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

INDEX

LEARN

DataBase ?

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

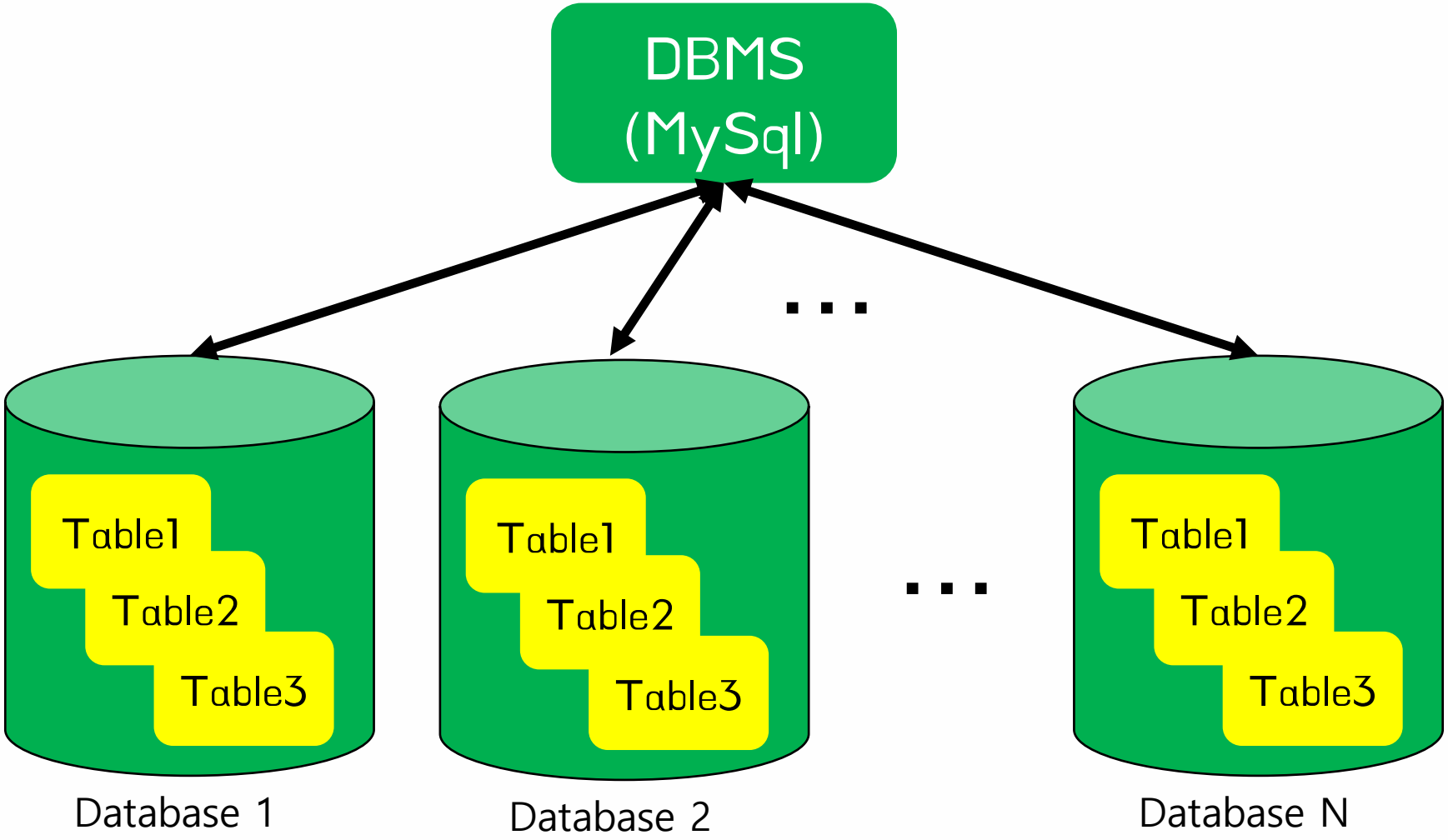
LEARN

Database

RDBMS

NoSQL

Database


DBMS
(MySql)
Table1
Table2
Table3
Table1
Table2
Table3
Table1
Table2
Table3
Database 1
Database 2
Database N

열(column)
필드(field)
속성(attribute)

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

행(row)
튜플(tuple)
레코드(record)

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)

