



사용자 인증을 통한 게시판 만들기

# |INTRODUCE



**김시훈**

- 세종대학교 컴퓨터공학과
- shuni93@naver.com



**황성영**

- 광운대학교 컴퓨터공학과
- sy930503@naver.com

# |INTRODUCE

강의자료

[https://github.com/shuni4481/node\\_note](https://github.com/shuni4481/node_note)

# |INDEX

- ▶ #1 - Node.js의 소개 및 환경설정 / 간단한 웹 애플리케이션 만들기
- ▶ #2 - Express 모듈을 통한 프로젝트 생성 / Route / rest API
- ▶ #3 - Database(Mysql) 연동 / 회원가입 & 로그인
- ▶ #4 - CRUD 게시판 구현

|LEARN

DataBase ?

여러 사람에 의해 공유되어 사용될 목적으로 통합하여 관리되는 데이터의 집합

자료를 구조화 하여 저장함으로써 검색과 갱신 등 효율을 높임

DBMS(Database Management System)가 관리를 함

|LEARN

Database



ORACLE®



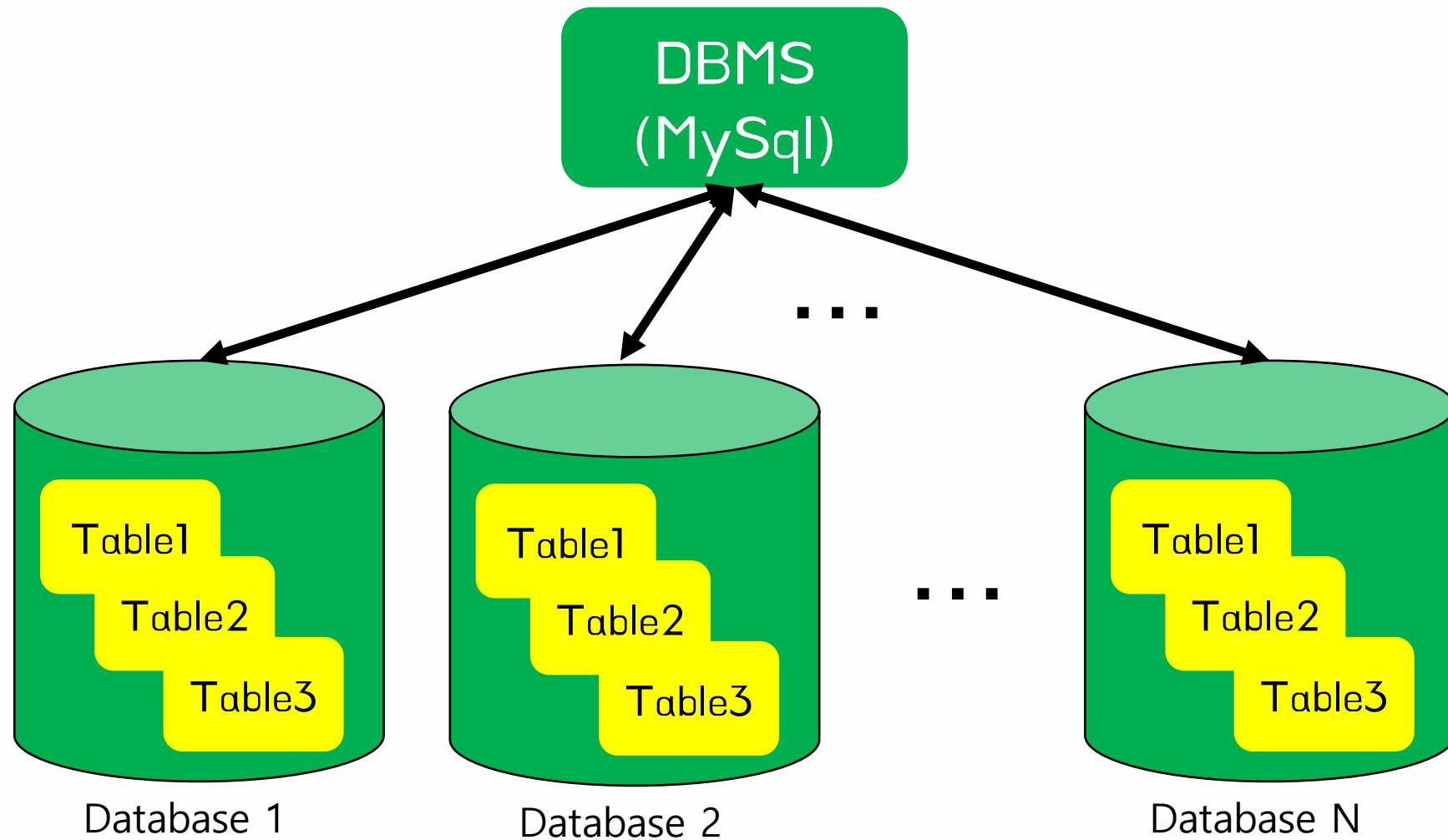
RDBMS



NoSQL

# |LEARN

Database



The diagram illustrates the structure of a database table. A red bracket on the left groups the first two rows as '행(row)', '튜플(tuple)', and '레코드(record)'. Another red bracket at the top groups the four columns as '열(column)', '필드(field)', and '속성(attribute)'. The table itself has four columns: '학번' (Student ID), '이름' (Name), '나이' (Age), and '학과' (Major). The data rows are: Row 1 (tuple/record) contains 학번 2013237292, 이름 김모씨, 나이 27, and 학과 컴퓨터공학과; Row 2 contains 학번 2017230323, 이름 이모씨, 나이 23, and 학과 전자공학과.

