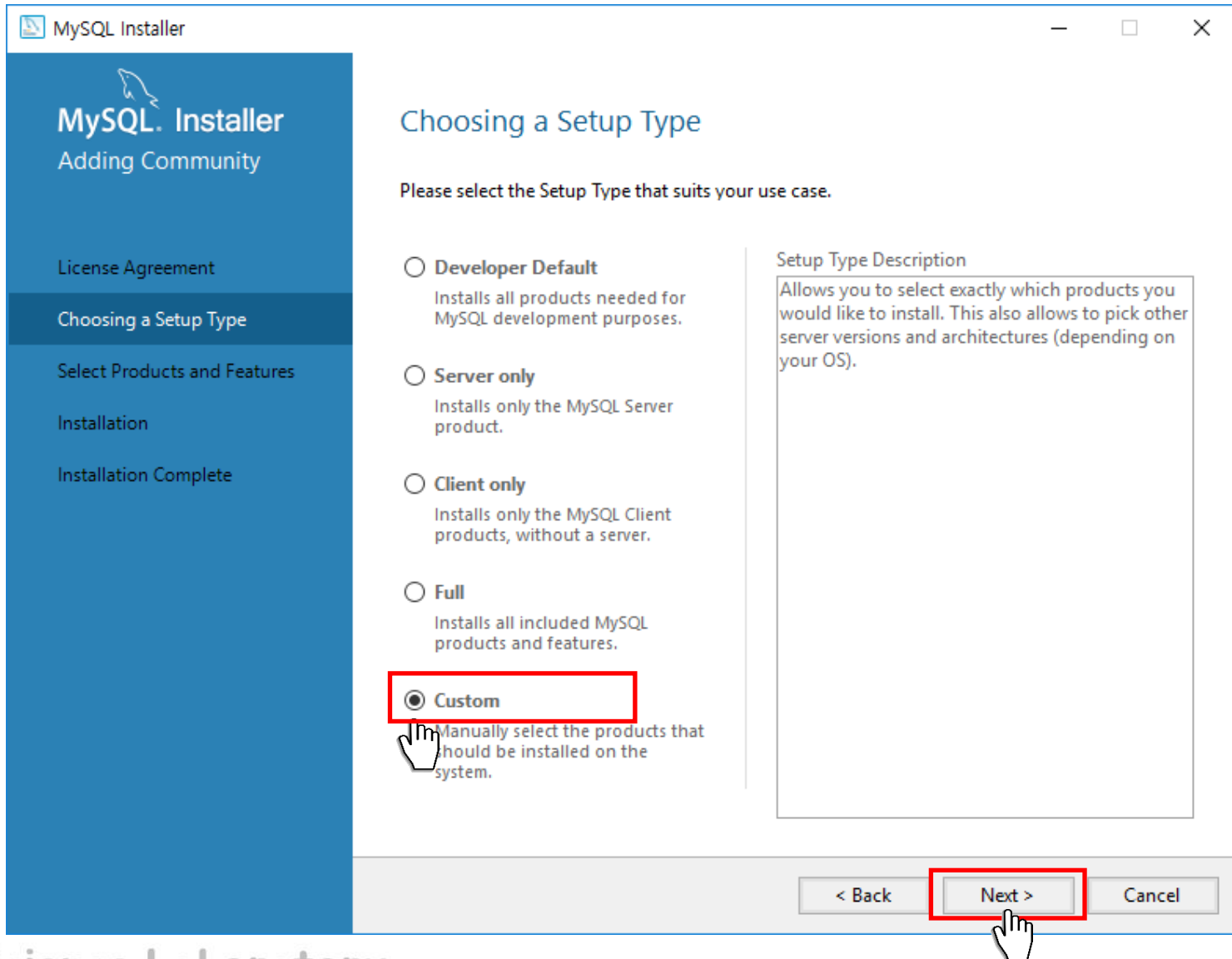
for an Oracle Web account

MySQL.com is using Oracle SSO for authentication. If you already have an Oracle Web account, click the Login link. Otherwise, you can signup for a free account by clicking the Sign Up link and following the instructions.

No thanks, just start my download.



5. Database(MySQL)



MySQL Installer

Adding Community

License Agreement

Choosing a Setup Type

Select Products and Features

Installation

Installation Complete

Choosing a Setup Type

Please select the Setup Type that suits your use case.

- ☐ **Developer Default**
Installs all products needed for MySQL development purposes.
- ☐ **Server only**
Installs only the MySQL Server product.
- ☐ **Client only**
Installs only the MySQL Client products, without a server.
- ☐ **Full**
Installs all included MySQL products and features.
- ☒ **Custom**
Manually select the products that should be installed on the system.

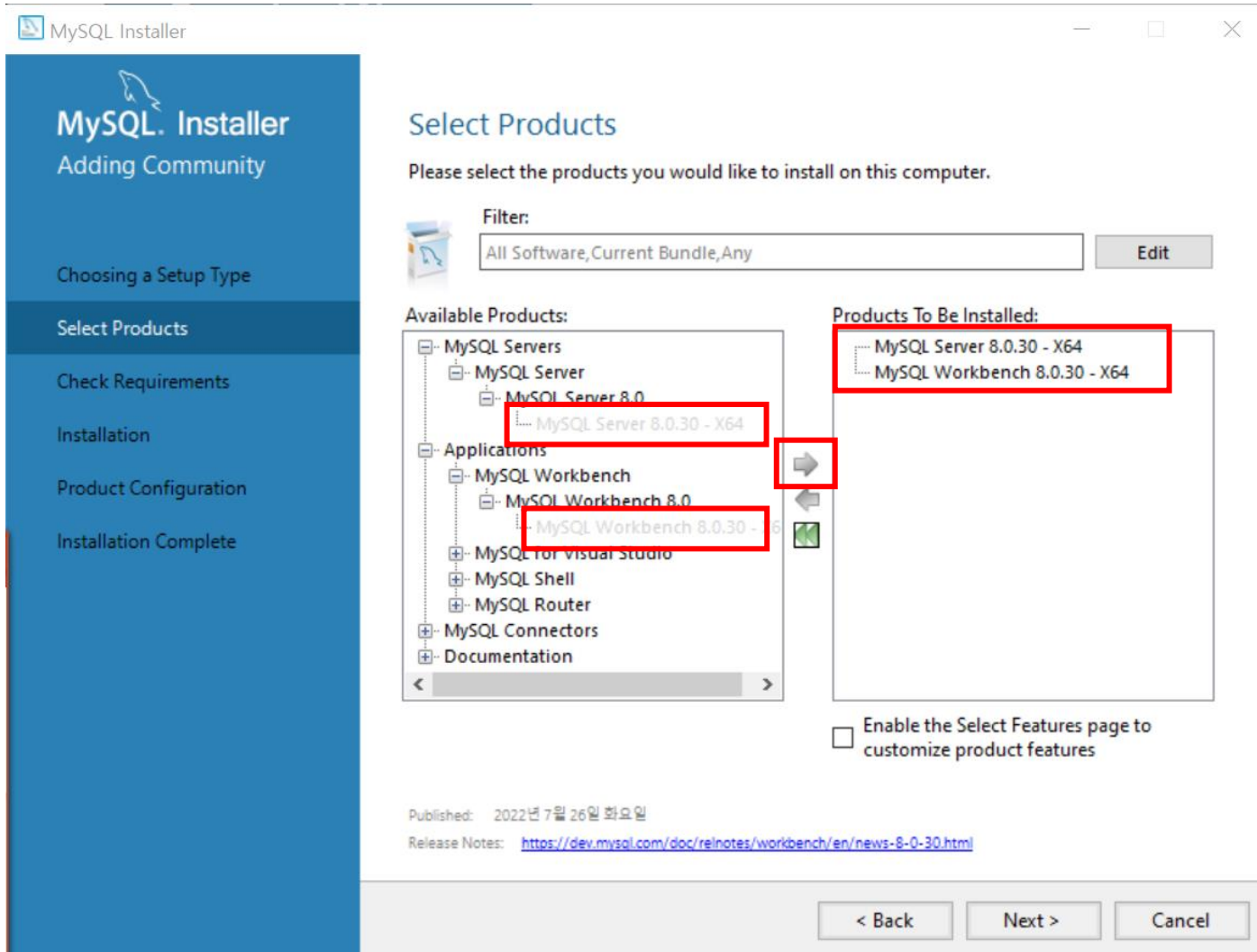
Setup Type Description

Allows you to select exactly which products you would like to install. This also allows to pick other server versions and architectures (depending on your OS).