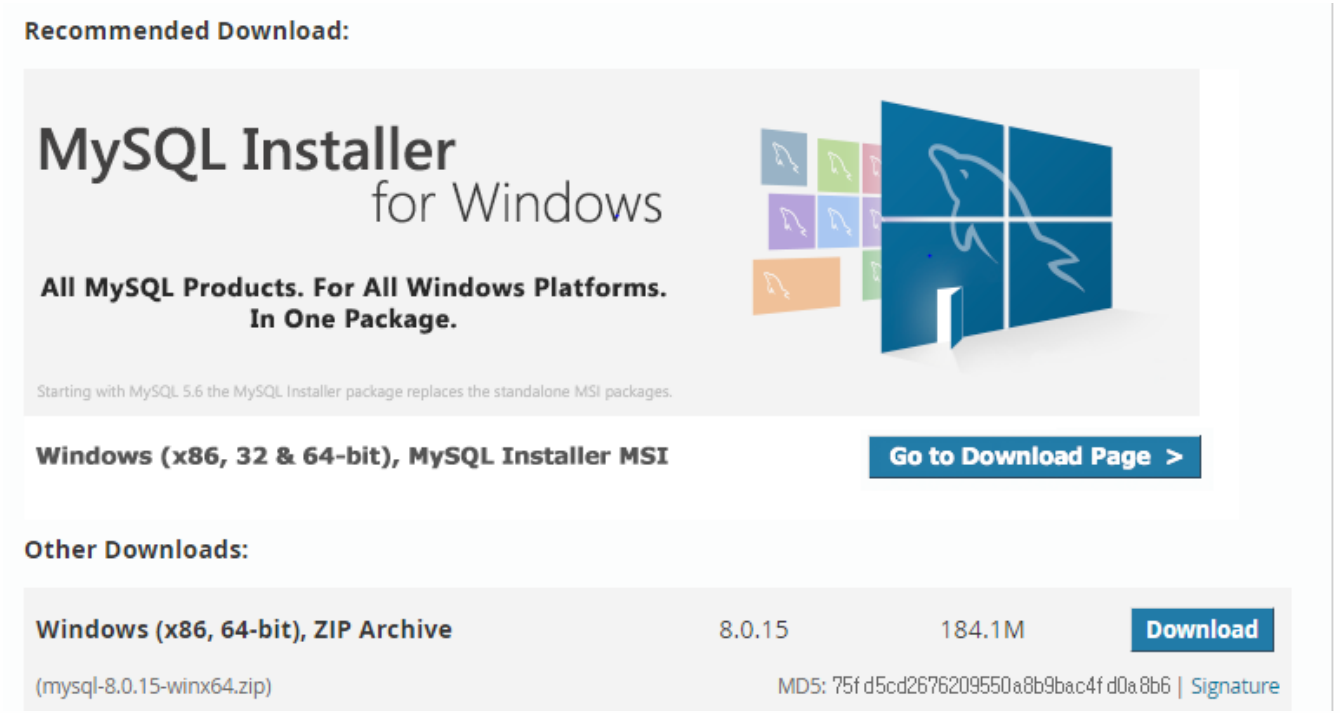
학생 코드	대학 코드	이름	나이
S001	K048	황성영	55
S002	K049	김시훈	55

