**8장. 몽고디비**

몽고DB 다운 링크 : <https://www.mongodb.com/download-center/community>

설치 가이드 : <https://blog.naver.com/didlc/221124944440>

설치 후에 cmd

cd 설치경로\bin

mongod --dbpath “C:\mongodb\data\db”

다른 cmd창 열어서

mongo

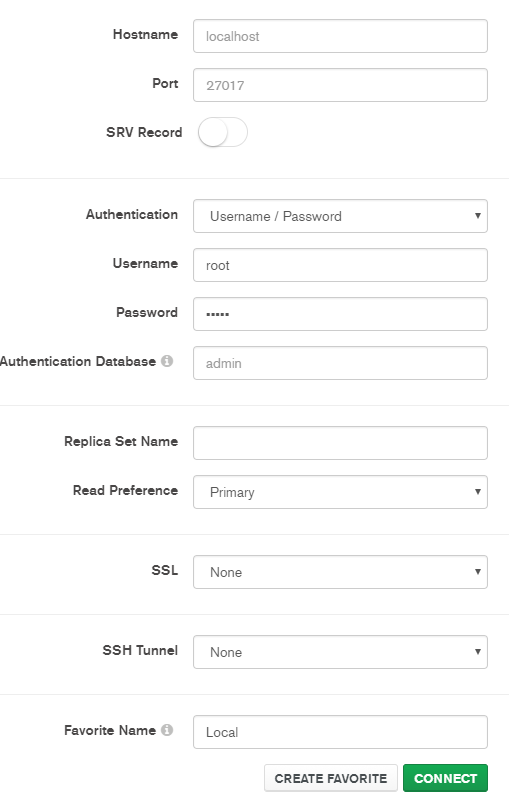
> use admin

> db.createUser({user:’이름’,pwd:’비밀번호’,role:[‘root’]})

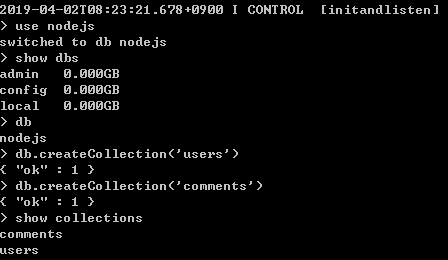
모든 콘솔 종료 후,

mongod --auth --dbpath “C:\mongodb\data\db”

mongo admin -u root -p pw123

**컴퍼스**

**[데이터 베이스 및 컬렉션 생성하기]**



**[생성]**

> db.users.save({name:'zero',age:24,married:false,comment:'안녕하세요. 간단히 몽

고디비 사용방법에 대해 알아봅시다',createdAt:new Date()});

WriteResult({ "nInserted" : 1 })

> db.users.save({name:'nero',age:32,married:true,comment:'안녕하세요. zero 친구

nero 입니다.',createdAt:new Date()});

WriteResult({ "nInserted" : 1 })

**[조회]**

> db.users.find({name:'zero'},{\_id:1})

{ "\_id" : ObjectId("5ca2a0c68c046e757c5d18be") }

> db.comments.save({commenter:ObjectId('5ca2a0c68c046e757c5d18be'),comment:'안녕

하세요. zero의 댓글글입니다. ',ceratedAt: new Date()});

WriteResult({ "nInserted" : 1 })

> db.users.find({});

{ "\_id" : ObjectId("5ca2a0c68c046e757c5d18be"), "name" : "zero", "age" : 24, "ma

rried" : false, "comment" : "안녕하세요. 간단히 몽고디비 사용방법에 대해 알아봅

시다", "createdAt" : ISODate("2019-04-01T23:37:42.203Z") }

{ "\_id" : ObjectId("5ca2a0f68c046e757c5d18bf"), "name" : "nero", "age" : 32, "ma

rried" : true, "comment" : "안녕하세요. zero 친구 nero 입니다.", "createdAt" : I

SODate("2019-04-01T23:38:30.386Z") }

> db.comments.find({});

{ "\_id" : ObjectId("5ca2a2268c046e757c5d18c0"), "commenter" : ObjectId("5ca2a0c6

8c046e757c5d18be"), "comment" : "안녕하세요. zero의 댓글글입니다. ", "ceratedAt"

: ISODate("2019-04-01T23:43:34.025Z") }

> db.users.find({},{\_id:0,name:1,married:1});

{ "name" : "zero", "married" : false }

{ "name" : "nero", "married" : true }

> db.users.find({age:{$gt:30},married:true},{\_id:0,name:1,age:1});

