

**Fastcampus**

**Computer Science SCHOOL**

**Database Basic**

2017.2.13

# Introduce

## 최우영

- Solutions Architect, Back-End Developer at unknown team
- Solutions Architect, Web Developer, Instructor
- Skills: Python, Golang, Julia, Node.js, Google Tag Manager, ...

blog: <https://ulgoon.github.io>

github: <https://github.com/ulgoon>

# Index

- Database
  - Introduction
  - Practice
  - SQL
  - RDB
  - NoSQL
  - Database with python

# data

- 컴퓨터가 처리할 수 있는 문자, 숫자, 소리, 그림 따위의 형태로 된 정보.
- Latin "Datum"의 복수형 "Data"에서 유래

# Database

- 체계화된 데이터의 모임
- 여러 응용 시스템들의 통합된 정보들을 저장하여 운영할 수 있는 공용 데이터들의 묶음

# DB?? DBMS??

DBMS(DataBase Management System)

- 데이터의 모임인 Database를 만들고, 저장, 관리 할 수 있는 기능을 제공하는 응용프로그램
- Oracle, Mysql, MariaDB, DB2, MS SQL Server, ..

# DBMS의 조상님

CURSOR	<-- -->	UP	DOWN	DELETE	Insert Mode: Ins
Char:	< >	Field: ↑	↓	Char: Del	Exit/Save: ^End
Word:	Home End	Page: PgUp	PgDn	Field: ^Y	Abort: Esc
		Help: F1		Record: ^U	Memo: ^Home

FIRSTNAME	Claire
LASTNAME	Buckman
ADDRESS	8307 Santa Anita Blvd
CITY	Oxnard
STATE	CA
ZIPCODE	93034
PHONE	(555)456-9059