< Back **Next >** Cancel

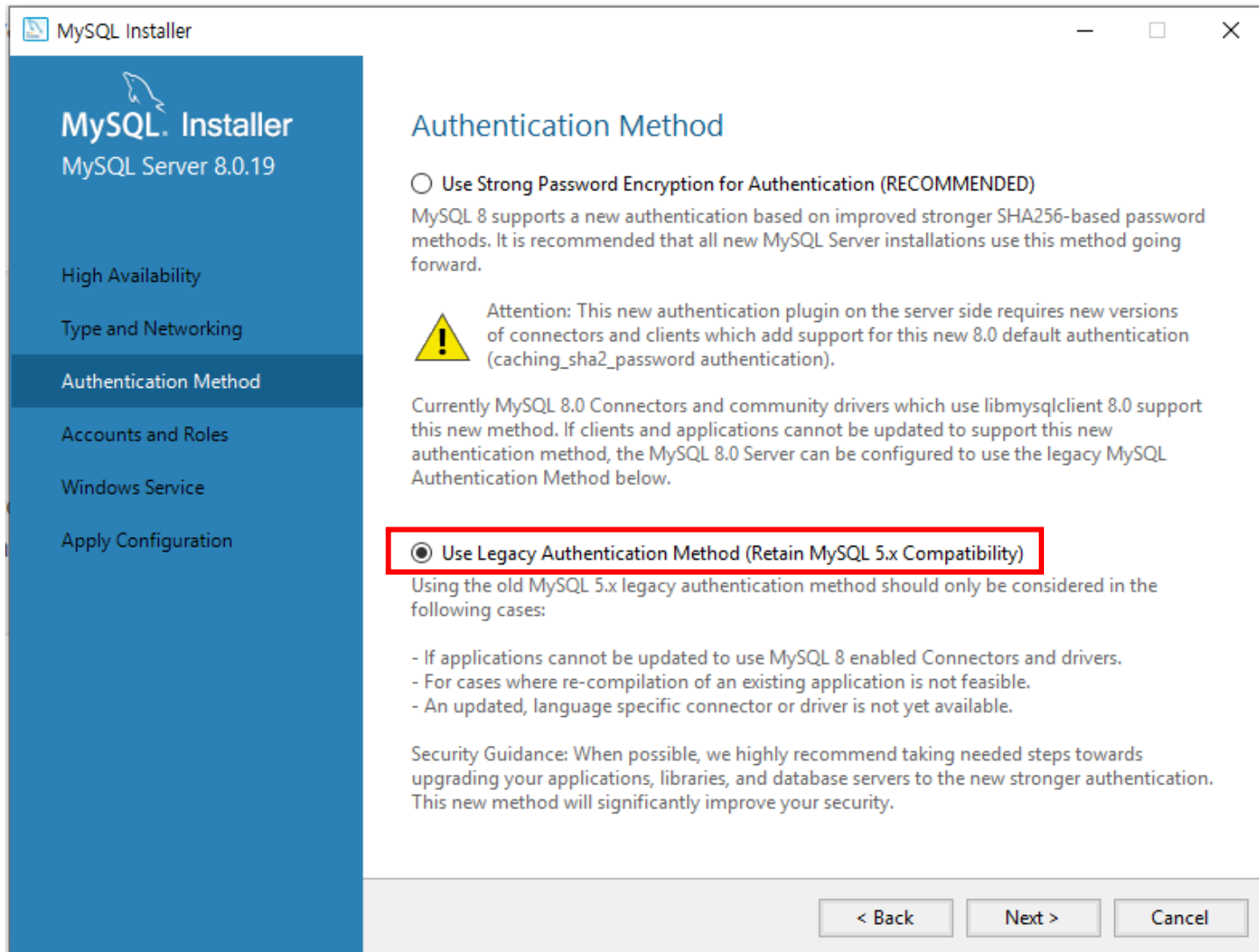


5. Database(MySQL)



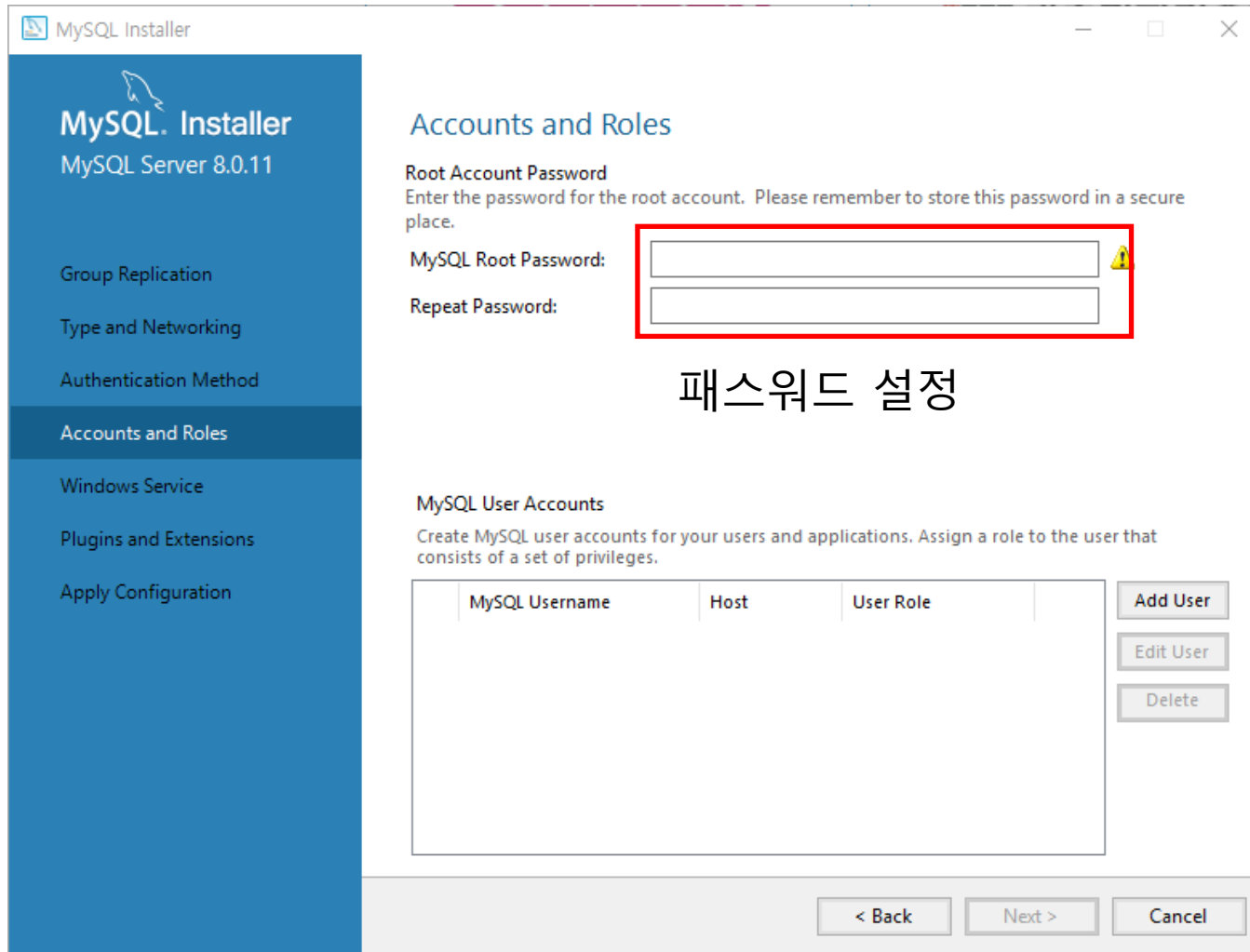


5. Database(MySQL)





5. Database(MySQL)



The screenshot shows the MySQL Installer window for MySQL Server 8.0.11. The left sidebar lists various configuration options, with 'Accounts and Roles' selected. The main area is titled 'Accounts and Roles' and contains two sections: 'Root Account Password' and 'MySQL User Accounts'. The 'Root Account Password' section has two input fields for the root password, which are highlighted with a red rectangle. The 'MySQL User Accounts' section features a table with columns for 'MySQL Username', 'Host', and 'User Role', along with 'Add User', 'Edit User', and 'Delete' buttons. At the bottom, there are '< Back', 'Next >', and 'Cancel' buttons.

