



데이터베이스(sql/nosql)

DB 수업내용



- DB소개
- RDBMS 이해
- SQL실습환경 셋팅
- SQLite로 SQL 기본 익히기
- mysql 설치, 설정, 사용자 생성
- SQL(with mysql)
 - CML(CRUD), DDL
 - Join, sql 함수
- **Nosql(mongoDB)**
 - **DB create, CRUD**

학습내용

- 실습 환경 구축
 - mongodb 서버, mongodb client 설치
 - mongodb 기본 문법 익히기
 -
- datagather 가상환경에서 jupyter notebook 설정
 - 가상환경에 ipykernel 설치
(dataVENV) \$ pip install ipykernel
 - 주피터 노트북에 가상환경 커널 등록
 - (dataVENV) \$ ipython kernel install --user --name=dataVENV



학습목표

빅데이터와 Nosql이 무엇인지 안다.

몽고DB를 설치하고, 클라이언트 T3 프로그램과 연결한다.

몽고의 DBMS, Collection, Document를 이해한다.

몽고DB CRUD 명령을 실행한다.

파이썬으로 크롤링한 데이터를 몽고 DB와 연동하여 입력하고,
조회한다.

빅데이터와 nosql 이해

빅데이터란?

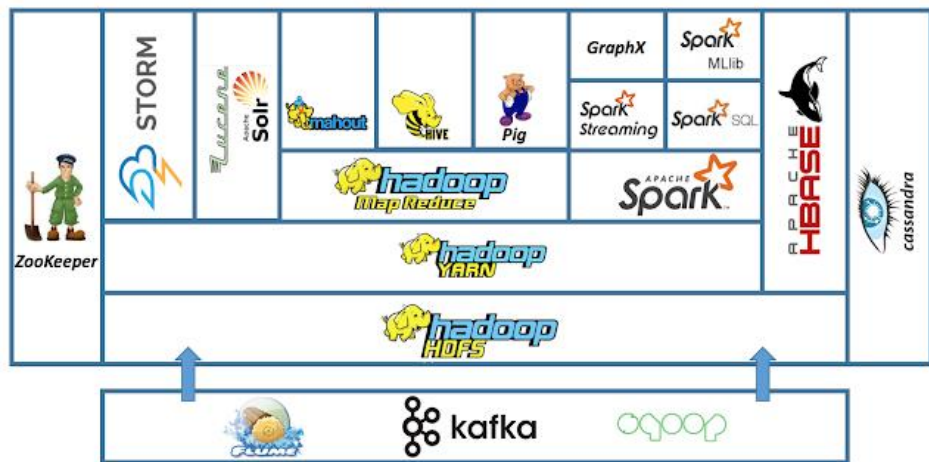
- 빅데이터 정의 - 위키백과

기존 데이터베이스 관리도구의 능력을 넘어서는 **대량의 정형** 또는 심지어 데이터베이스 형태가 아닌 **비정형의 데이터** 집합조차 포함한 데이터로부터 **가치를 추출하고 결과를 분석하는 기술**

- 빅데이터 특징 : 3V
 - 데이터의 **량(Volume)**
 - 데이터의 **증가 속도(Velocity)**
 - 데이터의 **종류(Variety)**

데이터 분산관리 시스템(Hadoop)

- Hadoop(High-Availability Distributed Object-Oriented Platform)
- 하둡 기본 구성
 - HDFS(Hadoop Distributed System) : 하둡 분산 파일 시스템
 - MapReduce : 분산 처리 프레임 워크
 - YARN(yet another resource negotiator) : 클러스터 자원 관리



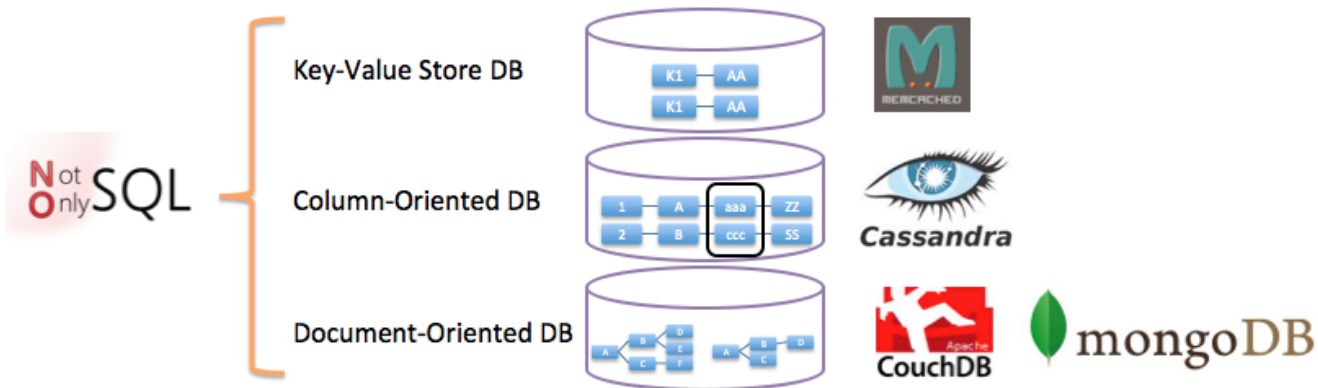
NoSQL DB 이해

- 비정형 데이터, Not only SQL
- RDBMS의 한계를 극복하기 위해 만들어진 새로운 형태의 데이터저장소

정형DB(SQL)	비정형DB(NoSQL)
정해진 규격 Schema, table – column	정해진 규격 없음
Join 가능	Join 불가능
트랜잭션 사용	트랜잭션 없음
분산처리 어려움	분산처리 쉬움

NoSQL 개요

- NoSQL 데이터베이스는 각 데이터베이스마다 기반으로 하는 데이터 모델이 다르므로, 데이터 모델별로 대표적인 데이터베이스를 알아 둘 필요가 있음
- 각 데이터베이스를 다루는 인터페이스가 다름



Why NoSQL ?