학생 Table

MYSQL 설치 및 세팅

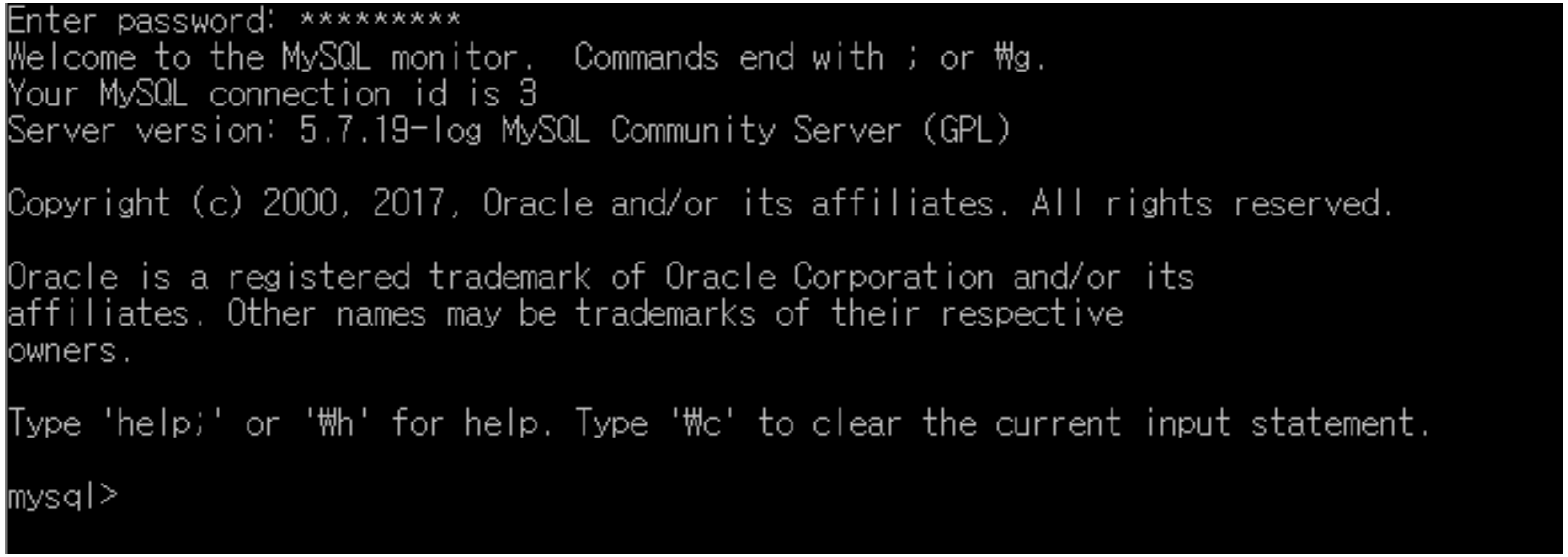
https://dev.mysql.com/downloads/mysql/

(컴퓨터 환경에 맞게)



MYSQL 설치 및 세팅

Window -> MySQL Command Line Client 실행
Mac -> terminal에 mysql -u root -p 명령어
설치 할 때 입력한 비밀번호 입력



Database 생성

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

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

[testDB라는 Database 생성]

Database 조회 및 삭제

```
mysql> show databases;
```

[Database 조회]

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

[Database 삭제]

Table 생성

```
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]

Data 입력

```
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]

Data 조회

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

Data 수정 및 삭제

```
mysql> update info set pw='123456' where id='까치';
```

[Data 수정]

```
mysql> delete from info where id='까치;
```

[Data 삭제]

Table 속성 조회 및 삭제

```
mysql> desc info;
```

[info Table 속성 조회]

```
mysql> drop table info;
```

[info Table 삭제]

