MongoDB 배우기

엘리스 이론강의 PPT 설명

**설치**

맥에서 어떻게 설치하는지 설명

**설치확인**

mongo

웹브라우저, localhost:27017

MongoDBCompass

**데이터베이스 생성하기**

db

show dbs

use mydb

db.myfriend.insert({“name”:”minjung”, age:30})

db.stats

db.dropDatabase()

**Collection 다루기**

컬렉션을 생성하는 2가지 방법

db.createCollection(“myfriend”)

db.createCollection(“diary”, {옵션})

show collections;

capped : Boolean타입. 이 값을 true로 설정하면 capped collection을 활성화 시킨다.

고정사이즈 생성. 공간이 모두 사용되면 처음데이터 삭제후 재사용.

non-cappted collection

일반적 컬렉션. 저장공간 자동증가

size : number타입이다. Capped collection을 위해 해당 컬렉션의 최대 사이즈를 ~bytes로 지정한다.

max : number타입이다. 해당 컬렉션에 추가 할 수 있는 최대 document 갯수를 설정한다.

db.diary.insertOne({“title”:”happy day”})

db.diary.validate() #컬렉션 정보 확인

db.diary.renameCollection(“바꿀 이름”);

db.diary.drop()

**CRUD배우기**

**READ**

db.colellection.find()

db.collection.find( {document} )

findOne() : 결과가 하나만 리턴되는 조회

count() : 행의 갯수를 리턴

distinct("컬렴명") : 중복을 제거

값 비교 옵션들

$eq (equals) 주어진 값과 일치하는 값

$gt (greater than) 주어진 값보다 큰 값

$gte (greather than or equals) 주어진 값보다 크거나 같은 값

$lt (less than) 주어진 값보다 작은 값

$lte (less than or equals) 주어진 값보다 작거나 같은 값

$ne (not equal) 주어진 값과 일치하지 않는 값

$in 주어진 배열 안에 속하는 값

$nin 주어빈 배열 안에 속하지 않는 값

db.inventory.find({qty: {$gt:40, $lt:60}})

db.inventory.find({item:{$in:["canvas", "mat", "father"]}})

논리연산자

{$or: [{ \_\_CONDITION1\_\_ }, { \_\_CONDITION2\_\_}]}

db.inventory.find({$or: [{item:"canvas"}, {qty:{$lt:55}}]})

선택적으로 출력하기

find({}, {\_\_PROJECTIONS\_\_, , ... })

db.inventory.find({}, {"item":true, "qty":true, \_id:false})

Sort 메소드

오름차순은 1, 내림차순은 -1

db.inventory.find({},{status:true, qty:true, \_id:false}).'''sort({"status":1, qty:-1})'''

db.inventory.find().limit(3)

db.inventory.find().skip(3)

Embedded doc 찾기

db.diary.find({"docs.title":"christmas..."})

db.diary.find({"docs.title":/ch/})

---------------------------------------------------

insertOne({데이터1})

db.diary.insertOne({"minjung":"good"}, {"chulsoo":"bad"}, {"hong":"soso"});

mydb> db.diary.find()

[ { \_id: ObjectId("618610c017edd513575babb5"), minjung: 'good' } ]

insertMany( [ {데이터1}, {데이터2},...,{데이터3} ] )

db.inventory.update({"status":"A"}, {name:"Kim mijeong"}) #덮어쓰기

db.inventory.update({name : "Kim mijeong"}, {$set : {age: 28}}) #찾아서 특정field 변경

db.inventory.update({"name":"Kim mijeong"}, '''{$unset : {age: 1}}''') #찾아서 특정field 제거

db.inventory.update({“name”:”Kim mijeong”}, {$inc: {age: 10}}, {multi:true})

[배열을 이용해서 작업하기]

배열데이터를 추가하는 방법

db.test.update({id:"jang"},{$set:{favorites:{city:["서울","안양"],movie:["쉬리","태극기휘날리며"]}})

배열의 update옵션

$addToSet : 기존 배열에 해당 값이 없을 떄 필드와 값을 추가 \*\*[중복을 체크할때 사용하는 연산자]\*\*

ex) db.test.update({id:"jang"},{$addToSet:{"favorites.city":"인천"}})

$push : 배열의 요소를 추가한다. \*\*[ 중복을 허용 ]\*\*

ex) db.test.update({id:"jang"},{$push:{"favorites.city":"부산"}})

$pop : 배열의 첫 번째나 마지막 요소를 제거한다.

1을 옵션을 이용하면 마지막 요소를 제거한다.