MySQL Installer
MySQL Server 8.0.11

Group Replication
Type and Networking
Authentication Method
Accounts and Roles
Windows Service
Plugins and Extensions
Apply Configuration

Accounts and Roles

Root Account Password
Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password:

Repeat Password:

MySQL User Accounts

Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

MySQL Username	Host	User Role
----------------	------	-----------

Add User
Edit User
Delete

< Back Next > Cancel

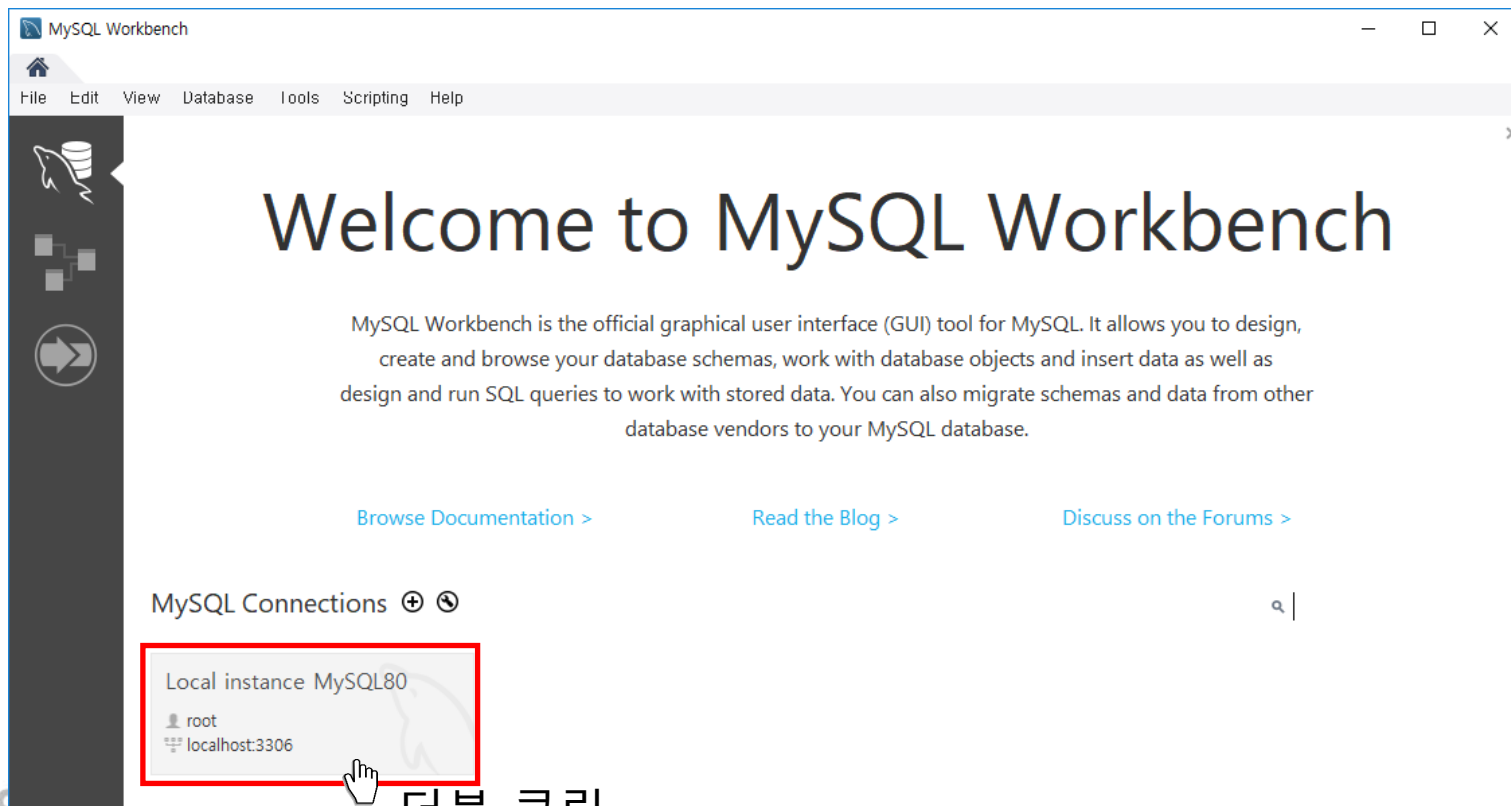
패스워드 설정



5. Database(MySQL)

■ MySQL Workbench

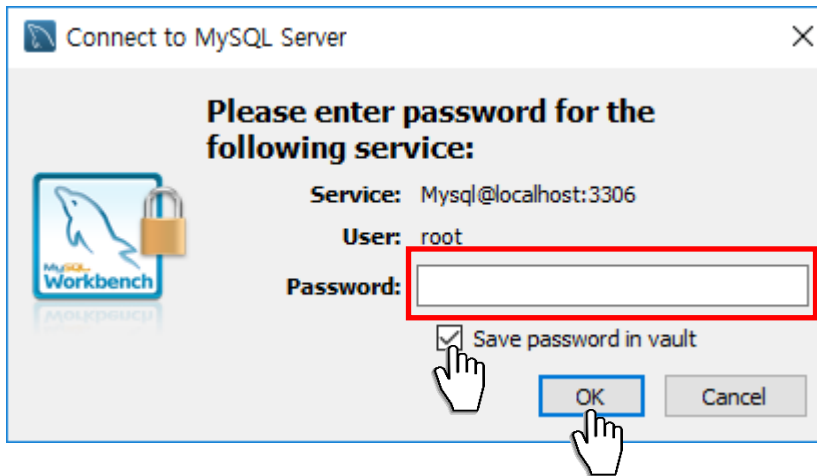
- 데이터베이스 설계 및 구현을 위한 GUI 프로그램
- SQL 편집, 데이터베이스 모델링, 데이터베이스 관리, 데이터베이스 마이그레이션 등의 기능을 제공함





5. Database(MySQL)

- MySQL Workbench 실행

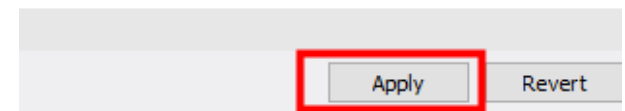
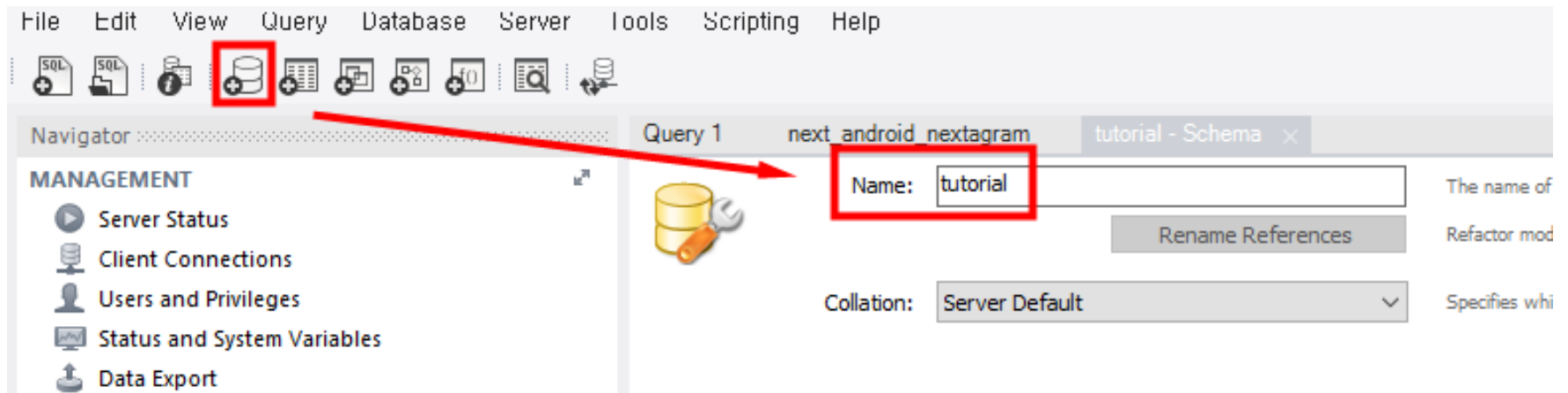


앞에서 설정한 암호입력



5. Database(MySQL)

■ Tutorial 데이터베이스 생성





5. Database(MySQL)

Apply SQL Script to Database

Review SQL Script
Apply SQL Script

Review the SQL Script to be Applied on the Database

Online DDL

Algorithm: Lock Type:

```
1 CREATE SCHEMA `tutorial` ;
2
```




5. Database(MySQL)

Apply SQL Script to Database

Review SQL Script

Apply SQL Script

Applying SQL script to the database

The following tasks will now be executed. Please monitor the execution.
Press Show Logs to see the execution logs.

☒ Execute SQL Statements

SQL script was successfully applied to the database.