학번	이름	나이	학과
2013237292	김모씨	27	컴퓨터공학과
2017230323	이모씨	23	전자공학과

테이블(table)

Primary Key(주 키)

대학 코드	대학 이름	위치
K048	광운대학교	서울 노원구
K049	세종대학교	서울 광진구

대학교 Table

Primary Key(주 키) Foreign Key(외래 키)

학생 코드	대학 코드	이름	나이
S001	K048	황성영	55
S002	K049	김시훈	55

학생 Table

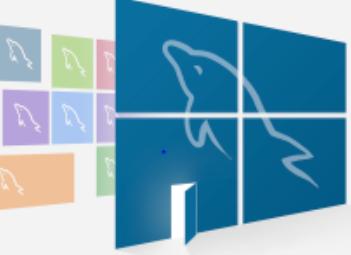
<https://dev.mysql.com/downloads/mysql/>  
(컴퓨터 환경에 맞게)

Recommended Download:

**MySQL Installer**  
for Windows

All MySQL Products. For All Windows Platforms.  
In One Package.

Starting with MySQL 5.6 the MySQL Installer package replaces the standalone MSI packages.



Windows (x86, 32 & 64-bit), MySQL Installer MSI

[Go to Download Page >](#)

Other Downloads:

Windows (x86, 64-bit), ZIP Archive

(mysql-8.0.15-winx64.zip)

8.0.15

184.1M

[Download](#)

MD5: 75f d5cd2676209550a8b9bac4fd0a8b6 | Signature

Window -> MySQL Command Line Client 실행

Mac -> terminal에 mysql -u root -p 명령어

설치 할 때 입력한 비밀번호 입력

```
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.7.19-log MySQL Community Server (GPL)

Copyright (c) 2000, 2017, 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>
```

```
mysql> create database testDB default character set utf8;
```

```
Query OK, 1 row affected (0.00 sec)
```

[testDB라는 Database 생성]

```
mysql> show databases;  
[Database 조회]
```

```
mysql> drop database testDB;  
Query OK, 0 rows affected (0.00 sec)  
[Database 삭제]
```

```
mysql> use testDB;
```

Database changed

[Database 사용]

Table명

```
mysql> create table 'info'(  
    'idx' int(11) not null auto_increment,  
    'id' varchar(50) not null,  
    'pw' varchar(50) not null,  
    primary key ('idx')  
)
```

collate='utf8\_general\_ci'; //한글 깨짐 방지

Query OK, 0 rows affected (0.05 sec)

[Table 생성]

```
mysql> use testDB;  
Database changed
```

[Database 사용]

```
mysql> insert into info ('id', 'pw') values ('까치', '1234');  
Query OK, 0 rows affected (0.05 sec)
```

[Column 생성]

```
mysql> select * from info;  
mysql> select 'id', 'pw' from info;  
mysql> select * from info where id='까치';
```

```
mysql> update info set pw='123456' where id='까치';  
[Data 수정]
```

```
mysql> delete from info where id='까치';  
[Data 삭제]
```

```
mysql> desc info;  
[info Table 속성 조회]
```

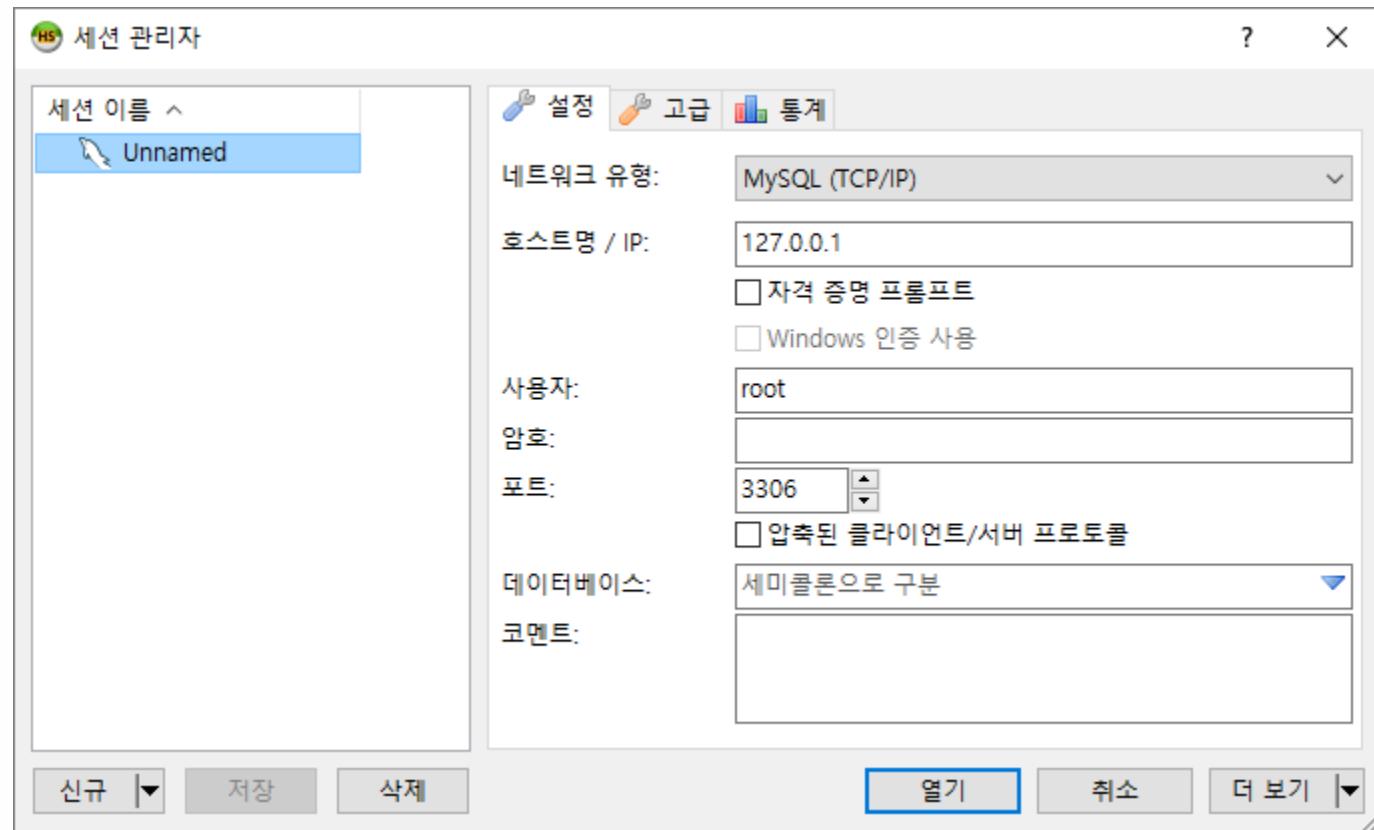
```
mysql> drop table info;  
[info Table 삭제]
```