EDIT	<C: >	CLIENTS	Rec: 1/49
------	-------	---------	-----------

# DBMS의 조상님

## dBASE

- 마이크로컴퓨터용 최초의 DBMS
- 1979년 Ashton이 개발
- SQL이 아닌 독자 스크립트 언어로 실행 -> dbf 파일 생성



# Characteristics

- 데이터의 무결성
- 데이터의 중복 방지
- 보안(추상화, 접근권한)
- 성능 향상
- 프로그램 수정과 유지 보수 용이

# Differences between DataBase & File System


















자기기술성

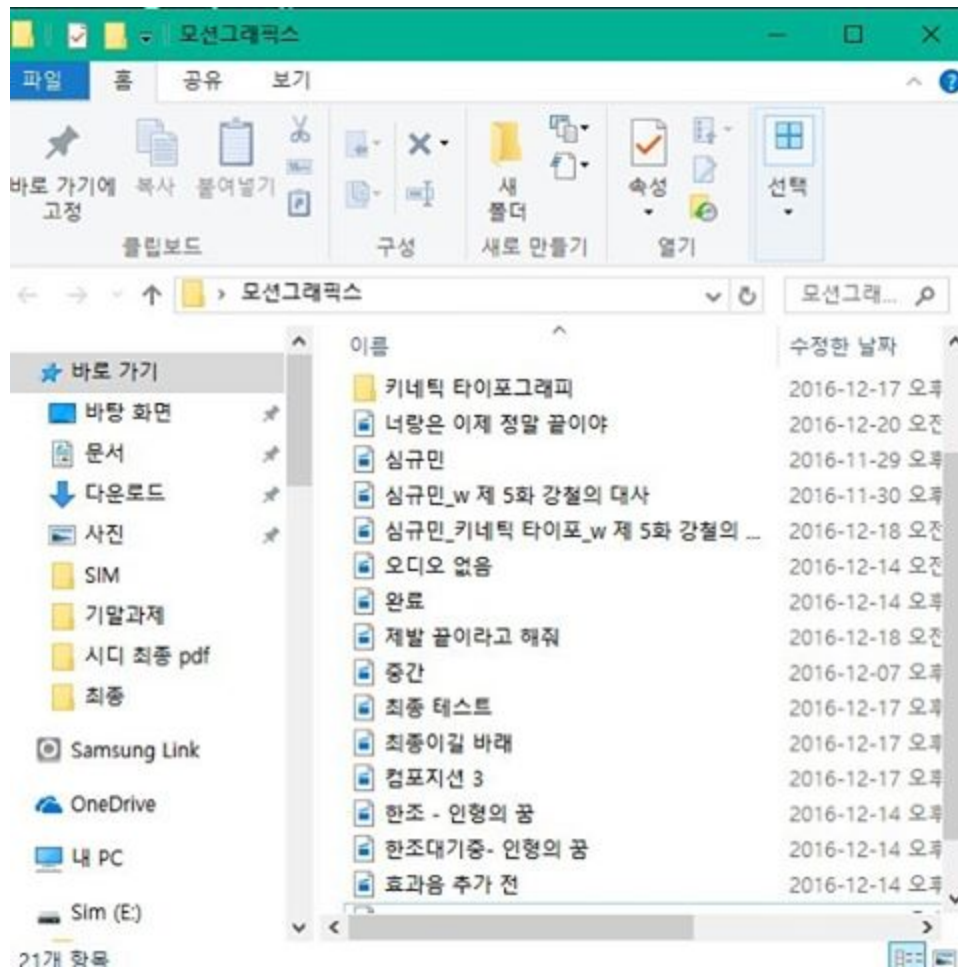
File System

- .hwp -> 한글
- .doc -> Microsoft Word
- .xls -> Microsoft Excel

DB

- Only SQL(RDBMS)

 01\_01\_01\_01.hwp  
 01\_01\_01\_02수정.hwp  
 01\_01\_01\_03수정1.hwp  
 01\_01\_01\_04수정2.hwp  
 01\_01\_01\_05완성본.hwp  
 01\_01\_01\_06완성본1.hwp  
 01\_01\_01\_07완성본2.hwp  
 01\_01\_01\_08최종완성본.hwp  
 01\_01\_01\_09최종완성본1.hwp  
 01\_01\_01\_10최종완성본2.hwp  
 01\_01\_01\_11최종완성본final.hwp  
 01\_01\_01\_12최종완성본final1.hwp  
 01\_01\_01\_13최종완성본final2.hwp  
 01\_01\_01\_14최종완성본final최종.hwp  
 01\_01\_01\_15최종완성본final최종1.hwp  
 01\_01\_01\_16최종완성본final최종2.hwp  
 01\_01\_01\_17유서.hwp



# Activity

## Rule

- Client, Server, DB의 역할을 부여받음
- Client: User의 요청을 Server에 전달하고 Server의 응답을 User(칠판)에 그려냄
- Server: Client의 요청을 DB에 전달하고 적절한 응답을 받아 Client에 전달
- DB: Server의 질문에 적절한 대답을 내놓음

# Activity



# Activity



# SQL(Structured Query Language)

데이터 관리를 위해 설계된 특수 목적의 프로그래밍 언어

UPDATE clause [UPDATE country  
SET clause [SET population = population + 1  
WHERE clause [WHERE name = 'USA';

Expression  
Statement  
Expression  
Predicate

The diagram shows an SQL UPDATE statement with three clauses: UPDATE, SET, and WHERE. The UPDATE clause is 'UPDATE country'. The SET clause is 'SET population = population + 1'. The WHERE clause is 'WHERE name = 'USA';'. Annotations with brackets and lines identify parts of the statement: 'Expression' points to 'population + 1' in the SET clause; 'Statement' points to the entire statement; 'Expression' points to ''USA'' in the WHERE clause; and 'Predicate' points to the entire WHERE clause.