{ "name" : "nero", "age" : 32 }

> db.users.find({$or:[{age:{$gt:30}},{married:false}]},{\_id:0,name:1,age:1});

{ "name" : "zero", "age" : 24 }

{ "name" : "nero", "age" : 32 }

* **$gt(초과), $gte(이상), $lt(미만), $lte(이하), $ne(같지 않음), $or(또는), $in(배열 요소 중 하나)**

> db.users.find({}, {\_id:0,name:1,age:1}).sort({age:-1})

{ "name" : "nero", "age" : 32 }

{ "name" : "zero", "age" : 24 }

> db.users.find({}, {\_id:0,name:1,age:1}).sort({age:-1}).limit(1)

{ "name" : "nero", "age" : 32 }

> db.users.find({}, {\_id:0,name:1,age:1}).sort({age:-1}).limit(1).skip(1)

{ "name" : "zero", "age" : 24 }

**[update]**

> db.users.update({name:'nero'},{$set:{comment:'안녕하세요. 이 필드를 바꿔보겠습

니다.'}});

WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

**[delete]**

> db.users.remove({name:'nero'})

WriteResult({ "nRemoved" : 1 })

**8. 몽구스 사용하기**

Microsoft Windows [Version 6.1.7601]

Copyright (c) 2009 Microsoft Corporation. All rights reserved.

**C:\Users\Administrator>cd C:\materK1m\newmimac\test\_kes**

**C:\materK1m\newmimac\test\_kes>express learn-mongoose --view=pug**

create : learn-mongoose\

create : learn-mongoose\public\

create : learn-mongoose\public\javascripts\

create : learn-mongoose\public\images\

create : learn-mongoose\public\stylesheets\

create : learn-mongoose\public\stylesheets\style.css

create : learn-mongoose\routes\

create : learn-mongoose\routes\index.js

create : learn-mongoose\routes\users.js

create : learn-mongoose\views\

create : learn-mongoose\views\error.pug

create : learn-mongoose\views\index.pug

create : learn-mongoose\views\layout.pug

create : learn-mongoose\app.js

create : learn-mongoose\package.json

create : learn-mongoose\bin\

create : learn-mongoose\bin\www

change directory:

> cd learn-mongoose

install dependencies:

> npm install

run the app:

> SET DEBUG=learn-mongoose:\* & npm start

**C:\materK1m\newmimac\test\_kes>cd learn-mongoose && npm i**

npm notice created a lockfile as package-lock.json. You should commit this file.

added 118 packages from 174 contributors and audited 247 packages in 7.611s

found 1 low severity vulnerability

run `npm audit fix` to fix them, or `npm audit` for details

**C:\materK1m\newmimac\test\_kes\learn-mongoose>npm i mongoose**

+ mongoose@5.4.20

added 20 packages from 14 contributors and audited 285 packages in 2.85s

found 1 low severity vulnerability

run `npm audit fix` to fix them, or `npm audit` for details

**[루트 폴더에 schemas 생성 index.js 생성]**

const mongoose = require('mongoose');

module.exports = () => {

const connect = () => {

if (process.env.NODE\_ENV !== 'production'){

mongoose.set('debug', true);

}

// 몽구스 연결

mongoose.connect('mongodb://root:pw123@localhost:27017/admin', {

dbName : 'nodejs',

}, (error) => {

if(error){

console.log('몽고디비 연결 에러', error);

}else{

console.log('몽고디비 연결 성공!');

}

});

};

connect();

mongoose.connection.on('error',(error) => {

console.error('몽고디비 연결 에러', error);

});

mongoose.connection.on('disconnected', () =>{

console.error('몽고디비 연결이 끊겼습니다. 연결을 재시도합니다.');

connect();

});

require('./user');

require('./comment');

}

**[app.js]**

var indexRouter = require('./routes/index');

var usersRouter = require('./routes/users');

var connect = require('./schemas');

var app = express();

connect();

**[ 스키마 정의하기 ]**

Schemas 폴더에 user.js와 comment.js를 만든 후,

* schemas/user.js

const mongoose = require('mongoose');

const{Schema} = mongoose;

const userSchema = new Schema({

name : {

type : String,

required : true,

unique : true,

},

age : {

type : Number,

required : true,

},

married : {

type : Boolean,

required : true,

},

comment : String,

createdAt : {

type : Date,

default : Date.now, // 데이터 생성 당시의 시간

},

});

module.exports = mongoose.model('User', userSchema);