## 데이터베이스 관리 도구

<https://www.heidisql.com/download.php>

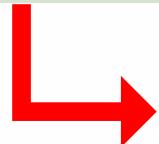
설치 후 실행 -> 신규 클릭 -> 암호입력 -> 열기



# LEARN

## UserTable 생성

no	id	password	name	email	address
INT(11) AUTO_INCREMENT	VARCHAR(50)	VARCHAR(50)	VARCHAR(50)	VARCHAR(50)	VARCHAR(50)



Primary key(주 키) ->자동 증가 옵션  
ex) 1, 2, 3, 4, 5 ...

```
mysql > create database note; //note라는 database 생성
CREATE TABLE `user` (
  `no` INT(11) NOT NULL AUTO_INCREMENT,
  `id` VARCHAR(50) NOT NULL,
  `password` VARCHAR(128) NOT NULL,
  `name` VARCHAR(50) NOT NULL,
  `email` VARCHAR(50) NOT NULL,
  `address` VARCHAR(50) NOT NULL,
  PRIMARY KEY (`no`) //주 키를 no로 설정
)COLLATE = 'utf8_general_ci' //한글 깨짐 방지
```

# LEARN

## UserTable



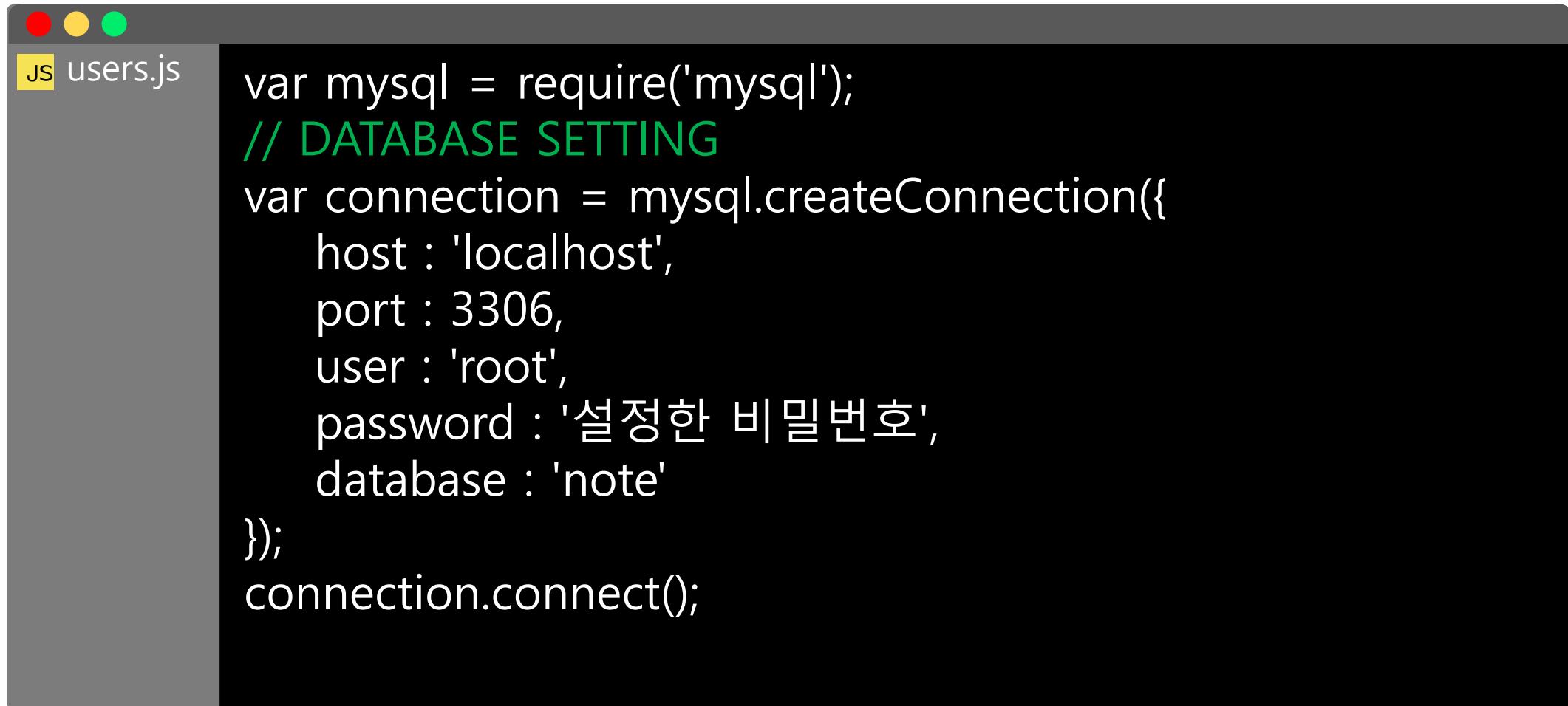
```
mysql > insert into user (id, password, name, email, address)
values('test', 'test', 'test', 'test', '서울');
mysql > select * from user;
```