# SQL - 데이터 정의언어

데이터를 정의

CREATE - DB 개체 정의

DROP - DB 개체 삭제

ALTER - DB 개체 정의 변경

# SQL - 데이터 조작언어

데이터 검색, 등록, 삭제, 갱신

INSERT - 행, 테이블 데이터 삽입

UPDATE - 테이블 업데이트

DELETE - 특정 행 삭제

SELECT - 테이블 검색 결과 집합

# SQL - 데이터 제어언어

데이터 액세스 제어

GRANT - 작업 수행권한 부여

REVOKE - 권한 박탈

# Let's Try SQL

[SQL try it editor](#)

# RDBMS vs NoSQL

구분	RDBMS	NoSQL
형태	Table	Key-value, Document, Column
데이터	정형 데이터	비정형 데이터
성능	대용량 처리시 저하	찾은 수정시 저하
스키마	고정	Schemeless
장점	안정적	확장성, 높은 성능
유명	Mysql, MariaDB, PostgreSQL	MongoDB, CouchDB, Redis, Cassandra

# RDBMS

[PostgreSQL Docs](#)

[MariaDB Docs](#)

name	age
John	17
Mary	21

```
rdb =  
{  
    name: [John, Mary],  
    age: [17, 21]  
}
```

Table == Relation

Primary Key	Attribute1	Attr2	Attr3	Attr4
Tuple1				
Tuple2				
Tuple3				
Tuple4				

# NoSQL

## MongoDB Docs

```
nosql =  
[  
    {  
        name: John,  
        age: 17  
    },  
    {  
        name: Mary,  
        age: 21  
    },  
    ...  
]
```



# Document vs Key-value

```
document
{
    key: value,
    key: {
        key: value,
        key: value
    }
}
```

```
key-value
{
    key: value,
    key: value,
    key: value
}
```

# **How to Design Database?**

# Schema

- Database의 구조와 제약조건에 대한 전반적인 명세 기술
- Database의 Blueprint
- 외부(서브)스키마, 개념스키마, 내부스키마로 구성

## 외부(서브) 스키마

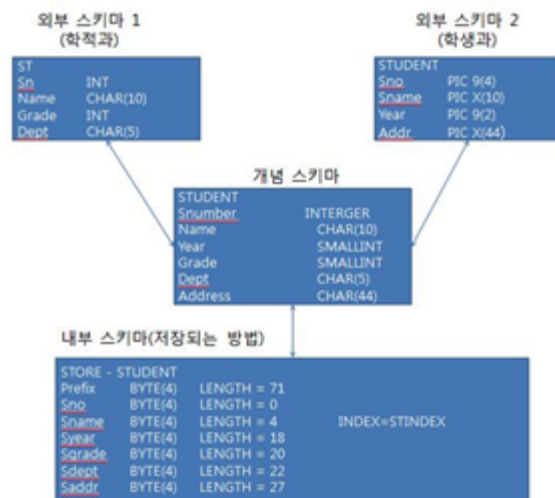
- 프로그램 사용자가 필요로 하는 데이터베이스의 논리적인 구조를 정의

## 개념 스키마

- 조직 전체의 관점에서의 구조와 관계를 정의
- 외부 스키마의 합과 그 사이의 데이터의 관계 등등
- 일반적인 스키마의 정의

## 내부 스키마

- 저장장치의 입장에서 데이터베이스가 저장되는 방법을 기술



# Design Database

Primary Key	Attribute1	Attr2	Attr3	Attr4
Tuple1				
Tuple2				
Tuple3				
Tuple4				



# SQLite

# SQLite with python

## SQLite - check sqlite version

```
$ python  
>> import sqlite3  
>> sqlite3.version  
>> sqlite3.sqlite_version
```

## SQLite - Create table

```
$ sqlite3 users.db  
sqlite> .tables  
sqlite> .exit  
  
$ vi users.db
```

## SQLite - insert data

- `sqlite3.connect` 메소드를 이용해서 DB 파일에 연결한 후 'Connection' 객체를 생성한다.
- Connection 객체를 통해 Cursor 객체를 생성한다.
- 'Cursor' 객체의 `execute` 메소드를 통해서 query를 실행한다.
- 'Connection' 객체의 `commit`를 이용하여 변경된 내용을 commit한다.
- DB와의 연결을 닫는다.