Database 기초 다지기

Database 기초

인상민 코치

/* elice */

01

데이터베이스 복습

01 데이터베이스 복습

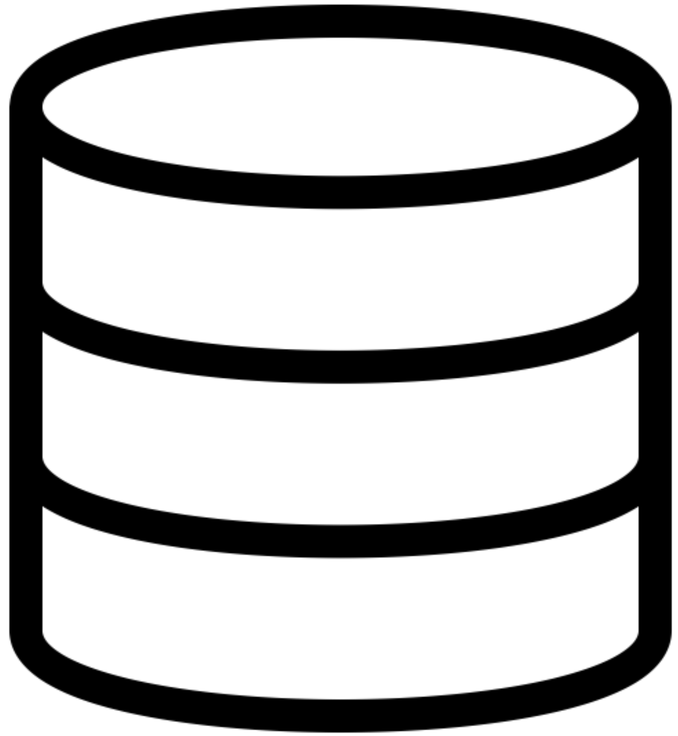
데이터 베이스란?

많은양의 데이터를 관리하기 위해 나온 솔루션

데이터를 여러 컴퓨터에 공유할 수 있다.

데이터를 규격과 정합성을 맞춰 저장할 수 있다.

데이터를 이쁘게 잘 정리하기 위해, 보존하기 위해 사용한다.



01 데이터베이스 복습

SQL이란

Standard Query Language
주로 데이터베이스에서 데이터를 쿼리할 때 사용을 하는 언어

database 종류 상관없이 대부분의 경우 sql을 사용함으로 데이터를 조작할 수 있다.

SQL 의 종류는 DDL, DML, DCL, TCL 로 나누어짐
DDL: Data Definition Language
DML: Data Manipulation Language
DCL: Data Control Language
TCL: Transaction Control Language

01 데이터베이스 복습

DDL이란

Data Definition Language
database나 table, 데이터 구조를 정의하고 수정하는 명령어들

CREATE: db 또는 table을 생성한다.
- CREATE DATABASE database_name;
- CREATE TABLE table_name;

DROP: db 또는 table을 삭제한다.
- DROP database_name l table_name;

ALTER: db 또는 table의 구조를 수정한다.
TRUNCATE: table을 (강력하게) 제거한다.

DML 이란

Data Manipulation Language
table의 데이터를 조작하는 명령어들

INSERT: table에 데이터를 적재한다.
- INSERT INTO table_name(column1, column2) VALUES(value1, value2);

UPDATE: table에 데이터를 수정한다.
- UPDATE table_name SET column1=value1 WHERE @@;

SELECT: table에서 데이터를 꺼내온다.
- SELECT column1 | * FROM table_name WHERE column1=value1;

DELETE: table에서 데이터를 삭제한다.
- DELETE FROM table_name WHERE column1=value1;

01 데이터베이스 복습

db의 종류

01 데이터베이스 복습

sqlite

간단한 작은 RDBMS(Relation DataBase Management System)

파이썬에 이미 내장되어있기에 따로 설치를 하지 않아도 된다.

