Fastcampus Data Science Extension SCHOOL

SQL(1) - SQL with sqlite3

Database

data

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

Data -> information

data - processing -> information

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, ..

Characteristics

- 데이터의 무결성 유지(정확성,일관성,유효성)
- 데이터의 중복 최소화
- 보안(추상화, 접근권한)
- 성능 향상
- 프로그램 수정과 유지 보수 용이

Differences between DataBase & File System

자기기술성

File System

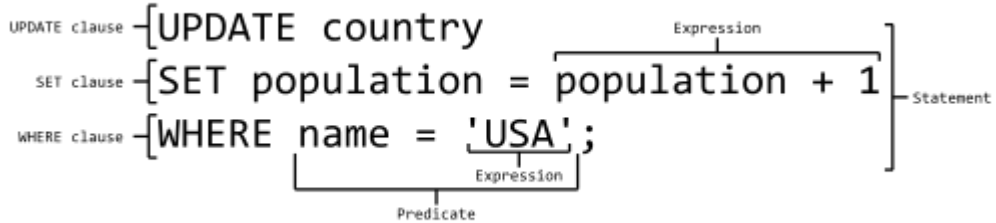
- .hwp -> 한글
- .doc -> Microsoft Word
- .xls -> Microsoft Excel

DB

- Only SQL(RDBMS)

SQL(Structured Query Language)

데이터 관리를 위해 설계된 특수 목적의 프로그래밍 언어



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

RDBMS vs NoSQL

구분	RDBMS	NoSQL
형태	Table	Key-value, Document, Column
데이터	정형 데이터	비정형 데이터
성능	대용량 처리시 저하	잦은 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

`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
}
```

Schema

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

외부(서브) 스키마

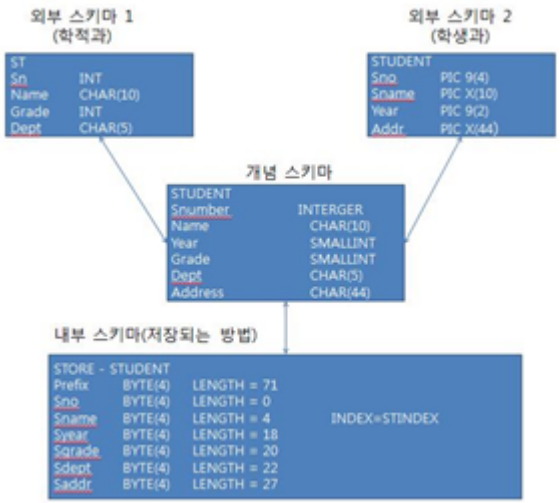
- 프로그램 사용자가 필요로 하는 데이터베이스의 논리적인 구조를 정의

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


외부 스키마 1
(학적과)
ST
Sn INT
Name CHAR(10)
Grade INT
Dept CHAR(5)
외부 스키마 2
(학생과)
STUDENT
Sno PIC 9(4)
Sname PIC X(10)
Year PIC 9(2)
Addr PIC X(44)
개념 스키마
STUDENT
Snumber INTERGER
Name CHAR(10)
Year SMALLINT
Grade SMALLINT
Dept CHAR(5)
Address CHAR(44)
내부 스키마(저장되는 방법)
STORE - STUDENT
Prefix BYTE(4) LENGTH = 71
Sno BYTE(4) LENGTH = 0
Sname BYTE(4) LENGTH = 4 INDEX=STINDEX
Syear BYTE(4) LENGTH = 18
Sgrade BYTE(4) LENGTH = 20
Sdept BYTE(4) LENGTH = 22
Saddr BYTE(4) LENGTH = 27

스키마를 설계해 봅시다

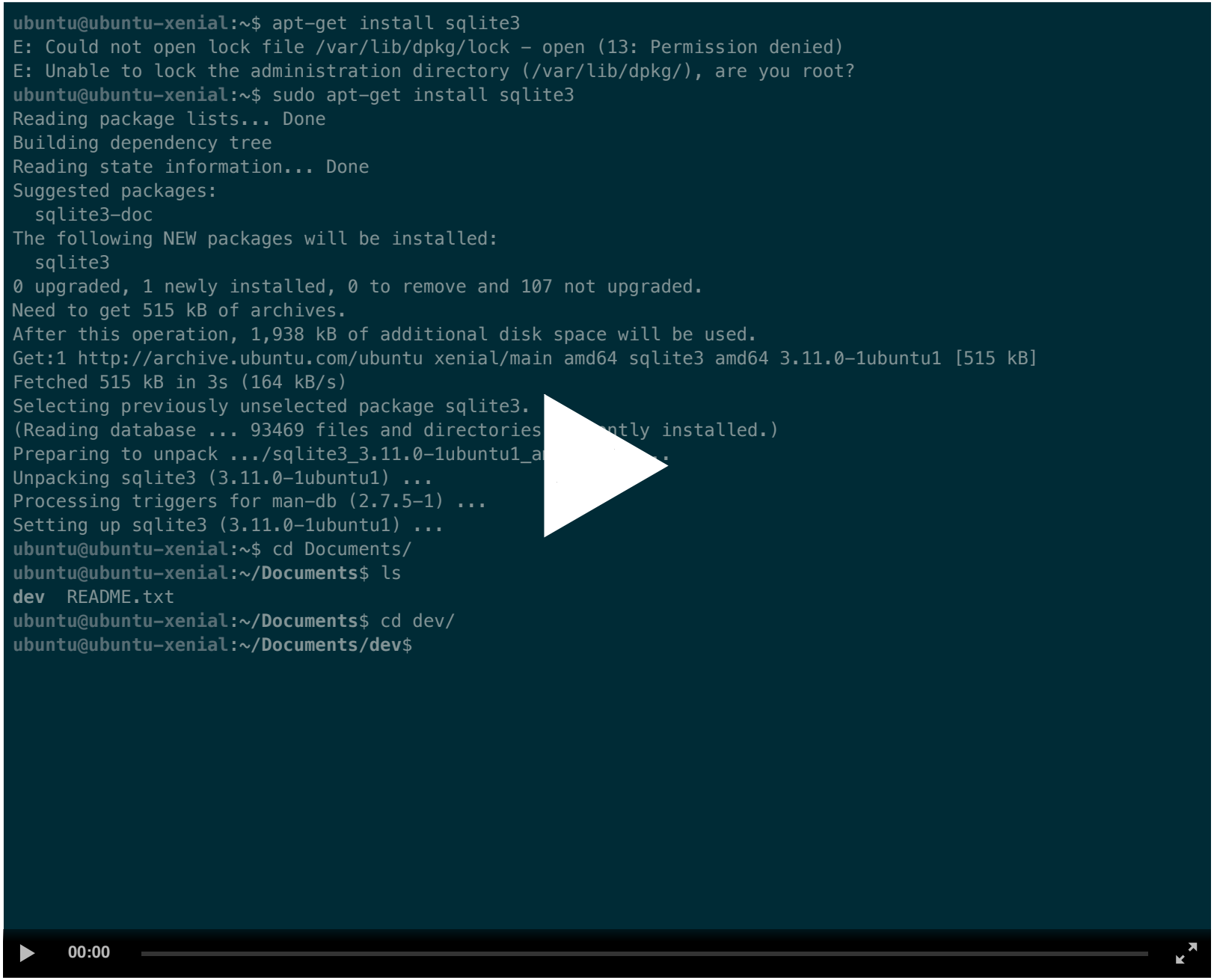
https://store.google.com/

SQLite

SQLite with python

- for windows:
  - https://www.sqlite.org/2018/sqlite-tools-win32-x86-3230100.zip
  - https://www.sqlite.org/download.html
- for mac: pre-installed(after 10.4)
- for linux: sudo apt-get install sqlite3

install sqlite3 on ubuntu



Recorded with asciinema

SQLite - check sqlite version

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

SQLite - Create db

macos, linux

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

windows