* schemas/comment.js

const mongoose = require('mongoose');

const { Schema } = mongoose;

const { Types : { ObjectId } } = Schema;

const commentSchema = new Schema({

commenter : {

type : ObjectId,

required : true,

ref : 'User', // User을 ref

},

comment : {

type : String,

required : true,

},

createdAt : {

type : Date,

default : Date.now,

},

});

module.exports = mongoose.model('Comment', commentSchema);

**[ 쿼리 수행하기 ]**

* view/mongoose.pug

doctype html

html

head

meta(charset='utf-8')

title 몽구스 서버

style.

table {

border: 1px solid black;

border-collapse: collapse;

}

table th, table td {

border: 1px solid black;

}

body

div

form#user-form

fieldset

legend 사용자 등록

div

input#username(type="text" placeholder="이름")

div

input#age(type="number" placeholder="나이")

div

input#married(type="checkbox")

label(for="married") 결혼 여부

button(type="submit") 등록

br

table#user-list

thead

tr

th 아이디

th 이름

th 나이

th 결혼여부

tbody

for user in users

tr

td= user.\_id

td= user.name

td= user.age

td= user.married ? '기혼' : '미혼'

br

div

form#comment-form

fieldset

legend 댓글 등록

div

input#userid(type="text" placeholder="사용자 아이디")

div

input#comment(type="text" placeholder="댓글")

button(type="submit") 등록

br

table#comment-list

thead

tr

th 아이디

th 작성자

th 댓글

th 수정

th 삭제

tbody

script(src='/mongoose.js')

* public/mongoose.js

// 사용자 이름 눌렀을 때 댓글 로딩

document.querySelectorAll('#user-list tr').forEach(function (el) {

el.addEventListener('click', function () {

var id = el.querySelector('td').textContent;

getComment(id);

});

});

// 사용자 로딩

function getUser() {

var xhr = new XMLHttpRequest();

xhr.onload = function () {

if (xhr.status === 200) {

var users = JSON.parse(xhr.responseText);

console.log(users);

var tbody = document.querySelector('#user-list tbody');

tbody.innerHTML = '';

users.map(function (user) {

var row = document.createElement('tr');

row.addEventListener('click', function () {

getComment(user.\_id);

});

var td = document.createElement('td');

td.textContent = user.\_id;

row.appendChild(td);

td = document.createElement('td');

td.textContent = user.name;

row.appendChild(td);

td = document.createElement('td');

td.textContent = user.age;

row.appendChild(td);

td = document.createElement('td');

td.textContent = user.married ? '기혼' : '미혼';

row.appendChild(td);

tbody.appendChild(row);

});

} else {

console.error(xhr.responseText);

}

};

xhr.open('GET', '/users');

xhr.send();

}

// 댓글 로딩

function getComment(id) {

var xhr = new XMLHttpRequest();

xhr.onload = function () {

if (xhr.status === 200) {

var comments = JSON.parse(xhr.responseText);

var tbody = document.querySelector('#comment-list tbody');

tbody.innerHTML = '';

comments.map(function (comment) {

var row = document.createElement('tr');

var td = document.createElement('td');

td.textContent = comment.\_id;

row.appendChild(td);

td = document.createElement('td');

td.textContent = comment.commenter.name;

row.appendChild(td);

td = document.createElement('td');

td.textContent = comment.comment;

row.appendChild(td);

var edit = document.createElement('button');

edit.textContent = '수정';

edit.addEventListener('click', function () { // 수정 클릭 시

var newComment = prompt('바꿀 내용을 입력하세요');

if (!newComment) {

return alert('내용을 반드시 입력하셔야 합니다');

}

var xhr = new XMLHttpRequest();

xhr.onload = function () {

if (xhr.status === 200) {

console.log(xhr.responseText);

getComment(id);

} else {

console.error(xhr.responseText);

}

};

xhr.open('PATCH', '/comments/' + comment.\_id);

xhr.setRequestHeader('Content-Type', 'application/json');

xhr.send(JSON.stringify({ comment: newComment }));

});

var remove = document.createElement('button');

remove.textContent = '삭제';

remove.addEventListener('click', function () { // 삭제 클릭 시

var xhr = new XMLHttpRequest();

xhr.onload = function () {

if (xhr.status === 200) {

console.log(xhr.responseText);

getComment(id);

} else {

console.error(xhr.responseText);

}

};

xhr.open('DELETE', '/comments/' + comment.\_id);

xhr.send();