HeidiSql

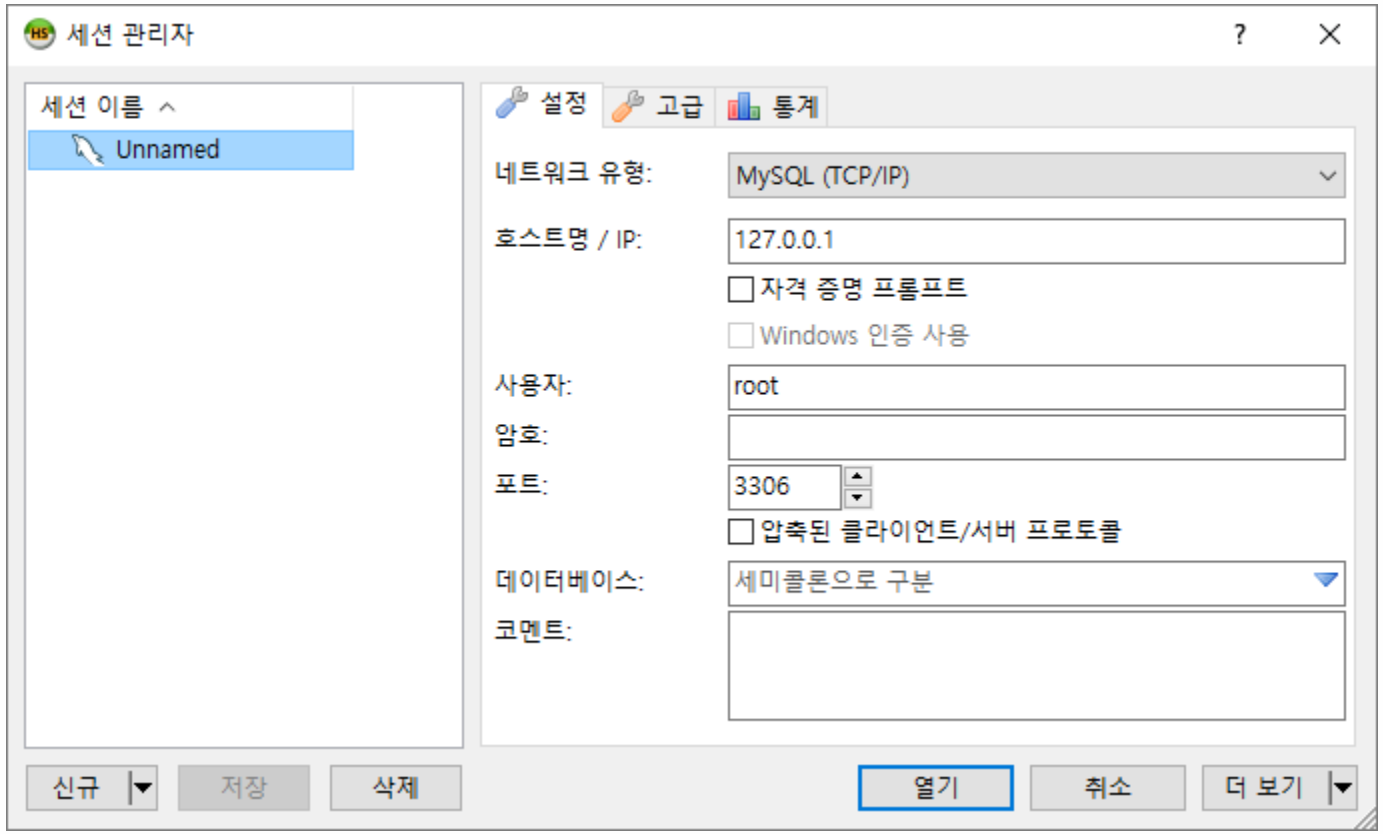
데이터베이스 관리 도구

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

설치 후 실행 -> 신규 클릭 -> 암호입력 -> 열기

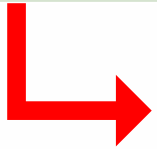
LEARN

HeidiSql



UserTable 생성

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



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

UserTable 생성

```
mysql > create database note; //note라는 database생성
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
```

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

Mysql 연결

mysql 모듈 설치

Mysql 연결

JS users.js

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

Mysql 연결

JS users.js

```
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);
    });
});
```

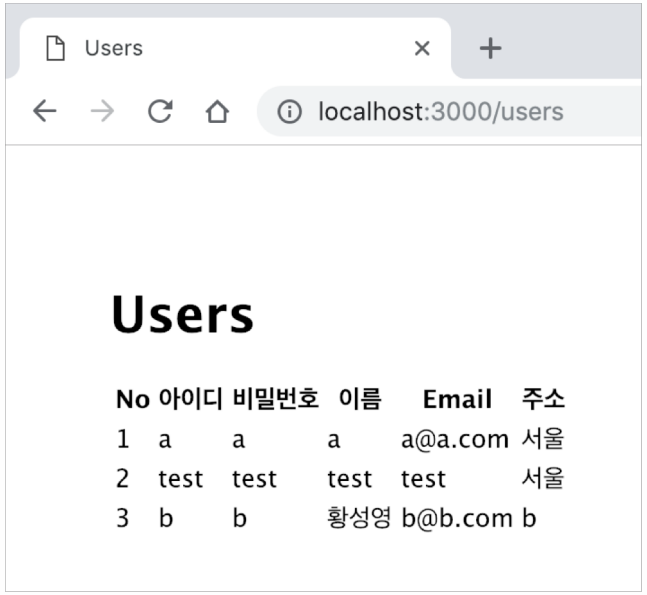
DOIT

Mysql 모듈 설치 & 연결 후 /users에 전체 user정보 띄우기

users.ejs파일에서

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

사용



Mysql 연결

JS users.js

```
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' });
});
```