Show Logs

Back

Finish

Cancel



5. Database(MySQL)

■ SQL 편집기 사용법

- 단일 명령문 실행
 - 실행하고자 하는 명령문으로 커서를 옮긴 다음 “Ctrl키 + Enter키” 혹은 번개+커서 아이콘 클릭
- 다중 명령문 실행
 - 실행하고자 하는 명령문들을 선택한 다음 “Ctrl키 + Shift키 + Enter키” 혹은 번개 아이콘 클릭
 - 명령문들을 선택하지 않으면 현재 편집창의 모든 명령문을 실행함
- 모든 명령문은 세미콜론(;)으로 끝나야 함





5. Database(MySQL)

- SELECT 문은 SQL 질의(또는 쿼리)라고도 부름
- 기본적인 SELECT 문은 다음과 같은 구조를 가짐

```
SELECT [DISTINCT] { * | column [alias], ...}  
FROM      table  
[ WHERE  condition ]  
[ GROUP BY      group_by_expression ]  
[ HAVING group_condition ]  
[ ORDER BY      column ];
```

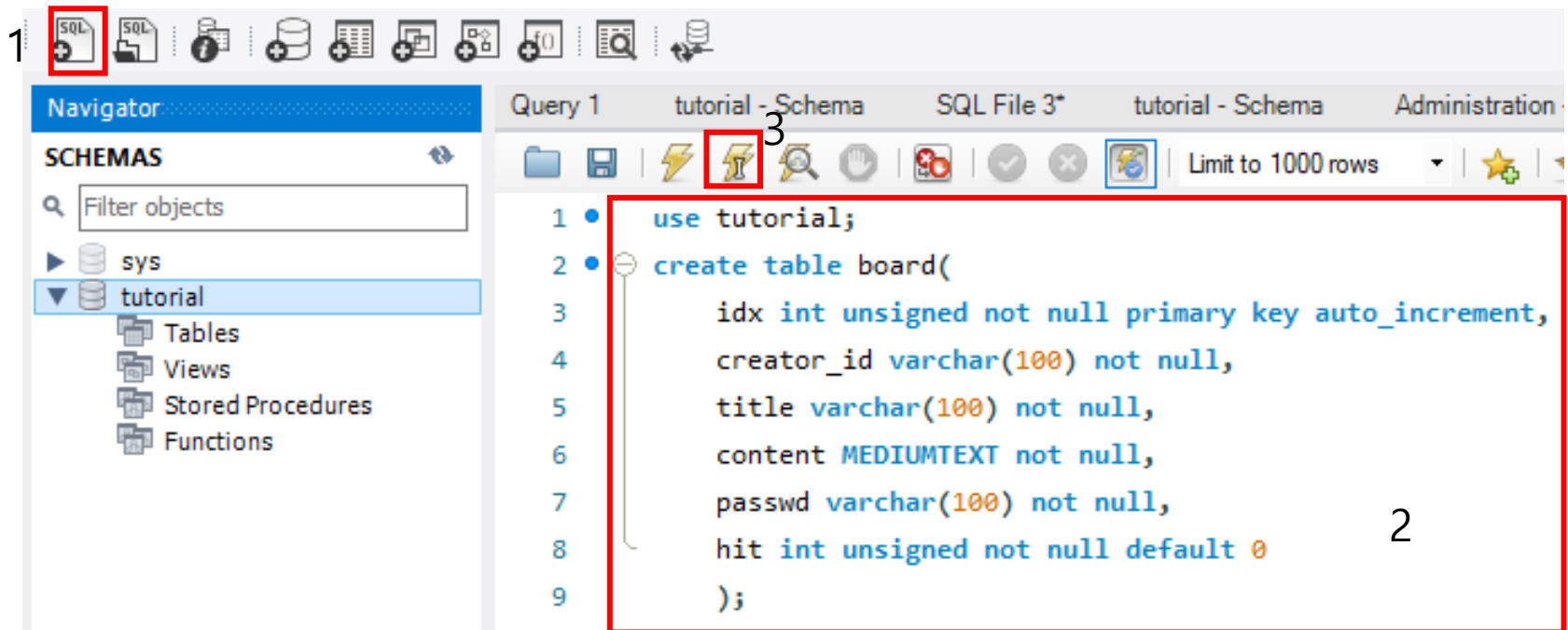
- SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY 각각을 절(clause)이라고 부르며 순서대로 나와야 함
- '[']' 는 생략 가능하다는 것을 의미하고, '|' 는 OR를 의미함
- condition은 TRUE 또는 FALSE 값을 가지는 조건식을 의미함
- 컬럼명 앞에 테이블명을 명시할 수 있음
 - 예: EMP 테이블의 NAME 컬럼을 EMP.NAME으로 표시



5. Database(MySQL) 연동

- board 테이블 생성

1



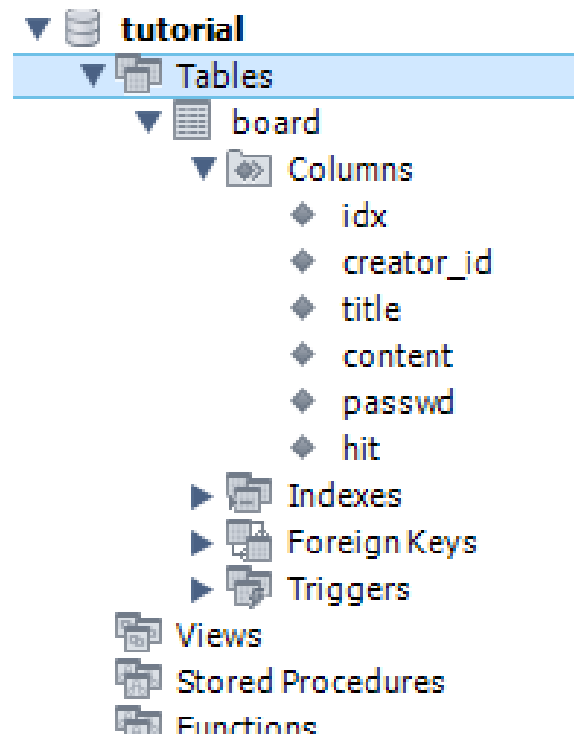
```
1 • use tutorial;
2 • create table board(
3     idx int unsigned not null primary key auto_increment,
4     creator_id varchar(100) not null,
5     title varchar(100) not null,
6     content MEDIUMTEXT not null,
7     passwd varchar(100) not null,
8     hit int unsigned not null default 0
9 );
```

2



5. Database(MySQL) 연동

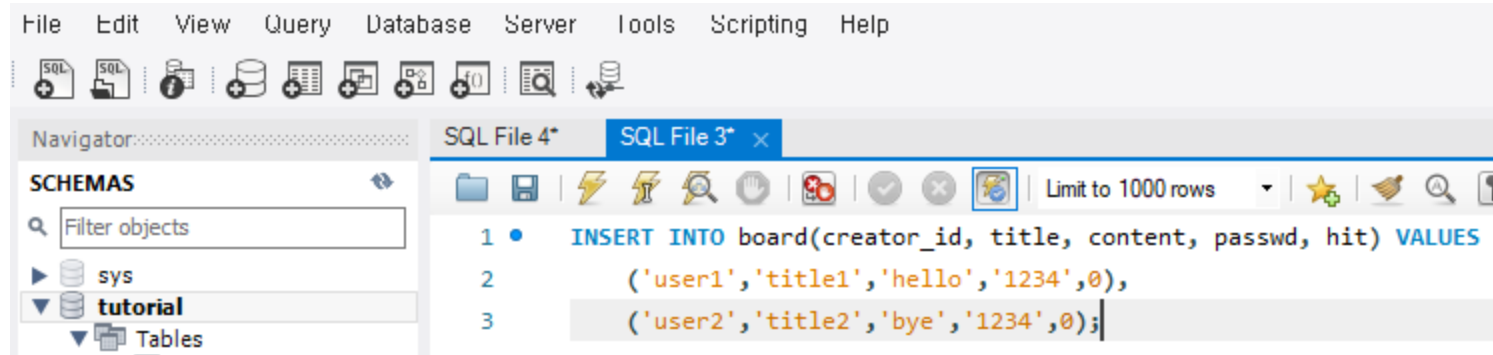
- Tables 우클릭 -> Refresh All 하여, 생성한 테이블 확인