});

td = document.createElement('td');

td.appendChild(edit);

row.appendChild(td);

td = document.createElement('td');

td.appendChild(remove);

row.appendChild(td);

tbody.appendChild(row);

});

} else {

console.error(xhr.responseText);

}

};

xhr.open('GET', '/comments/' + id);

xhr.send();

}

// 사용자 등록 시

document.getElementById('user-form').addEventListener('submit', function (e) {

e.preventDefault();

var name = e.target.username.value;

var age = e.target.age.value;

var married = e.target.married.checked;

if (!name) {

return alert('이름을 입력하세요');

}

if (!age) {

return alert('나이를 입력하세요');

}

var xhr = new XMLHttpRequest();

xhr.onload = function () {

if (xhr.status === 201) {

console.log(xhr.responseText);

getUser();

} else {

console.error(xhr.responseText);

}

};

xhr.open('POST', '/users');

xhr.setRequestHeader('Content-Type', 'application/json');

xhr.send(JSON.stringify({ name: name, age: age, married: married }));

e.target.username.value = '';

e.target.age.value = '';

e.target.married.checked = false;

});

// 댓글 등록 시

document.getElementById('comment-form').addEventListener('submit', function (e) {

e.preventDefault();

var id = e.target.userid.value;

var comment = e.target.comment.value;

if (!id) {

return alert('아이디를 입력하세요');

}

if (!comment) {

return alert('댓글을 입력하세요');

}

var xhr = new XMLHttpRequest();

xhr.onload = function () {

if (xhr.status === 201) {

console.log(xhr.responseText);

getComment(id);

} else {

console.error(xhr.responseText);

}

};

xhr.open('POST', '/comments');

xhr.setRequestHeader('Content-Type', 'application/json');

xhr.send(JSON.stringify({ id: id, comment: comment }));

e.target.userid.value = '';

e.target.comment.value = '';

});

* routes/index.js

var express = require('express');

var User = require('../schemas/user');

var router = express.Router();

router.get('/', function (req, res, next) {

User.find({})

.then((users) => {

res.render('mongoose', { users });

})

.catch((err) => {

console.error(err);

next(err);

});

});

module.exports = router;

* routes/user.js

const mongoose = require('mongoose');

const{Schema} = mongoose;

const userSchema = new Schema({

name : {

type : String,

required : true,

unique : true,

},

age : {

type : Number,

required : true,

},

married : {

type : Boolean,

required : true,

},

comment : String,

createdAt : {

type : Date,

default : Date.now, // 데이터 생성 당시의 시간

},

});

module.exports = mongoose.model('User', userSchema);

* routes/comments.js

const mongoose = require('mongoose');

const { Schema } = mongoose;

const { Types : { ObjectId } } = Schema;

const commentSchema = new Schema({

commenter : {

type : ObjectId,

required : true,

ref : 'User', // User을 ref

},

comment : {

type : String,

required : true,

},

createdAt : {

type : Date,

default : Date.now,

},

});

module.exports = mongoose.model('Comment', commentSchema);

* app.js

var indexRouter = require('./routes/index');

var usersRouter = require('./routes/users');

var commentRouter = require('./routes/comments');

var connect = require('./schemas');

var app = express();

connect();

// view engine setup

app.set('views', path.join(\_\_dirname, 'views'));

app.set('view engine', 'pug');

app.use(logger('dev'));

app.use(express.static(path.join(\_\_dirname, 'public')));

app.use(express.json());

app.use(express.urlencoded({ extended: false }));

app.use(cookieParser());

app.use(express.static(path.join(\_\_dirname, 'public')));

app.use('/', indexRouter);

app.use('/users', usersRouter);

app.use('/comments', commentRouter);

**[ 실행 ]**

c:\mongodb\bin\mongod.exe --auth --dbpath c:\mongodb\data\db

mongo admin -u root -p pw123

C:\materK1m\newmimac\test\_kes\learn-mongoose>npm start

> learn-mongoose@0.0.0 start C:\materK1m\newmimac\test\_kes\learn-mongoose

> node ./bin/www

(node:12276) DeprecationWarning: current URL string parser is deprecated, and will be removed in a future version. To use the new parser, pass option { useNewUrlParser: true } to MongoClient.connect.

Mongoose: users.ensureIndex({ name: 1 }, { unique: true, background: true })

(node:12276) DeprecationWarning: collection.ensureIndex is deprecated. Use createIndexes instead.

몽고디비 연결 성공!

<http://localhost:3000/>