회원가입 view

https://github.com/shuni4481/node_note/blob/master/views/signup.ejs

LEARN

Header & Footer 모듈화

views에 includes폴더 생성

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

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

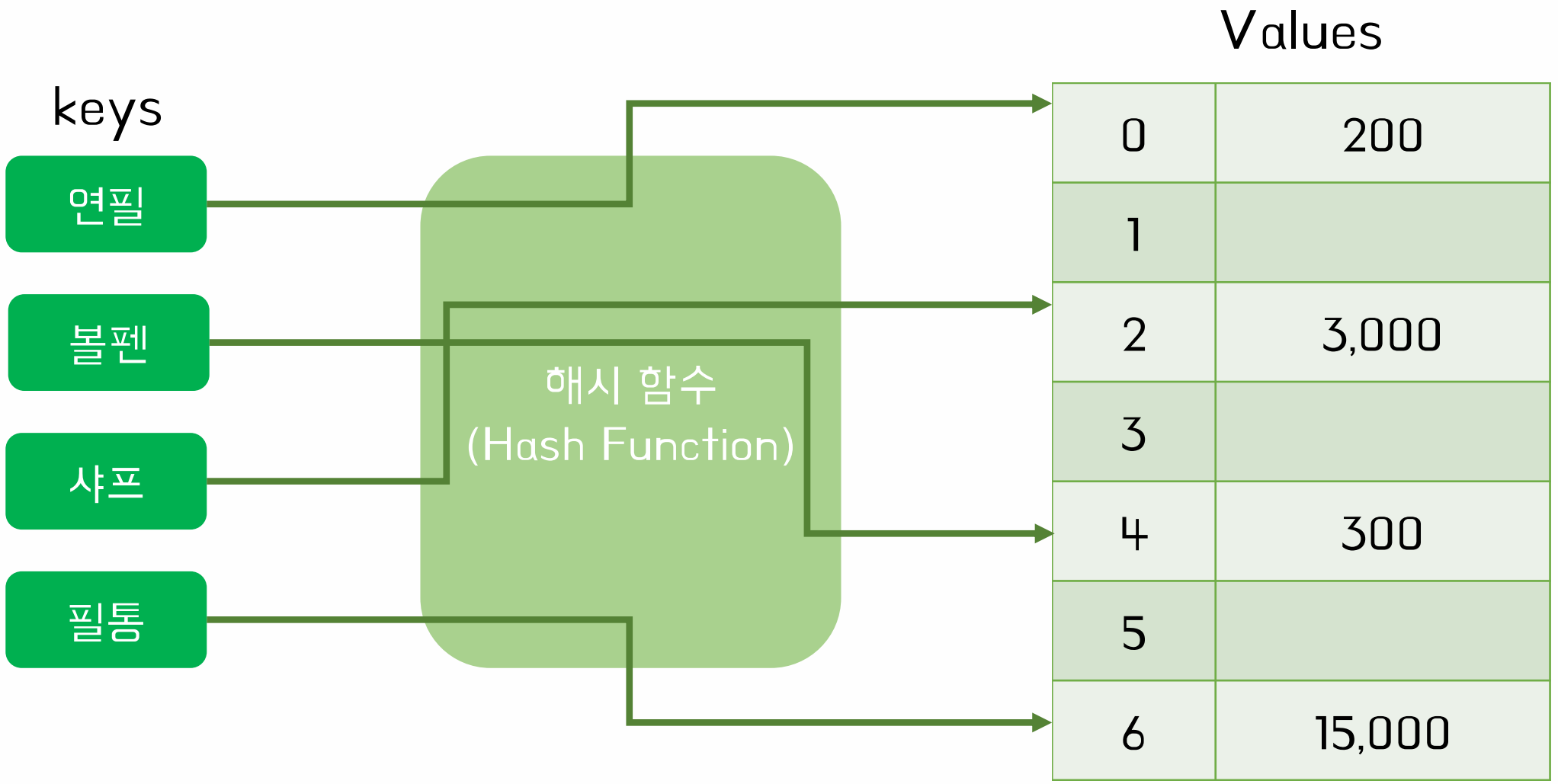
LEARN

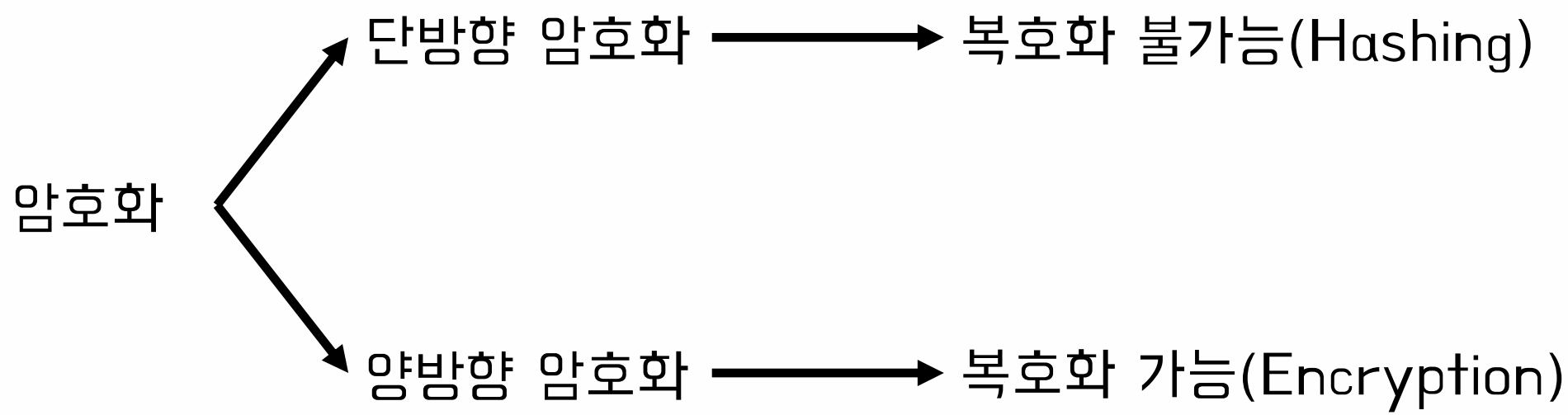
회원가입 route

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('/');
    });
});
```

암호화


keys
연필
볼펜
샤프
필통
해시 함수