no	id	password	name	email	address
1	test	test	test	test	서울



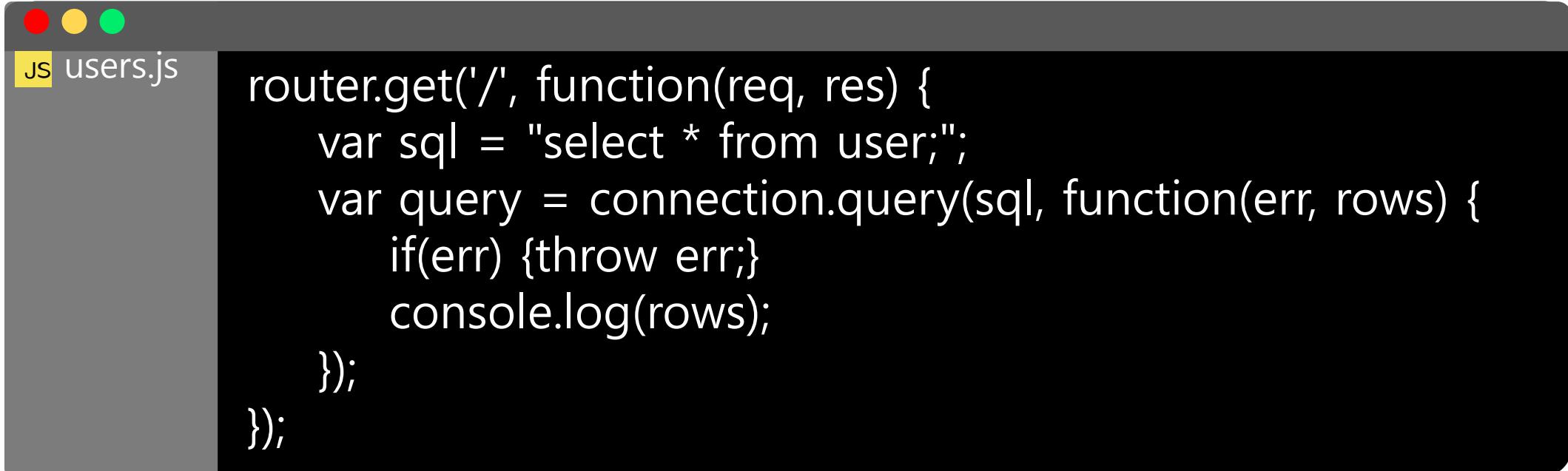
```
$npm install mysql
```

mysql 모듈 설치



The image shows a screenshot of a code editor window. The title bar has three colored circles (red, yellow, green) and the file name "users.js". The code editor displays the following JavaScript code:

```
var mysql = require('mysql');
// DATABASE SETTING
var connection = mysql.createConnection({
    host : 'localhost',
    port : 3306,
    user : 'root',
    password : '설정한 비밀번호',
    database : 'note'
});
connection.connect();
```



The image shows a screenshot of a code editor window. On the left, there's a dark sidebar with three circular icons (red, yellow, green) at the top, followed by a yellow folder icon labeled "users.js". The main area is a black code editor window containing the following JavaScript code:

```
router.get('/', function(req, res) {
  var sql = "select * from user;";
  var query = connection.query(sql, function(err, rows) {
    if(err) {throw err;}
    console.log(rows);
  });
});
```

## Mysql 모듈 설치 & 연결 후 /users에 전체 user정보 띄우기

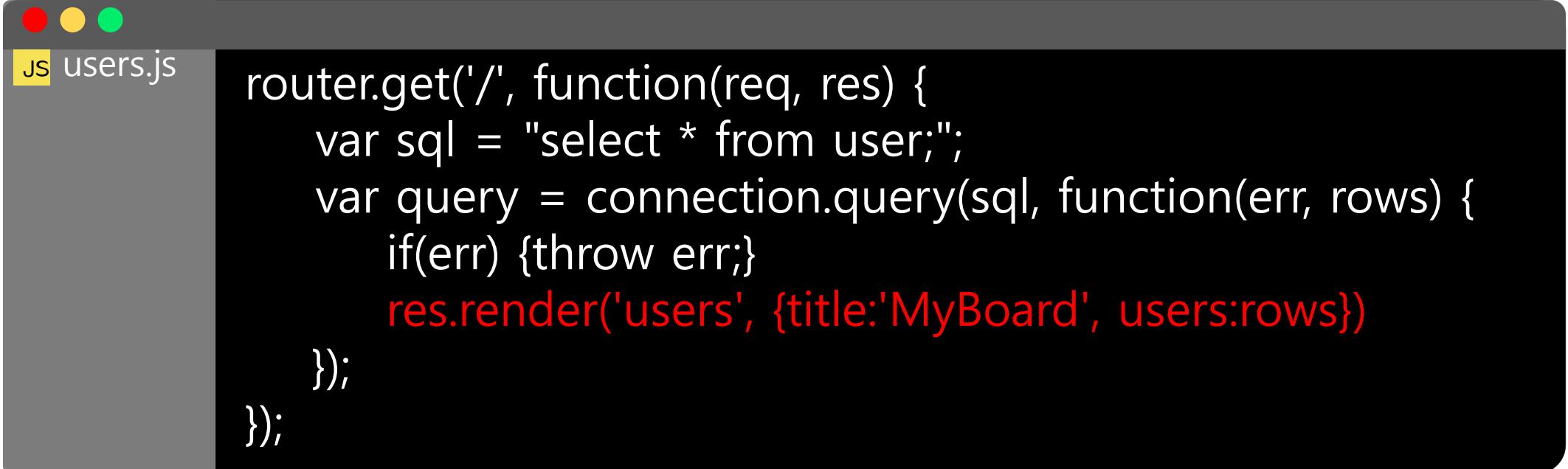


users.ejs파일에서

```
<%for(var i = 0; i < users.length; i++) {%
<%}%>
```

사용

No	아이디	비밀번호	이름	Email	주소
1	a	a	a	a@a.com	서울
2	test	test	test	test	서울
3	b	b	황성영	b@b.com	b