5. Database(MySQL) 연동

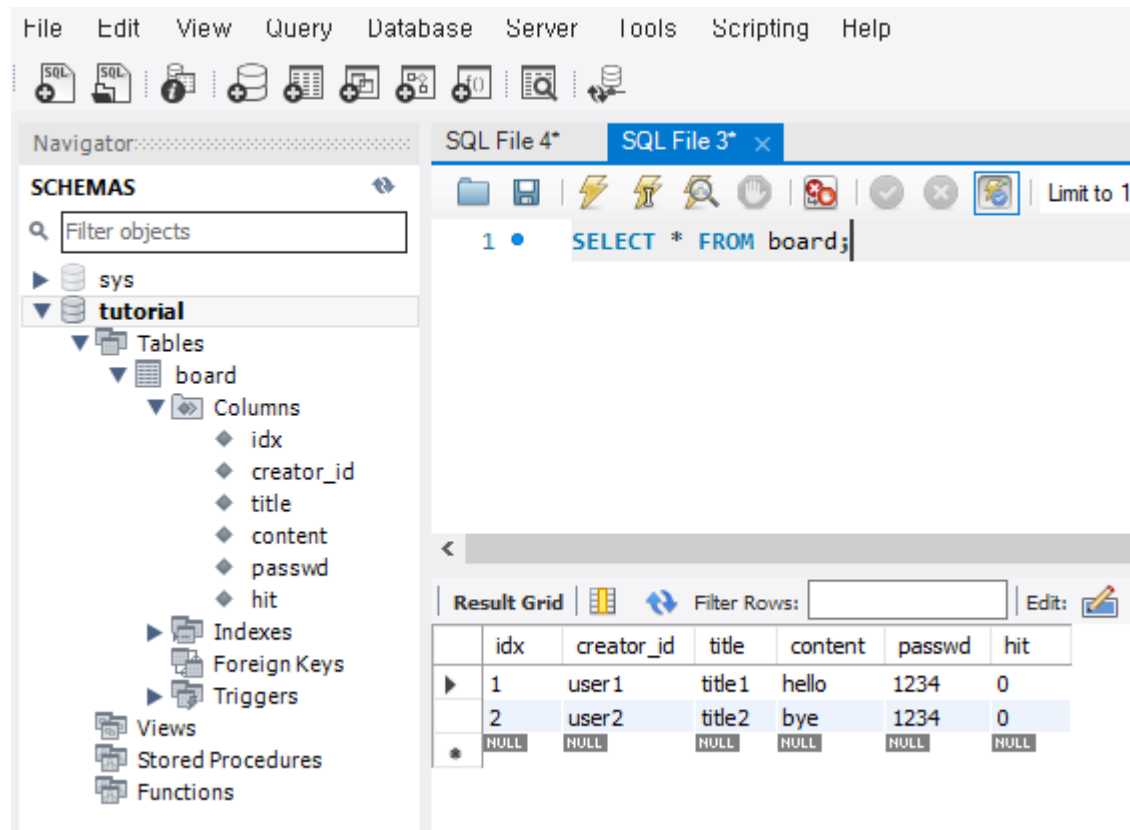
- Test 데이터 입력





5. Database(MySQL) 연동

- Test 데이터 확인



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'tutorial' expanded, showing the 'board' table. The main editor window shows a query: `SELECT * FROM board;`. The 'Result Grid' at the bottom displays the following data:

	idx	creator_id	title	content	passwd	hit
▶	1	user1	title1	hello	1234	0
	2	user2	title2	bye	1234	0
*	NULL	NULL	NULL	NULL	NULL	NULL



5. Database(MySQL) 연동

- Node.js 에 mysql 설치 > npm install mysql

```
명령 프롬프트
C:\myNode\joinForm>npm install mysql
+ mysql@2.18.1
added 9 packages from 14 contributors and audited 64 packages in 1.417s
found 0 vulnerabilities

C:\myNode\joinForm>
```




5. Database(MySQL) 연동

■ routesWindex.js 수정

```
index.js x index.ejs x joinForm.js x package.json x
1  var express = require('express');
2  var router = express.Router();
3  var mysql = require('mysql');
4  var pool = mysql.createPool({
5    connectionLimit: 5,
6    host: 'localhost',
7    user: 'root',
8    password: '1234',
9    database: 'tutorial'
10 });
11
12 /* GET home page. */
13 router.get('/', function(req, res, next){
14   pool.getConnection(function(err, connection){
15     //Use the connection
16     connection.query('SELECT * FROM board', function(err, rows){
17       if(err) console.error("err : "+err);
18       console.log("rows : " +JSON.stringify(rows));
19
20       res.render('index', {title: 'test', rows: rows});
21       connection.release();
22
23       // Don't use the connection here, it has been returned to the pool.
24     });
25   });
26 });
27
28
29
30
31 module.exports = router;
```



5. Database(MySQL) 연동

- 연동이 잘 안된 경우 Mysql Client에서 초기 비밀번호 입력

```
MySQL 8.0 Command Line Client - Unicode
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 46
Server version: 8.0.19 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '1234';
Query OK, 0 rows affected (0.02 sec)

mysql>
```



5. Database(MySQL) 연동

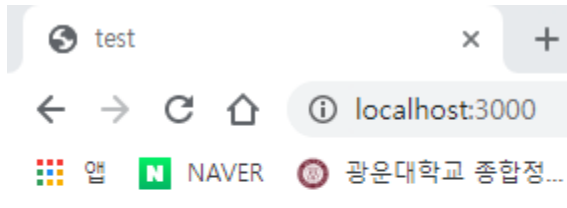
- views#index.ejs 수정

```
index.js  x  index.ejs  x  joinForm.js  x  packa
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title><%= title %></title>
5      <link rel='stylesheet' href='/stylesheets/style.css' />
6    </head>
7    <body>
8      <h1><%= title %></h1>
9      <p>Welcome to <%= title %></p>
10     <p><%= rows[0].idx %></p>
11     <p><%= rows[0].creator_id %></p>
12     <p><%= rows[0].title %></p>
13     <p><%= rows[0].content %></p>
14     <p><%= rows[0].regdate %></p>
15   </body>
16 </html>
```



5. Database(MySQL) 연동

■ 출력 결과를 분석 및 확인



test

Welcome to test

1

user1

title1

hello



6. 게시판 구축

- URL 설계(GET Method)
 - 게시판 글 목록 표시 화면
 - <http://localhost/board/list/?page=1>
 - 게시판 글 조회 화면
 - <http://localhost/board/read/?idx=1>
 - 게시판 글 쓰기 화면
 - <http://localhost/board/write>
 - 게시판 글 수정 화면
 - <http://localhost/board/modify>
- URL 설계(POST Method)
 - 게시판 글 쓰기 액션
 - <http://localhost/board/write>
 - 게시판 글 수정 액션
 - <http://localhost/board/modify>



6. 게시판 구축

■ app.js 소스 추가

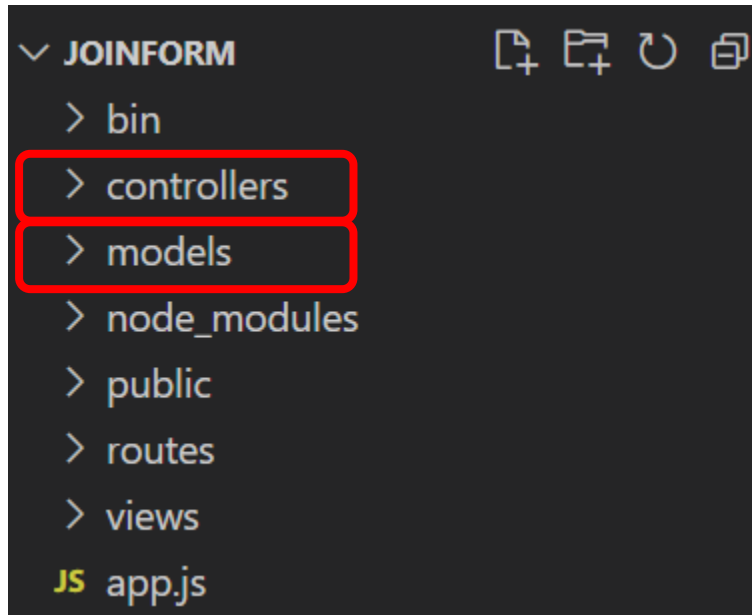
```
7  var indexRouter = require('./routes/index');  
8  var usersRouter = require('./routes/users');  
9  var join = require('./routes/joinForm');  
10 var board = require('./routes/board');
```

```
24 app.use('/', indexRouter);  
25 app.use('/users', usersRouter);  
26 app.use('/join', join);  
27 app.use('/board', board);
```



6. 게시판 구축

- Controllers, models 디렉토리 생성





6. 게시판 구축

- routes\board.js 생성

```
board.js
1  var express = require('express');
2  var router = express.Router();
3  var listController = require('../controllers/listController');
4
5  router.get('/', listController.getListFirst);
6  router.get('/list/:idx', listController.getList);
7
8  module.exports = router;
```




6. 게시판 구축

- controllers\listControllers.js 생성

```
JS listController.js •
controllers > JS listController.js > ...
1  var listModel=require('../models/listModel');
2  var express=require('express');
3
4
5  exports.getList=(req,res,next)=>{
6      listModel.getList((rows)=>{
7          console.log('rows: '+JSON.stringify(rows));
8          res.render('list',{title: "게시판 전체 글 조회", rows: rows});
9      });
10 }
11 exports.getListFirst=(req,res)=>{
12     res.redirect('/board/list/1');
13 }
```



6. 게시판 구축

■ models\listModel.js 생성

```
JS board.js  JS listModel.js  ●
models > JS listModel.js > ...
1  var mysql = require('mysql');
2  var connection = mysql.createConnection({
3    connectionLimit: 5,
4    host: 'localhost',
5    user: 'root',
6    password: '1234',
7    database: 'tutorial'
8  });
9
10 module.exports={getList(callback){
11   connection.query('SELECT idx, creator_id, title, hit FROM board',(err,rows,fileds)=>{
12     if(err) throw err;
13     callback(rows);
14   });
15 }
16 }
17
```