```
# double click sqlite3 client
sqlite> .open users.db
sqlite> .tables
sqlite> .exit
```

SQLite commands

```
.open {db_name}
.help

.print {"some_string"}

.databases
.schema {table_name}
.show
.tables
.mode {list|ascii|csv|column|html|tabs}

.exit
.quit
```

Let's learn SQL

SQLite Data type

- TEXT
- INTEGER
- NUMERIC(date, boolean, ..)
- REAL(floating point(8bytes))
- BLOB(stored exactly as it was input)

create and drop table

```
sqlite> CREATE TABLE user(
   ...> id integer primary key,
   ...> name text not null,
   ...> age integer,
   ...> lang text,
   ...> locale text);
sqlite> .tables
```

```
sqlite> DROP TABLE user;
sqlite> .tables
```

modify table

```
sqlite> ALTER TABLE user
   ...> RENAME TO user_old;
sqlite> .tables
```

```
sqlite> ALTER TABLE user
   ...> ADD password text;
sqlite> .tables
```

INSERT & SELECT

```
sqlite> SELECT * FROM user;
sqlite> SELECT id, name FROM user;
```

```
sqlite> INSERT INTO user(id, name, age, lang, locale, password)
   ...> VALUES(...),
   ...> (...),
   ...> (...);
```

WHERE

```
sqlite> SELECT id, name FROM user WHERE {condition};
```

Comparison Operator	Description
= or ==	Equal
<> or !=	Not Equal
>	Greater Than
>=	Greater Than or Equal
<	Less Than
<=	Less Than or Equal

Comparison Operator	Description
IN ( )	Matches a value in a list ex) IN ('a', 'b', 'c')
NOT	Negates a condition
BETWEEN	Within a range (inclusive) ex) BETWEEN a AND b
IS NULL	NULL value
IS NOT NULL	Non-NULL value
LIKE	Pattern matching with % and _ ex) LIKE '%as%' or 'fa_t'
EXISTS	Condition is met if subquery returns at least one row
AND	All of the conditions that must be met
OR	Any of the conditions that must be met

remove column

```
sqlite> begin transaction;
sqlite> create temporary table user_old_backup(
   ...> id, name, age, lang);
sqlite> .tables
temp.user_old_backup  user_old
sqlite> insert into temp.user_old_backup
   ...> select id, name, age, lang from user_old;
sqlite> drop table user_old;

sqlite> create table user(id, name, age, lang);
sqlite> insert into user
   ...> select * from temp.user_old_backup;
sqlite> drop table temp.user_old_backup;
sqlite> commit;
```

import & export data with csv

```
sqlite> .mode csv
sqlite> .import datain.csv another_users
```

```
sqlite> .header on
sqlite> .mode csv
sqlite> .once ./dataout.csv
sqlite> select * from user;

Windows
sqlite> .system dataout.csv

MAC
sqlite> .system open dataout.csv

Linux
sqlite> .system xdg-open dataout.csv
```

SQL Practice with trySQL Editor

https://www.w3schools.com/sql/trysql.asp?filename=trysql_select_all

1. customers의 모든 데이터를 선택하세요
2. customers에서 Country 가 Germany, France 이거나 도시가 Berlin, London 인 ContactName, Country, City를 선택하세요
3. Suppliers에서 Country가 Germany, France, UK 인 모든 데이터를 선택하세요
4. Products에서 Price 가 10 이하인 ProductName, SupplierID를 선택하세요
5. Products에서 Price가 10 이하인 SupplierID를 가지는 Supplier의 SupplierID, City, Country를 선택하세요