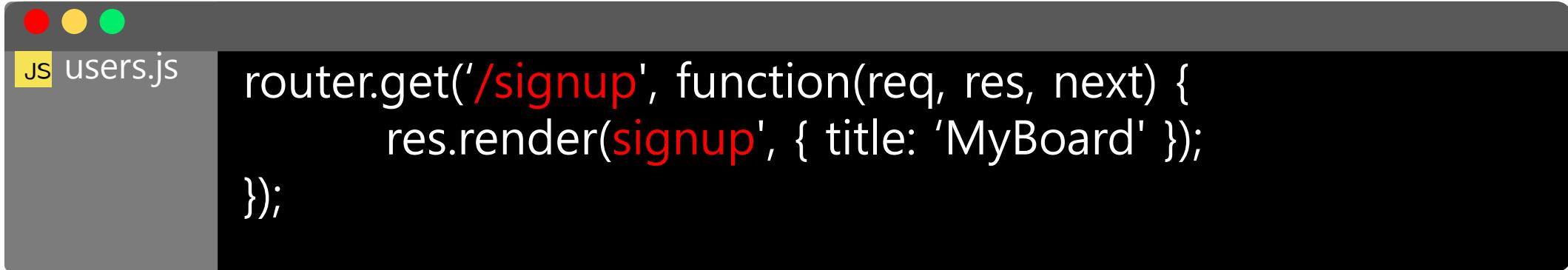


The screenshot shows a code editor window with a dark theme. In the top left corner, there are three small colored circles (red, yellow, green). Below them, a yellow tab bar displays the file name "users.js". The main code area contains the following JavaScript code:

```
router.get('/', function(req, res) {
  var sql = "select * from user;";
  var query = connection.query(sql, function(err, rows) {
    if(err) {throw err;}
    res.render('users', {title:'MyBoard', users:rows})
  });
});
```

# | LEARN

## 회원가입 route



```
JS users.js
router.get('/signup', function(req, res, next) {
  res.render(signup, { title: 'MyBoard' });
});
```

[https://github.com/shuni4481/node\\_note/blob  
/master/views/signup.ejs](https://github.com/shuni4481/node_note/blob/master/views/signup.ejs)

views에 includes폴더 생성

```
<%include ./includes/header.ejs %>
<%include ./includes/footer.ejs %>
```