6. 게시판 구축

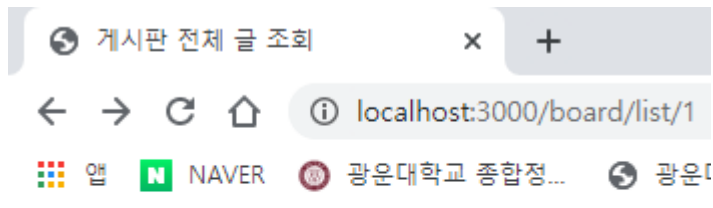
■ views\list.ejs 생성

```
list.ejs  x  read.ejs  x  write.ejs  x  board.js
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title><%= title %></title>
5      <link rel='stylesheet' href='/stylesheets/style.css'/>
6    </head>
7    <body>
8      <h1><%= title %></h1>
9      <a href="/board/write">글쓰기로 이동</a>
10     <br>
11     <br>
12     <table border="1">
13       <tr>
14         <td>번호</td>
15         <td>작성자</td>
16         <td>제목</td>
17         <td>조회수</td>
18       </tr>
19     <%
20       for(var i=0; i<rows.length; i++){
21         var oneItem = rows[i];
22       %>
23       <tr>
24         <td><%=oneItem.idx%></td>
25         <td><%=oneItem.creator_id%></td>
26         <td><%=oneItem.title%></td>
27         <td><%=oneItem.hit%></td>
28       </tr>
29     <%
30       }
31     %>
32   </table>
33 </body>
34 </html>
```



6. 게시판 구축

■ 결과 확인



게시판 전체 글 조회

[글쓰기로 이동](#)

번호	작성자	제목	조회수
1	user1	title1	0
2	user2	title2	0



6. 게시판 구축

■ 글쓰기

- board.js에 소스 추가

```
board.js
1  var express = require('express');
2  var router = express.Router();
3  var listController = require('../controllers/listController');
4  var writeController = require('../controllers/writeController');
5
6  router.get('/', listController.getListFirst);
7  router.get('/list/:idx', listController.getList);
8  router.get('/write', writeController.writeForm);
9  router.post('/write', writeController.writeData);
10
11 module.exports = router;
```



6. 게시판 구축

- 글쓰기
 - Controllers\writeController.js 생성

```
JS board.js • JS writeController.js X
controllers > JS writeController.js > writeData > writeData > writeModel.insertData() callback
1  var writeModel=require('../models/writeModel');
2  var express=require('express');
3
4  exports.writeForm=(req,res)=>{
5      res.render('write',{title: "게시판 글 쓰기"});
6  }
7  exports.writeData=(req,res)=>{
8      var creator_id = req.body.creator_id;
9      var title = req.body.title;
10     var content = req.body.content;
11     var passwd = req.body.passwd;
12     var datas = [creator_id, title, content, passwd];
13     writeModel.insertData(datas,()=>{
14         res.redirect('/board');
15     });
16 };
```



6. 게시판 구축

- 글쓰기
 - models\writeModel.js 생성

```
JS board.js • JS writeModel.js X
models > JS writeModel.js > ...
1  var mysql = require('mysql');
2  var connection = mysql.createConnection({
3    connectionLimit: 5,
4    host: 'localhost',
5    user: 'root',
6    password: '1234',
7    database: 'tutorial'
8  });
9
10 module.exports={insertData(datas,callback){
11   var sql="INSERT INTO board(creator_id, title, content, passwd) VALUES(?,?,?,?)";
12   connection.query(sql,datas,function(err,rows){
13     if(err) console.error("err : " +err);
14     console.log("rows : " + JSON.stringify(rows));
15     callback();
16   });
17 }
18 }
19
```



6. 게시판 구축

- 글쓰기
- views#write.ejs

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <script src="/javascripts/jquery-3.6.1.min.js"></script>
5     <title><%= title %></title>
6     <link rel='stylesheet' href='/stylesheets/style.css' />
7   </head>
8   <body>
9     <h1><%= title %></h1>
10    <form action="/board/write" method="post" onsubmit="return onWriteSubmit()">
11      <table border="1">
12        <tr>
13          <td>작성자</td>
14          <td><input type="text" name="creator_id" id="creator_id" required/></td>
15        </tr>
16        <tr>
17          <td>제목</td>
18          <td><input type="text" name="title" id="title" required/></td>
19        </tr>
20        <tr>
21          <td>내용</td>
22          <td><textarea name="content" id="content" cols="30" rows="10" required></textarea></td>
23        </tr>
24        <tr>
25          <td>패스워드</td>
26          <td><input type="password" name="passwd" id="passwd" required/></td>
27        </tr>
28        <tr>
29          <td colspan="2">
30            <button type="submit">글쓰기</button>
31          </td>
32        </tr>
33      </table>
34    </form>
```

< 소스코드 1/2 >



6. 게시판 구축

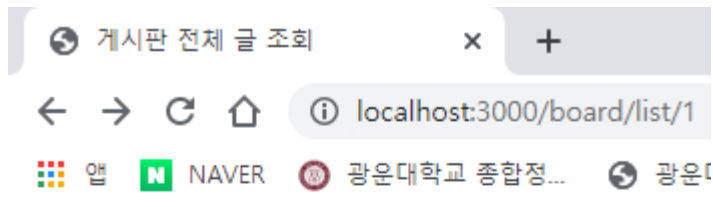
■ 글쓰기

```
35 <script>
36     function onWriteSubmit(){
37         if ( $("#creator_id").val().trim() == ""){
38             var message = "아이디를 입력해 주세요.";
39             $("#creator_id").val("");
40             $("#creator_id").focus();
41             alert(message);
42             return false;
43         }
44         if ( $("#title").val().trim() == "" ){
45             var message = "제목을 입력하세요.";
46             $("#title").val("");
47             $("#title").focus();
48             alert(message);
49             return false;
50         }
51         if ( $("#content").val().trim() == ""){
52             var message = "본문 내용을 입력해 주세요.";
53             $("#content").val("");
54             $("#content").focus();
55             alert(message);
56             return false;
57         }
58         if ( $("#passwd").val().trim() == ""){
59             var message = "패스워드를 입력하세요.";
60             $("#passwd").val("");
61             $("#passwd").focus();
62             alert(message);
63             return false;
64         }
65     }
66 </script>
67 </body>
68 </html>
```



6. 게시판 구축

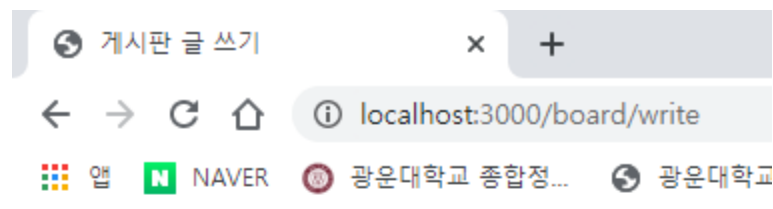
글쓰기 결과 확인



게시판 전체 글 조회

글쓰기로 이동

번호	작성자	제목	조회수
1	user1	title1	0
2	user2	title2	0



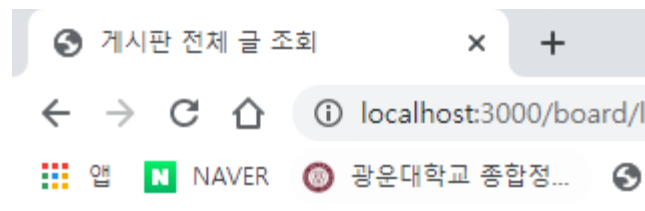
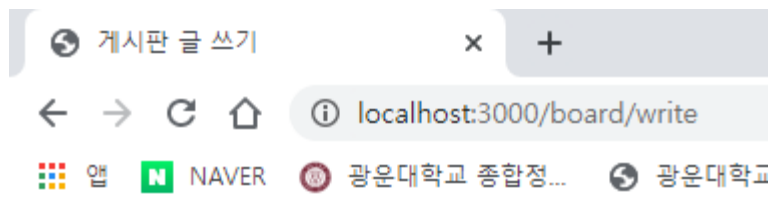
게시판 글 쓰기

작성자	<input type="text"/>
제목	<input type="text"/>
내용	<div></div>
패스워드	<input type="text"/>
글쓰기	



6. 게시판 구축

■ 결과 확인



게시판 글 쓰기

작성자	<input type="text" value="user3"/>
제목	<input type="text" value="글쓰기테스트"/>
내용	<div>테스트 테스트</div>
패스워드	<input type="password" value="...."/>
<input type="button" value="글쓰기"/>	

게시판 전체 글 조회

[글쓰기로 이동](#)

번호	작성자	제목	조회수
1	user1	title1	0
2	user2	title2	0
3	user3	글쓰기테스트	0



6. 게시판 구축

■ 조회

- board.js 파일에 아래 소스 추가