(Hash Function)
Values
0
200
1
2
3,000
3
4
300
5
6
15,000


암호화
단방향 암호화
복호화 불가능(Hashing)
양방향 암호화
복호화 가능(Encryption)

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");
```

DOIT

crypto 모듈을 사용하여 회원가입 & 로그인 비밀번호 부분 처리

LEARN

회원가입 route

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...

Cookie

정보가 클라이언트에 저장됨
정보가 남아있음
누구나 접근 가능

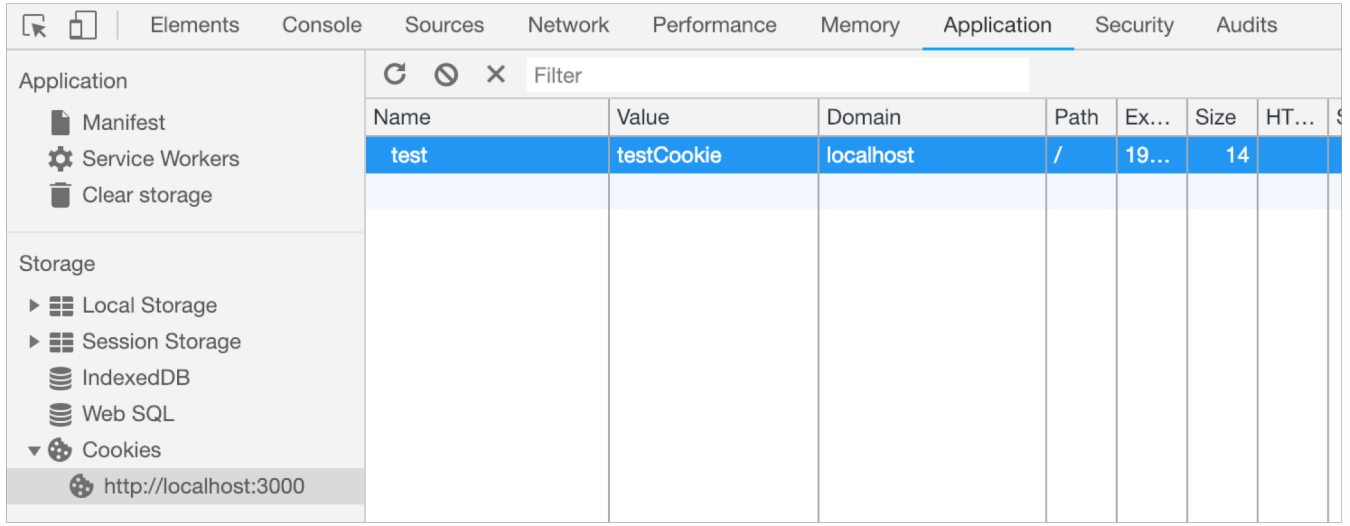
정보가 서버에 저장됨
클라이언트는 세션 정보만 가짐
브라우저 종료 시 / 서버 세션 만료 시 정보 없어짐