- 1 을 옵션을 이용하면 첫 번째 요소를 제거한다.

ex) db.test.update({id:"jang"},{$pop:{"favorites.city":1}})

ex) db.test.update({id:"jang"},{$pop:{"favorites.city":-1}})

$each : addToSet이나 push에서 사용할 수 있다.

여러개의 값을 배열에 추가할때 사용

db.test.update({id:"jang"},{$push:{"favorites.city":{$each:["천안","가평","군산"]}}})

$sort : 정렬 ( 1 => 오름 차순 , -1 => 내림 차순)

db.test.update({id:"jang"},{$push:{"favorites.city":{$each:["천안","가평","군산"],$sort:1}}})

$pull : 배열에서 조건에 맞는 요소를 제거

ex) db.test.update({id:"jang"},{$pull:{"favorites.city":"가평"}})

$pullAll : 여러 개를 조건을 정의하여 제거

ex) db.test.update({id:"jang"},{$pullAll:{"favorites.city":["천안","]}})

-----------------------------------------------------------

db.collection.deleteOne()

db.collection.deleteMany()

번외

* **변수 선언**

선언할 변수명 = 데이터값

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/c1831708-427f-4b2d-9fd5-4e7b14fe89cf/9933EB505B7FA19E05>

for문을 이용해서 데이터 삽입도 가능

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/c8de63a1-5bfc-45ad-a16b-a7c3e061510a/9950FA425B7FA23A25>

* **while문 사용해서 콜렉션 만들기**

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/27fa9ab4-bb57-4ce5-977a-31cea3498bda/99C38F4F5B838EFE2D>

* **기본 명령어**

**CSV import 하기**

**find 연습하기**

**Example Data**

db.inventory.insertMany( [

{ item: "canvas", qty: 100, size: { h: 28, w: 35.5, uom: "cm" }, status: "A" },

{ item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status: "A" },

{ item: "mat", qty: 85, size: { h: 27.9, w: 35.5, uom: "cm" }, status: "A" },

{ item: "mousepad", qty: 25, size: { h: 19, w: 22.85, uom: "cm" }, status: "P" },

{ item: "notebook", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status: "P" },

{ item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },

{ item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },

{ item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" },

{ item: "sketchbook", qty: 80, size: { h: 14, w: 21, uom: "cm" }, status: "A" },

{ item: "sketch pad", qty: 95, size: { h: 22.85, w: 30.5, uom: "cm" }, status: "A" }

]);

db.inventory.insertMany( [

{ item: "b", qty: 100, size: { h: 28, w: 35.5, uom: "cm" }, status: "A" },

{ item: "c", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status: "A" },

{ item: "d", qty: 85, size: { h: 27.9, w: 35.5, uom: "cm" }, status: "A" },

{ item: "e", qty: 25, size: { h: 19, w: 22.85, uom: "cm" }, status: "P" },

{ item: "f", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status: "P" },

{ item: "g", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },

{ item: "h", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },

{ item: "i", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" },

{ item: "j", qty: 80, size: { h: 14, w: 21, uom: "cm" }, status: "A" },

{ item: "k pad", qty: 95, size: { h: 22.85, w: 30.5, uom: "cm" }, status: "A" },

{ item: "l", qty: 100, size: { h: 28, w: 35.5, uom: "cm" }, status: "A" },

{ item: "m", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status: "A" },

{ item: "n", qty: 85, size: { h: 27.9, w: 35.5, uom: "cm" }, status: "A" },

{ item: "o", qty: 25, size: { h: 19, w: 22.85, uom: "cm" }, status: "P" },

{ item: "p", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status: "P" },

{ item: "q", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },

{ item: "r", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },

{ item: "s", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" },

{ item: "t", qty: 80, size: { h: 14, w: 21, uom: "cm" }, status: "A" },

{ item: "u pad", qty: 95, size: { h: 22.85, w: 30.5, uom: "cm" }, status: "A" },

{ item: "v", qty: 80, size: { h: 14, w: 21, uom: "cm" }, status: "A" },

{ item: "w", qty: 95, size: { h: 22.85, w: 30.5, uom: "cm" }, status: "A" },

{ item: "x", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status: "P" },

{ item: "y", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },

{ item: "z", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },

{ item: "11", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" }

]);

**어린왕자 챗봇만들기**

텍스트데이터 전처리

문장 찾기

db.getCollection('ip').find({addr:/중구/})

db.getCollection('ip').find({addr:/^서울/})

db.getCollection('ip').find({addr:/2동$/})

db.collection.find({name:{$regex: “문자열”}})

db.collection.find({name:{$regex: “문자열”, ‘$options’:’i’}}) #대소문자 구분없이