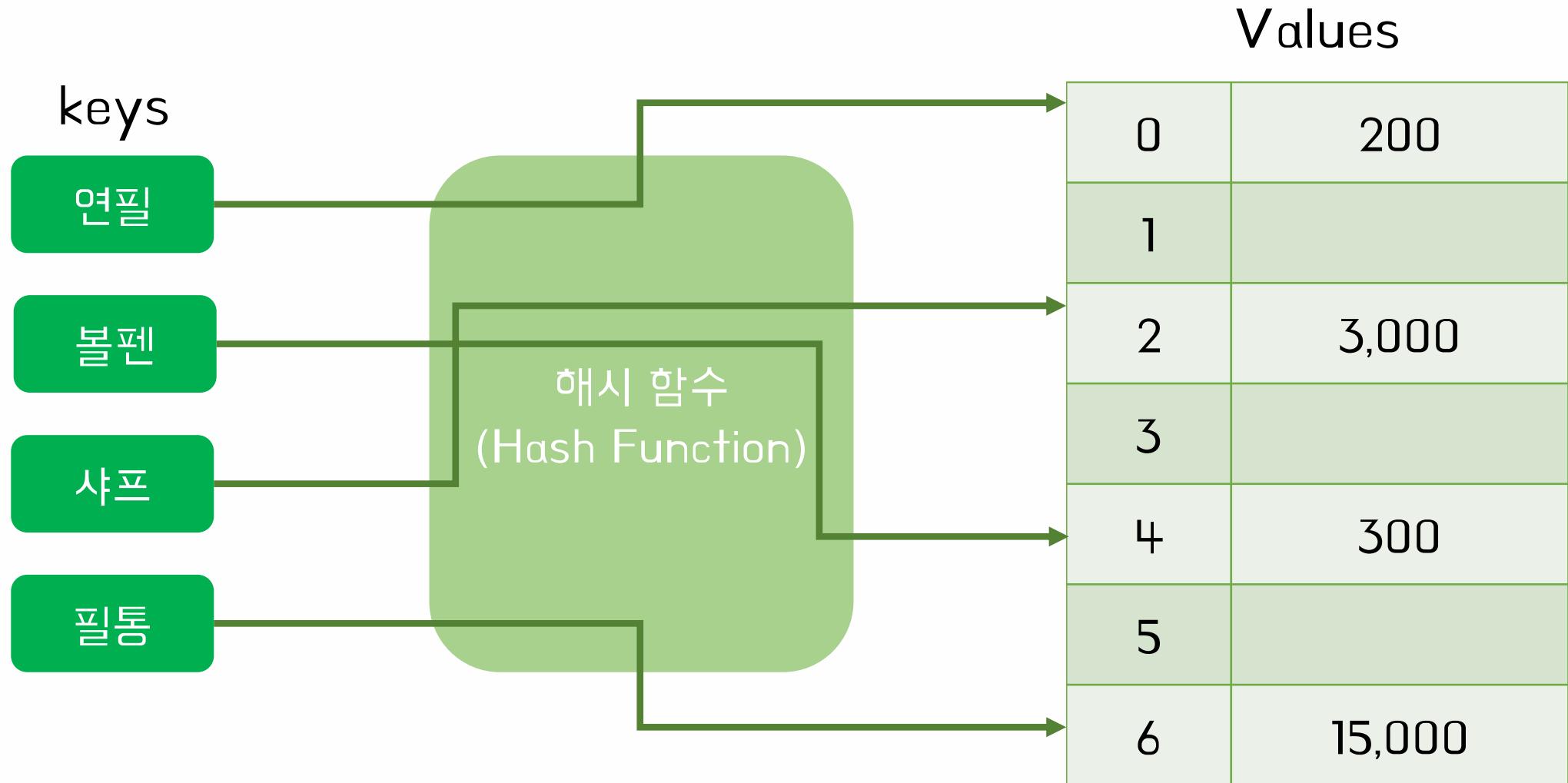


JS users.js

```
router.post('/signup', function (req, res) {
  var id = req.body.id;
  var password = req.body.password;
  var name = req.body.name;
  var email = req.body.email;
  var address = req.body.address;
  var sql = "insert into user (id, password, name, email, address) values("
  + id + "','" + password + "','" + name + "','" +email+ "','" +address +
  ")";
  var query = connection.query(sql, function (err, rows) {
    if (err) {
      throw err;
    }
    console.log("Data inserted!");
    res.redirect('/');
  });
});
```

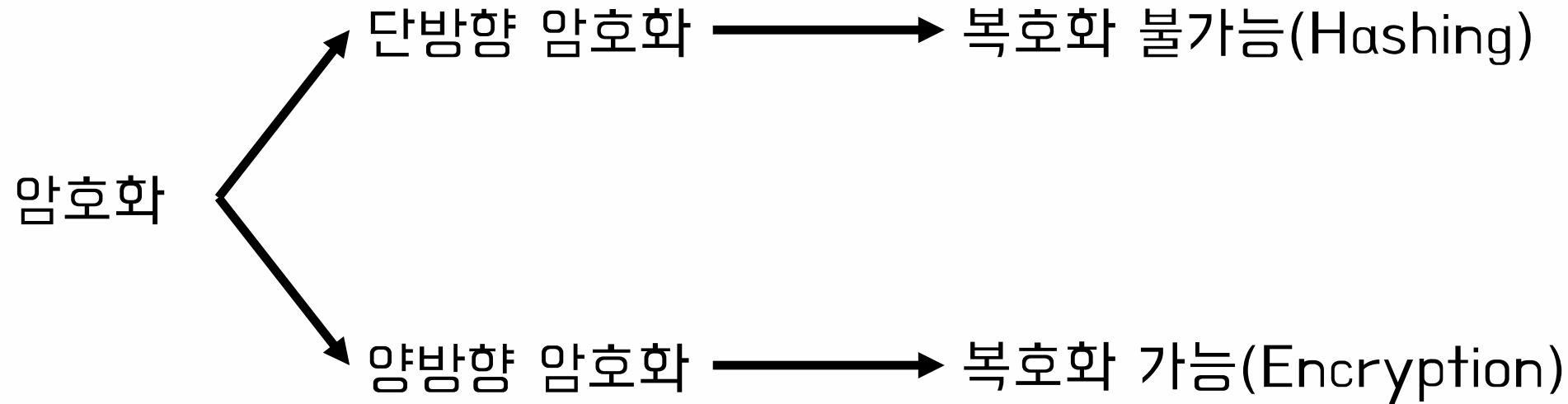
# LEARN

암호화



# | LEARN

암호화



## SHA 알고리즘

1993년에 [미국 표준 기술 연구소](#)(NIST)에 의해 안전한 해시 표준  
 (Secure Hash Standard, [FIPS](#) PUB 180)으로 출판

알고리즘	해시값 크기	내부 상태 크기	블록 크기	길이 한계	워드 크기	과정 수	사용되는 연산	충돌
SHA-0	160	160	512	64	32	80	+ , and, or, xor, rotl	발견됨
SHA-1	160	160	512	64	32	80	+ , and, or, xor, rotl	발견됨
SHA-256 /224	256/224	256	512	64	32	64	+ , and, or, xor, shr, rotr	-
SHA-512/384	512/384	512	1024	128	64	80	+ , and, or, xor, shr, rotr	-



```
$npm install crypto
```

crypto 모듈 설치



JS users.js

```
var crypto = require('crypto');
var hashpass =
crypto.createHash("sha512").update(password).digest("hex");
```

crypto 모듈을 사용하여 회원가입 & 로그인 비밀번호 부분 처리



No	아이디	비밀번호	이름	Email	주소
1	a	a	a	a@a.com	서울
2	test	test	test	test	서울
3	b	b	황성영	b@b.com	b
4	abc	ddaf35a193617abacc417349ae20413112e6fa4e89a97ea20a9eeee64b55d39a2192992a274fc1a836ba3c23a3feebbd454d4423643ce80e2a9ac94fa54ca49f	abc	abc@abc.com	abc



JS users.js

```
router.post('/signup', function (req, res) {
  var id = req.body.id;
  var password = req.body.password;
  var hashpass = crypto.createHash("sha512").update(password).digest("hex");
  var name = req.body.name;
  var email = req.body.email;
  var address = req.body.address;
  var sql = "insert into user (id, password, name, email, address) values('" +
    id + "','" + hashpass + "','" + name + "','" + email + "','" + address + "')";
  var query = connection.query(sql, function (err, rows) {
    if (err) {
      throw err;
    }
    console.log("Data inserted!");
    res.redirect('/');
  });
});
```

정보를 유지하고 요청을 보낼 때 참조  
ex) 로그인, 장바구니, 자동로그인...