LEARN

Cookie

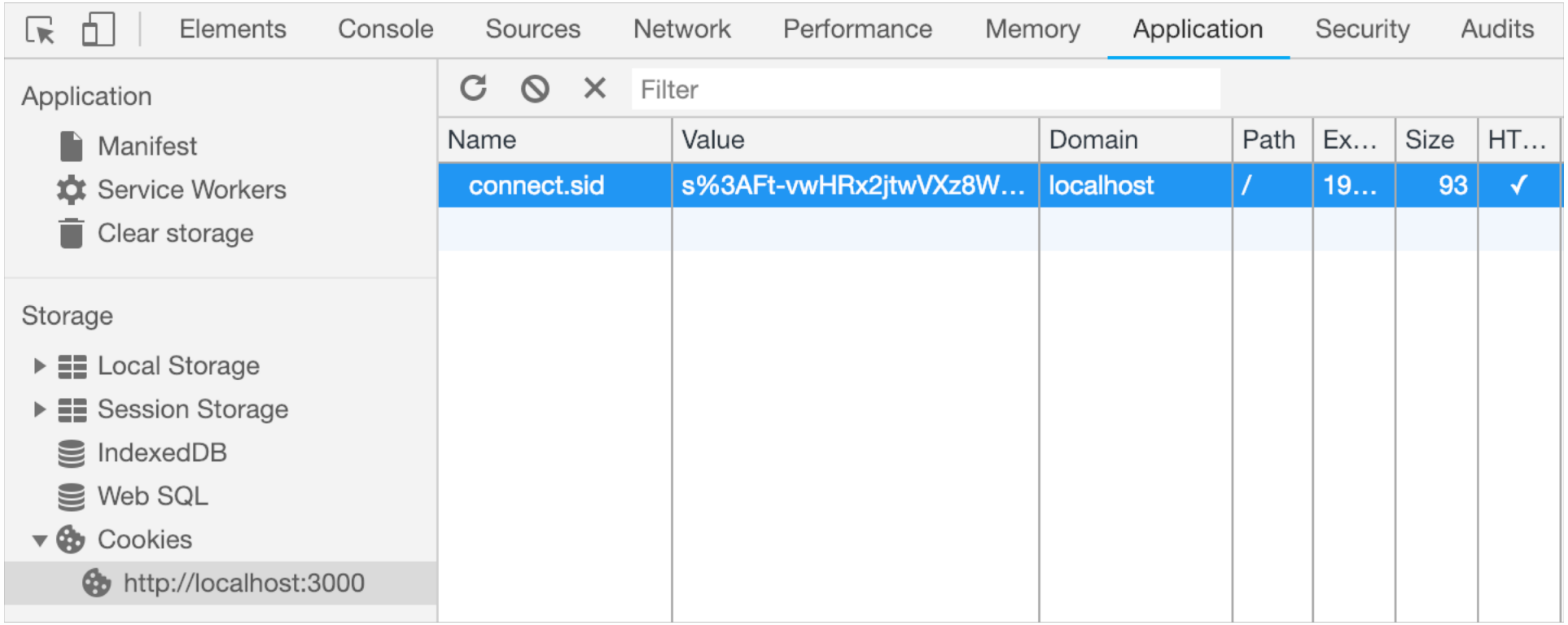
JS index.js

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



LEARN

Session



LEARN

로그인 route

JS users.js

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

로그인 view

https://github.com/shuni4481/node_note/blob/master/views/login.ejs

passport 다운

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

passport 모듈 & express-session 설치

LEARN

로그인 route

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());
```

로그인 route

로그인 route

index.js

```
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":{}}
```

DOIT

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