```
board.js
1  var express = require('express');
2  var router = express.Router();
3  var listController = require('../controllers/listController');
4  var writeController = require('../controllers/writeController');
5  var readController = require('../controllers/readController');
6
7  router.get('/', listController.getListFirst);
8  router.get('/list/:idx', listController.getList);
9  router.get('/write', writeController.writeForm);
10 router.post('/write', writeController.writeData);
11 router.get('/read/:idx', readController.readData);
12
13 module.exports=router;
```



6. 게시판 구축

■ 조회

- controllers\readController.js 생성

```
JS board.js    JS readController.js X
controllers > JS readController.js > [?] <unknown>
1  var readModel=require('../models/readModel');
2  var express=require('express');
3
4  module.exports={
5      readData: function (req,res, next){
6          var idx = req.params.idx;
7          readModel.getData(idx,(row)=>{
8              console.log('1개 글 조회 결과 확인 : ',row);
9              res.render('read',{title: "글 조회", row: row[0]});
10             });
11      }
12  }
```



6. 게시판 구축

■ 조회

- models\readModel.js 생성

```
JS board.js JS readModel.js X
models > JS readModel.js > ...
1  var mysql = require('mysql');
2  var connection = mysql.createConnection({
3    connectionLimit: 5,
4    host: 'localhost',
5    user: 'root',
6    password: '1234',
7    database: 'tutorial'
8  });
9
10 module.exports={getData(idx,callback){
11   connection.query('SELECT idx, creator_id, title, content, hit FROM board WHERE idx=?;',idx,(err,row,fields)=>{
12     if(err) throw err;
13     callback(row);
14   });
15 }
16 }
```



6. 게시판 구축

- 조회
- views/wread.ejs 생성

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <script src="/javascripts/jquery-3.6.1.min.js"></script>
5   <title><%= title %></title>
6   <link rel='stylesheet' href='/stylesheets/style.css' />
7 </head>
8 <body>
9   <h1><%= title %></h1>
10
11   <form action="/board/update" method="get">
12     <table border="1">
13       <input type="hidden" name="idx" value="<%=row.idx%>" />
14       <tr>
15         <td>작성자</td>
16         <td><%=row.creator_id%></td>
17       </tr>
18       <tr>
19         <td>제목</td>
20         <td><%=row.title%></td>
21       </tr>
22       <tr>
23         <td>내용</td>
24         <td><%=row.content%></td>
25       </tr>
26       <tr>
27         <td>조회수</td>
28         <td><%=row.hit%></td>
29       </tr>
30       <tr>
31         <td colspan="2">
32           <button type="submit">글 수정</button>
33           <a href="/board">리스트로 돌아가기</a>
34         </td>
35       </tr>
36     </table>
37   </form>
38 </body>
39 </html>
```



6. 게시판 구축

■ 조회

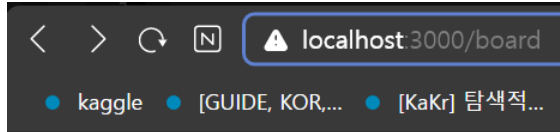
- list.ejs 파일에 글 조회 링크 추가

```
list.ejs board.js package.json index.js index.ejs join.js
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title><%= title %></title>
5     <link rel='stylesheet' href='/stylesheets/style.css' />
6   </head>
7   <body>
8     <h1><%= title %></h1>
9     <a href="/board/write">글쓰기로 이동</a>
10    <br>
11    <br>
12    <table border="1">
13      <tr>
14        <td>번호</td>
15        <td>작성자</td>
16        <td>제목</td>
17        <td>조회수</td>
18      </tr>
19      <%
20        for(var i=0; i<rows.length; i++){
21          var oneItem = rows[i];
22        %>
23        <tr>
24          <td><%=oneItem.idx%></td>
25          <td><%=oneItem.creator id%></td>
26          <td><a href="/board/read/<%=oneItem.idx%>"><%=oneItem.title%></a></td>
27          <td><%=oneItem.hit%></td>
28        </tr>
29      <%
30        }
31      %>
32    </table>
33  </body>
34 </html>
```




6. 게시판 구축

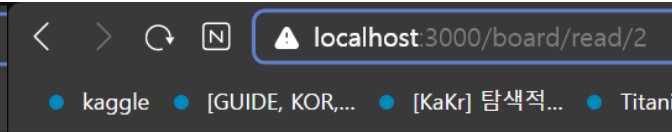
■ 조회 결과 확인



게시판 전체 글 조회

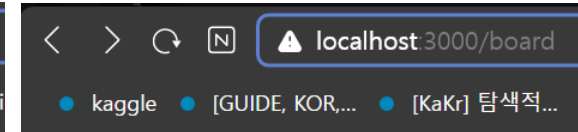
[글쓰기로 이동](#)

번호	작성자	제목	조회수
1	user1	title1	0
2	user2	title2	0
3	cid	1234	0



글 조회

작성자	user2
제목	title2
내용	bye
조회수	0
글 수정	리스트로 돌아가기



게시판 전체 글 조회

[글쓰기로 이동](#)

번호	작성자	제목	조회수
1	user1	title1	0
2	user2	title2	0
3	cid	1234	0



6. 게시판 구축

- 업데이트
 - board.js 파일에 아래 소스 추가

```
board.js
1  var express = require('express');
2  var router = express.Router();
3  var listController = require('../controllers/listController');
4  var writeController = require('../controllers/writeController');
5  var readController = require('../controllers/readController');
6  var updateController = require('../controllers/updateController');
7  const multer = require('multer');
8
9  router.get('/', listController.getListFirst);
10 router.get('/list/:idx', listController.getList);
11 router.get('/write', writeController.writeForm);
12 router.post('/write', writeController.writeData);
13 router.get('/read/:idx', readController.readData);
14 router.get('/update', updateController.updateForm);
15 router.post('/update', multer().none(), (req, res) => {updateController.updateData(req, res)});
16
17 module.exports = router;
```



6. 게시판 구축

■ 업데이트

- controllersWupdateController.js 생성

```
JS board.js  JS updateController.js X
controllers > JS updateController.js > ...
1  var updateModel=require('../models/updateModel');
2  var express=require('express');
3  var url = require('url');
4
5  exports.updateForm=(req,res,next)=>{
6      var queryData = url.parse(req.url, true).query;
7      var idx = queryData.idx;
8      updateModel.getData(idx,(row)=>{
9          console.log('update에서 1개 글 조회 결과 확인 : ',row);
10         res.render('update',{title: "글 수정", row: row[0]});
11     });
12 }
13
14
15
16 exports.updateData=(req,res)=>{
17     var idx = req.body.idx;
18     var creator_id = req.body.creator_id;
19     var title = req.body.title;
20     var content = req.body.content;
21     var passwd = req.body.passwd;
22     var datas = [creator_id, title, content, idx, passwd];
23     console.log("data : " , datas);
24     console.log(JSON.stringify(req.body));
25     updateModel.updateData(datas,(result)=>{
26         if(result.affectedRows == 0){
27             res.send("<script>alert('패스워드가 일치하지 않거나, 잘못된 요청으로 인해 변경되지 않았습니다.');"</script>");
28         }
29         else{
30             res.redirect('/board/read/'+idx);
31         }
32     });
33 }
```



6. 게시판 구축

- 업데이트
 - models\updateModel.js 생성

```
JS board.js ● JS updateModel.js X
models > JS updateModel.js > ...
1  var mysql = require('mysql');
2  var connection = mysql.createConnection({
3    connectionLimit: 5,
4    host: 'localhost',
5    user: 'root',
6    password: '1234',
7    database: 'tutorial'
8  });
9
10 exports.getData=(idx,callback)=>{
11   connection.query('SELECT idx, creator_id, title, content FROM board WHERE idx=?;',idx,(err,row,fields)=>{
12     if(err) throw err;
13     callback(row);
14   });
15 }
16
17 exports.updateData=(datas,callback)=>{
18   var sql="UPDATE board SET creator_id=?, title=?, content=? WHERE idx=? AND passwd=?";
19   connection.query(sql,datas,function(err,result){
20     if(err) console.error("글 수정 중 에러 발생 err : " +err);
21     callback(result);
22   });
23 }
```