- RDBMS를 기본으로 사용하지만, **초당 데이터가 수십만개씩 쌓이는 서비스**가 많아지면서(SNS, 온라인 서비스등), **NoSQL을 사용하는 경우가 많아지고 있음**
- 경험적 수치
 - 95% read, 5% write 경우는 RDBMS 가 성능이 나쁘지 않음
 - 50% write > 인 경우 RDBMS는 성능 저하 또는 불안정
 - NoSQL + Redis (In memory cache) 등을 고려하게 됨



MongoDB

MongoDB 소개



- 비정형 DB 중에 가장 널리 쓰이고 있음
- 정보가 많고 처음에 익히기에 쉬움
- JSON 기반의 Document 데이터 관리
- 프로그래밍에 다루는 데이터 포맷
 - 데이터형은 저장할 때 정해짐
 - 정수(int), 소수점(float), 문자(string)
 - CSV, JSON등

MongoDB 소개

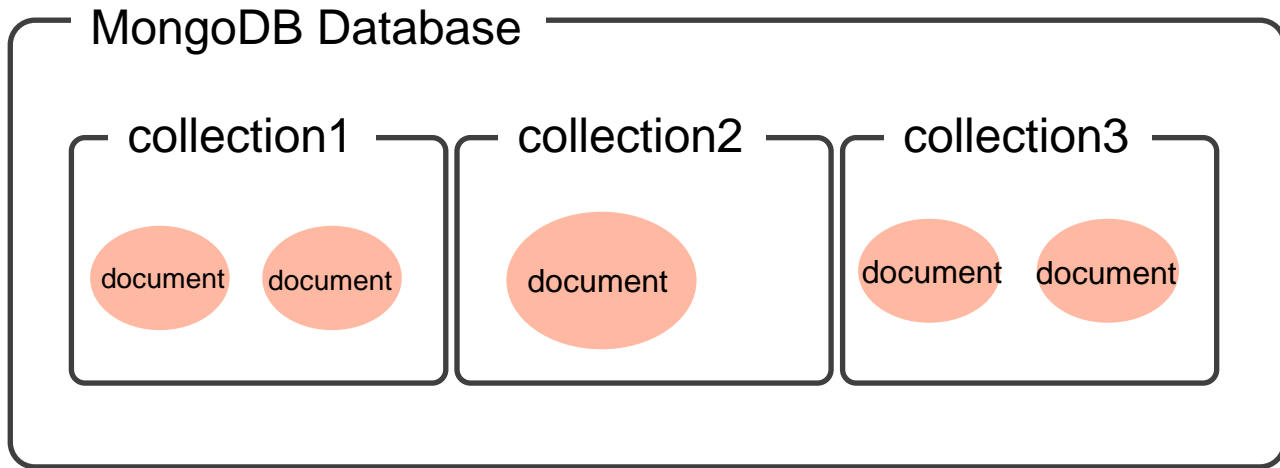
JSON data 예)

```
document = { "id": "01", "language": "java",  
              "edition": { "first": "1st", "second": "2nd", "third": "third" }  
            }
```

mongoDB Docment 예)

```
{  
  "_id": ObjectId("5099803df3f42312312391"),  
  "username": "song",  
  "name": { first: "joy", last: "Lee" }  
}
```

MongoDB 데이터 구조



용어	설명
MongoDB	Collection의 집합
Collection	Document의 집합, RDBMS의 Table과 같은 위치
Document	JSON 타입의 데이터 집합

Mysql vs MongoDB




Mysql	MongoDB
Database	Database
Table	Collection
Tuple / Row	Document
Column	Key/Field
Table join	Embedded Documents
Primary Key	Primary Key(_id)
mysqld	mongod
mysql / dbeaver	mongo / robo 3T, dbeaver

MongoDB 설치 및 환경설정



- 로컬 환경(ubuntu18.04)에 설치
- mongodb 설치(서버)
- mongo client 설치
 - studio 3T 설치
- 첨부 파일 참고 : 1.mongodb_setup_ex00.pdf

MongoDB 기본 기능



- MongoDB CRUD 익히기
- 첨부 파일 참고 : [2.mongodb_setup_ex02.pdf](#)

MongoDB와 파이썬 프로그래밍

- mongodb python 모듈 설치
\$ pip3.8 install pymongo
- python 코드 내에 pymongo 사용 과정

명령	설명
import pymongo	pymongo 라이브러리 import
conn = pymongo.MongoClient()	Mongodb 접속 객체 생성 *원격: pymongo.MongoClient("mongodb:ip주소")
db = conn.mydbs	Database 객체 생성 또는 선택
col_it = db.mycollection	Collection 객체 생성 또는 선택
col_it.insert_one()	database의 collection에 CRUD 명령

MongoDB와 파이썬 프로그래밍

- 키 : ""로 감싸야 함.
 - `post = {"키1": "값1", "키2": "값2" ...}`
 - `col_it.insert_one(post)`
- `insert_many([{ } , { }])`
 - `posts = { [{"키1": "값1", "키2": "값2"}, {"키3": "값3" }] }`
 - `col_it.insert_many(posts)`
- 실습 : 첨부파일 참조