세팅하기고 사용하기가 매우 간편하다.

여러 컴퓨터에서 접근하지 힘들다.

가벼운 db기에 성능적으로 느리다.

02

파이썬으로 sqlite 사용하기

02 파이썬으로 sqlite 사용하기

설치

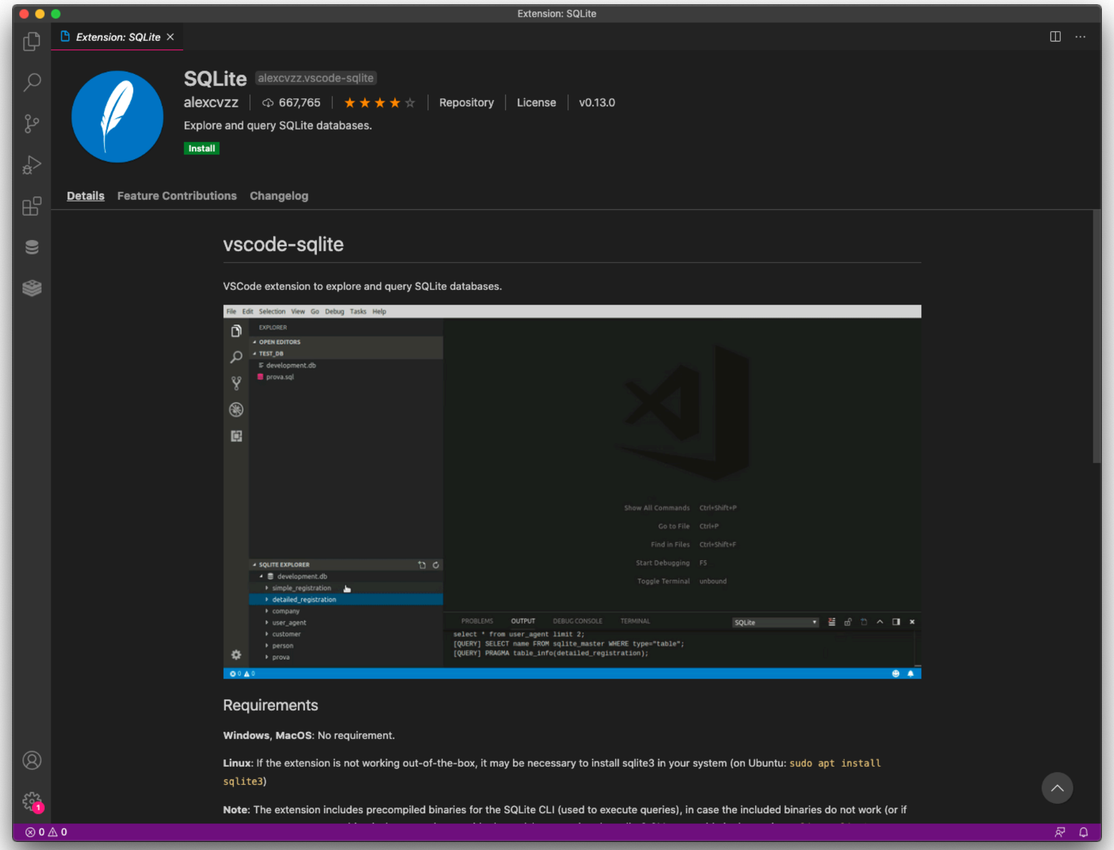
파이썬 자체에 sqlite 가 내장되어 있기에 아래와 같이 import 해서 사용하기만 하면 된다.

```
import sqlite3
connection = sqlite3.connect("db_name.db")
connection.~~~
```

02 파이썬으로 sqlite 사용하기

설치

vs code 에서 SQLite Extension 설치



/* elice */

02 파이썬으로 sqlite 사용하기

db 만들기

```
import sqlite3
con = sqlite3.connect("test.db")
cur = con.cursor()

cur.execute("CREATE TABLE ~~~")
cur.execute("INSERT INTO  ~~~")

cur.execute("SELECT  ~~~")
```