정보가 클라이언트에 저장됨

정보가 남아있음

누구나 접근 가능

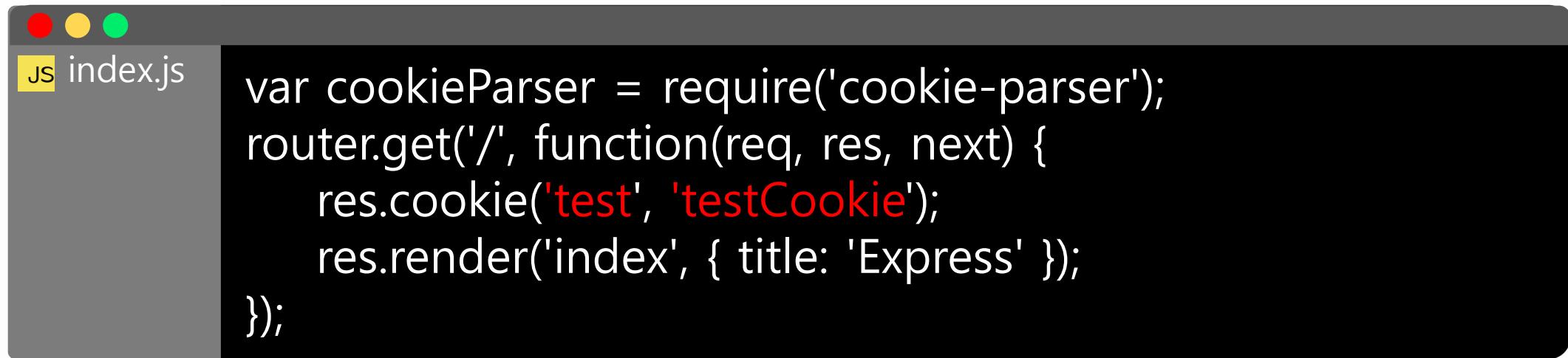
정보가 서버에 저장됨

클라이언트는 세션 정보만 가짐

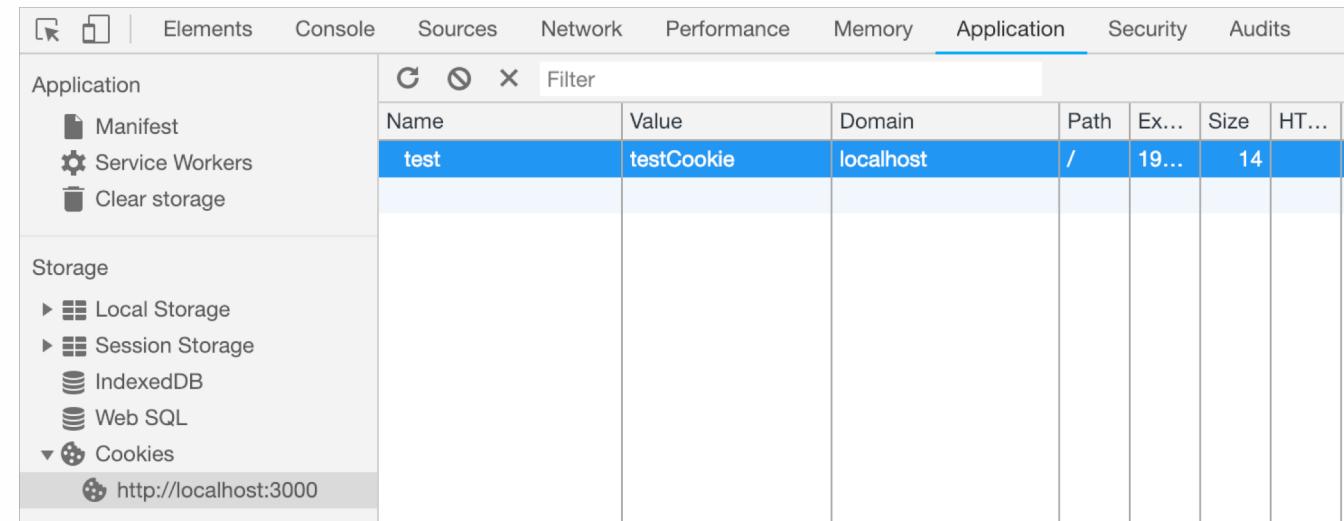
브라우저 종료 시 / 서버 세션 만료 시 정보 없어짐

# LEARN

## Cookie



```
index.js
var cookieParser = require('cookie-parser');
router.get('/', function(req, res, next) {
  res.cookie('test', 'testCookie');
  res.render('index', { title: 'Express' });
});
```



The screenshot shows the Chrome DevTools Application tab open. On the left, there's a sidebar with sections for Application (Manifest, Service Workers, Clear storage), Storage (Local Storage, Session Storage, IndexedDB, Web SQL, Cookies), and Network. The Cookies section is expanded, showing a list of cookies for the domain http://localhost:3000. One cookie is selected: 'test' with the value 'testCookie'. The main pane displays a table with columns: Name, Value, Domain, Path, Ex..., Size, HT..., and S... (partially visible). The 'test' row is highlighted in blue.

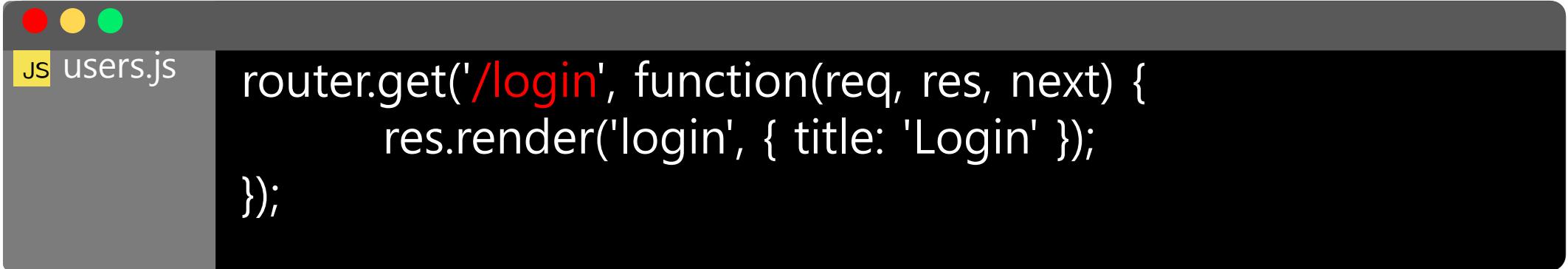
Name	Value	Domain	Path	Ex...	Size	HT...	S...
test	testCookie	localhost	/	19...	14		