6. 게시판 구축

■ 업데이트

- views\wupdate.ejs 생성

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <script src="/javascripts/jquery-3.6.1.min.js"></script>
5     <title><%= title %></title>
6     <link rel='stylesheet' href='/stylesheets/style.css'/>
7   </head>
8   <body>
9     <h1><%= title %></h1>
10    <form action="/board/update" method="post" enctype="multipart/form-data" onsubmit="return onWriteSubmit()">
11      <table border="1">
12        <input type="hidden" name="idx" value="<%=row.idx%>"/>
13        <tr>
14          <td>작성자</td>
15          <td><input type="text" name="creator_id" id="creator_id" value="<%=row.creator_id%>" required/></td>
16        </tr>
17        <tr>
18          <td>제목</td>
19          <td><input type="text" name="title" id="title" value="<%=row.title%>" required/></td>
20        </tr>
21        <tr>
22          <td>내용</td>
23          <td><textarea type="content" name="content" cols="30" rows="10" required><%=row.content%></textarea></td>
24        </tr>
25        <tr>
26          <td>패스워드</td>
27          <td><input type="password" name="passwd" id="passwd" required/></td>
28        </tr>
29        <tr>
30          <td colspan="2">
31            <button type="submit">글쓰기</button>
32          </td>
33        </tr>
34      </table>
35    </form>
```



6. 게시판 구축

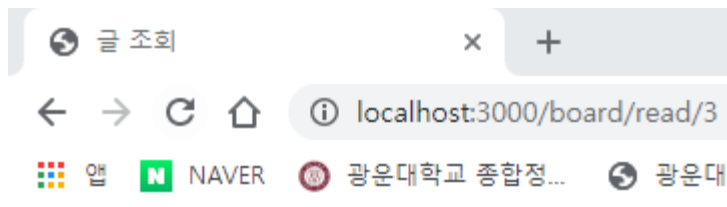
■ 업데이트

```
36      <script>
37          function onWriteSubmit(){
38              if ($("#creator_id").val().trim() == ""){
39                  var message = "아이디를 입력해 주세요.";
40                  $("#creator_id").val("");
41                  $("#creator_id").focus();
42                  alert(message);
43                  return false;
44              }
45              if ($("#title").val().trim() == ""){
46                  var message = "제목을 입력해 주세요.";
47                  $("#title").val("");
48                  $("#title").focus();
49                  alert(message);
50                  return false;
51              }
52              if ($("#content").val().trim() == ""){
53                  var message = "본문 내용을 입력하세요.";
54                  $("#content").val("");
55                  $("#content").focus();
56                  alert(message);
57                  return false;
58              }
59              if ($("#passwd").val().trim() == ""){
60                  var message = "패스워드를 입력하세요.";
61                  $("#passwd").val("");
62                  $("#passwd").focus();
63                  alert(message);
64                  return false;
65              }
66          }
67      </script>
68  </body>
69  </html>
```



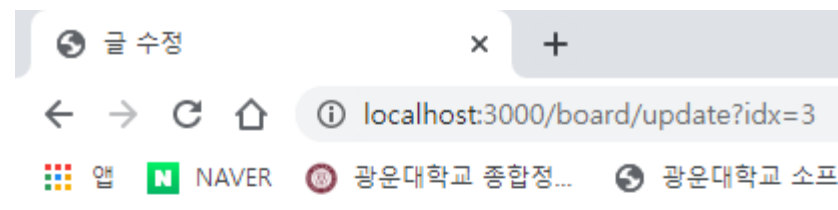
6. 게시판 구축

- 업데이트 조회
 - 수정 화면 확인



글 조회

작성자	user3
제목	글쓰기테스트
내용	테스트 테스트
조회수	0
글 수정	리스트로 돌아가기



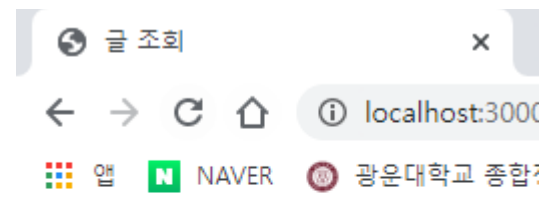
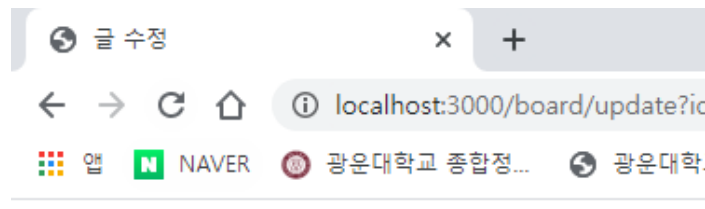
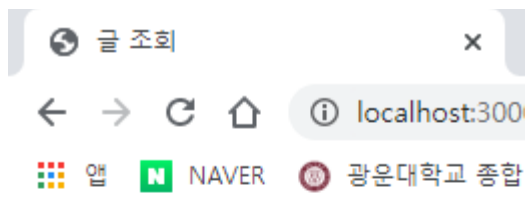
글 수정

작성자	<input type="text" value="user3"/>
제목	<input type="text" value="글쓰기테스트"/>
내용	<div>테스트 테스트</div>
패스워드	<input type="password"/>
<input type="button" value="글쓰기"/>	



6. 게시판 구축

- 업데이트 쓰기
 - 업데이트 결과 확인



글 조회

작성자	user3
제목	글쓰기테스트
내용	테스트 테스트
조회수	0
글 수정	리스트로 돌아가기

글 수정

작성자	user3
제목	글쓰기테스트
내용	수정 수정
패스워드
글쓰기	

글 조회

작성자	user3
제목	글쓰기테스트
내용	수정 수정
조회수	0
글 수정	리스트로 돌아가기



MySQL 기본

- show databases
 - 데이터베이스 리스트 확인

```
C:\Program Files\MySQL\MySQL Server 5.5\bin> mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sakila |
| sys |
| world |
+-----+
6 rows in set (0.00 sec)

mysql>
```



MySQL 기본

- create database [DB명]
 - 데이터베이스를 선택

```
C:\Program Files\MySQL\bin> mysql> create database basic;
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| basic |
| mysql |
| performance_schema |
| sakila |
| sys |
| world |
+-----+
7 rows in set (0.00 sec)
```



MySQL 기본

- use [DB명]
 - 데이터베이스를 선택



C:\WP

```
mysql> use basic;  
Database changed  
mysql>
```



MySQL 기본

- create table
 - 테이블 생성



C:\Program Files\MySQL\MySQL

```
mysql> create table sample(  
  -> item varchar(20),  
  -> price int);  
Query OK, 0 rows affected (0.04 sec)  
  
mysql>
```



MySQL 기본

- show tables
 - 테이블 리스트 출력

```
C:\WProgram
mysql> show tables;
+-----+
| Tables_in_basic |
+-----+
| sample          |
+-----+
1 row in set (0.00 sec)

mysql>
```



MySQL 기본

- desc [table명]
 - 테이블 속성 확인

```
C:\Program Files\MySQL\MySQL Server 5.7\bin
mysql> desc sample;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| item  | varchar(20)   | YES  |     | NULL    |       |
| price | int(11)       | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```



MySQL 기본

- insert
 - 데이터 삽입



C:\Program Files\MySQL\MySQL Server 5.7\bin\mysql>

```
mysql> insert into sample(item, price) value('사과', 2000);  
Query OK, 1 row affected (0.00 sec)
```



MySQL 기본

■ select

- 데이터 출력

C:\Program Files\MySQL

```
mysql> select * from sample;
```

```
+-----+-----+  
| item  | price |  
+-----+-----+  
| 사과  | 2000  |  
+-----+-----+
```

```
1 row in set (0.00 sec)
```

```
mysql> select item from sample;
```

```
+-----+  
| item  |  
+-----+  
| 사과  |  
+-----+
```

```
1 row in set (0.00 sec)
```




MySQL 기본

- select ... where
 - 데이터 조건 출력

```
C:\Program Files\MySQL\MySQL Ser
mysql> select * from sample where item='사과';
+-----+-----+
| item  | price |
+-----+-----+
| 사과  | 2000  |
+-----+-----+
1 row in set (0.00 sec)

mysql> select * from sample where item='포도';
Empty set (0.00 sec)
```

MySQL 기본

- update
 - 데이터 수정

```
C:\Program Files\MySQL\MySQL Server 5.7\bin
mysql> select * from sample where item='사과';
+-----+-----+
| item  | price |
+-----+-----+
| 사과  | 2000  |
+-----+-----+
1 row in set (0.00 sec)

mysql> update sample set price=3000 where item='사과';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from sample where item='사과';
+-----+-----+
| item  | price |
+-----+-----+
| 사과  | 3000  |
+-----+-----+
1 row in set (0.00 sec)
```



MySQL 기본

- delete
 - 데이터 삭제

```
C:\Program Files\MySQL\MySQL
mysql> select * from sample;
+-----+-----+
| item  | price |
+-----+-----+
| 사과  | 3000  |
| 포도  | 1500  |
+-----+-----+
2 rows in set (0.00 sec)

mysql> delete from sample where item='사과';
Query OK, 1 row affected (0.01 sec)

mysql> select * from sample;
+-----+-----+
| item  | price |
+-----+-----+
| 포도  | 1500  |
+-----+-----+
1 row in set (0.00 sec)
```