02 파이썬으로 sqlite 사용하기

프로그래밍 언어

데이터베이스

02 파이썬으로 sqlite 사용하기

웹 프레임워크

프로그래밍 언어

데이터베이스

03

ORM, 그러나 sqlalchemy를 곁들인

03 ORM, 그러나 sqlalchemy를 곁들인

ORM

Object Relational Mapper

database 데이터를 python object(class)를 이용해서 데이터를 조작할 수 있다.

장점: 객체지향적인 코드를 작성하여 더 직관적인 코드를 작성할 수 있다.

단점: 라이브러리의 내부 구현에 따라 잘못 동작하여 성능 저하가 발생하는 경우가 있다.

03 ORM, 그러나 sqlalchemy를 곁들인

sqlalchemy 설치하기

```
$ pip3 install sqlalchemy
```

03 ORM, 그러나 sqlalchemy를 곁들인

sqlalchemy 사용해보기

model/_init_.py

```
from sqlalchemy.ext.declarative \
    import declarative_base
Base = declarative_base()
```

model/user_model.py

```
from sqlalchemy import Column, Integer, String, Sequence
from model import Base
class UserModel(Base):
    __tablename__ = "user"
    id = Column(Integer, Sequence("user_id_seq"),primary_key=True)
    username = Column(String(50))
    password = Column(String(50))
    name = Column(String(15))
    email = Column(String(320))
    def __init__(self, username, password, name, email):
        self.username = username
        self.password = password
        self.name = name
        self.email = email
```

03 ORM, 그러나 sqlalchemy를 곁들인

sqlalchemy 사용해보기

main.py

```
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
from model import Base
engine = create_engine("sqlite:///test.db")
Base.metadata.create_all(engine)

Session = sessionmaker(bind=engine)
session = Session()
user = UserModel("username", "1234",
    "sangmin", "me@sangmin.in")
session.add(user)
session.commit()
```

03 ORM, 그러나 sqlalchemy를 곁들인

sqlalchemy 사용해보기

main.py

```
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
from model import Base
engine = create_engine("sqlite:///test.db")
Base.metadata.create_all(engine)

Session = sessionmaker(bind=engine)
session = Session()
user = session.query(UserModel).filter(
    UserModel.username=="username"
).one_or_none()
print(user)
```

```
session.delete(user)
session.commit()

user = session.query(UserModel).filter(
    UserModel.username=="username"
).one_or_none()
print(user)
```

/* elice */

03 ORM, 그러나 sqlalchemy를 곁들인

웹 프레임워크

프로그래밍 언어

ORM

SQLAlchemy

데이터베이스

SQLite

04

db를 이용한 서버 만들기

04 db를 이용한 서버 만들기

사용자 관리 서버

기능: 회원가입 / 로그인 / 비밀번호 변경 / 회원탈퇴
조건: 사용자데이터를 데이터베이스에 저장하고 관리한다!

method, url
- 회원가입: POST /signup
- 로그인: POST /signin
- 비밀번호 변경: PATCH /user/password
- 회원탈퇴: DELETE /user

04 db를 이용한 서버 만들기

TODO List, 사용자 관리를 곁들인

기능: 이전 실습 서버 + TODOLIST
조건: 사용자가 로그인했을 때 해당 사용자의 TODO 만 관리할 수 있어야 한다.

method, url
- TODO 목록: GET /todos?a=a&b=b
- TODO 생성: POST /todo
- TODO 완료 처리: PATCH /todo
- TODO 삭제: DELETE /todo

Credit

/* elice */

콘텐츠 제작자
인상민

강사
인상민

감수자
000

디자인
박주연

Contact

TEL

010-9606-2864

E-MAIL

me@sangmin.in