The screenshot shows the Chrome DevTools interface with the 'Application' tab selected. On the left, there's a sidebar with 'Manifest', 'Service Workers', and 'Clear storage' options under 'Application'. Below that is a 'Storage' section with 'Local Storage', 'Session Storage', 'IndexedDB', 'Web SQL', and 'Cookies' sections. The 'Cookies' section is expanded, and a cookie named 'connect.sid' is listed. The table has columns for Name, Value, Domain, Path, Expires, Size, and HT... (HTTP). The 'Value' column for 'connect.sid' contains a long string of characters starting with 's%3AFt-vwHRx2jtwVXz8W...'. The 'Domain' column shows 'localhost', and the 'Path' column shows '/'. The 'Expires' column shows '19...', 'Size' is '93', and 'HT...' has a checked checkbox.

Name	Value	Domain	Path	Ex...	Size	HT...
connect.sid	s%3AFt-vwHRx2jtwVXz8W...	localhost	/	19...	93	✓

# | LEARN

로그인 route



```
JS users.js
router.get('/login', function(req, res, next) {
    res.render('login', { title: 'Login' });
});
```

[https://github.com/shuni4481/node\\_note/blob  
/master/views/login.ejs](https://github.com/shuni4481/node_note/blob/master/views/login.ejs)

```
$npm install passport  
$npm install passport-local  
$npm install connect-flash  
$npm install express-session
```

passport 모듈 & express-session 설치



JS app.js

```
var passport = require('passport');
var flash = require('connect-flash');
var session = require('express-session');

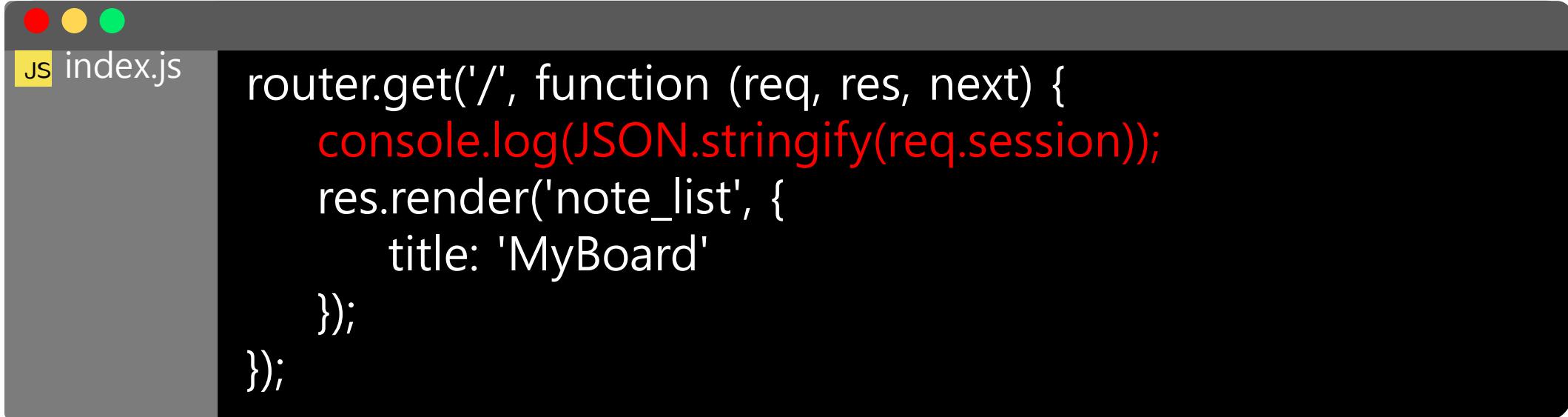
app.use(session({
  secret: 'keyboard cat',
  resave: false,
  saveUninitialized: true
}));

app.use(flash());
app.use(passport.initialize());
app.use(passport.session());
```

| LEARN

로그인 route





The image shows a screenshot of a code editor window. The title bar has three colored dots (red, yellow, green) and the file name "index.js". The code in the editor is:

```
router.get('/', function (req, res, next) {
  console.log(JSON.stringify(req.session));
  res.render('note_list', {
    title: 'MyBoard'
  });
});
```

로그인시 세션이 잘 생성되는지 확인

```
$npm start
```

```
> test@0.0.0 start /Users/shuni/node/test_js/test_js
```

```
> node ./bin/www
```

//처음 접속 후

```
{"cookie": {"originalMaxAge": null, "expires": null, "httpOnly": true, "path": "/"}}
```

//로그인 후

```
{"cookie": {"originalMaxAge": null, "expires": null, "httpOnly": true, "path": "/"}, "passport": {"user": {"id": "c", "name": "c", "no": 11}}}
```

//로그아웃 후

```
{"cookie": {"originalMaxAge": null, "expires": null, "httpOnly": true, "path": "/"}, "passport": {}}
```

## 로그인 했을 시

1. req.session.passport 체크 & req.session.passport.no체크 후 로그인 버튼은 로그아웃으로 바꾸기
2. 로그인 사용자(passport.no가 존재하는 사용자)가 /login url로 접속했을시 홈 화면으로 redirect 시키기
3. /users/logout으로 GET형식으로 요청을 하면 req.logout() 를 통해 session 삭제하는 route구현

---

THANK YOU