07/31 Flask 보강 #1

김병철 코치

데이터베이스

데이터들을 모은 일종의 컨테이너

DBMS

그럼 이 컨테이너를 어찌 관리하려고?

DBMS

관계형 (Relation)

데이터를 일종의 “표” 형식으로 저장함.

키-값 형 (KV Store)

모든 데이터를 키와 값의 쌍으로 매핑한다

객체형 (Object)

데이터를 객체처럼 사용한다.

문서형 (Document)

인덱스를 제외하고 아무거나 넣을 수 있다.

SQL

SQL (Structured Query Language)

관계형 데이터베이스에서 자료를 처리하기 위해 사용되는 언어.

NoSQL (비 관계형 데이터베이스)를 제외한 대부분의 데이터베이스에서 사용한다.

왜 NoSQL 인지 아시겠어요?

관계형 (Relation)

데이터를 일종의 "표" 형식으로 저장함.

키-값 형 (KV Store)

모든 데이터를 키와 값의 쌍으로 매핑한다

객체형 (Object)

데이터를 객체처럼 사용한다.

문서형 (Document)

인덱스를 제외하고 아무거나 넣을 수 있다.

몽-고

mongoDB

문서형 DBMS로, JSON 형식의 데이터 구조를 갖고 있음.
SQL의 제약조건에서 벗어나 자유로운 확장성이 특징임.

데이터베이스와 서버

혹시 이런 데이터를 찾아주실래요?

데이터베이스와 서버

검색 결과는 이러네요~

데이터베이스와 서버

서로 분리되어 통신하므로,

서버는 DB와 연결이 되어야 함!

ORM

테이블

객체

ORM

테이블

객체

데이터베이스도 객체로 다루면 좋지 않을까?

ORM

- Object Relation Mapping의 약자.
- 객체와 관계형 데이터베이스의 데이터를 자동으로 매핑 해주는 것
- ORM을 통해 객체의 관계를 인식하여 SQL을 자동 생성함.

또 다른 장점

DBMS를 바꾸고 싶은데…

또 다른 장점

ORM을 통해 코드를 작성했다면,

DBMS가 바뀌어도 **그대로 사용할 수 있다!**

(단, 관계형 DB만)

SQLAlchemy

앞에서 확인한 ORM의 파이썬 버전!

SQL은 빡빡합니다.

기본 키

외래 키

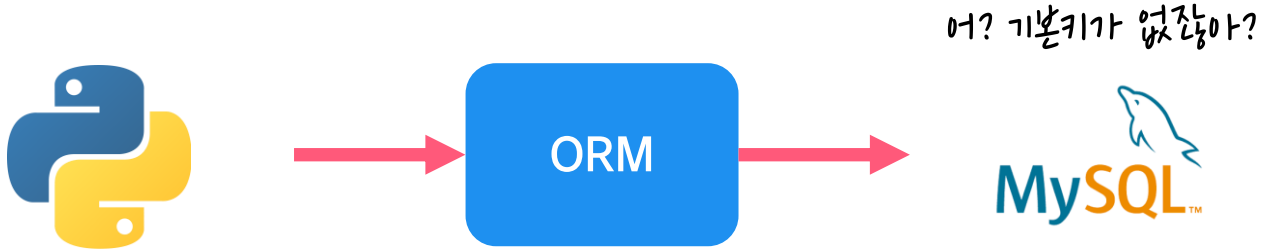
복합 키

제약 조건이 빡빡하다보니…

아무 데이터나

넣어버리면…

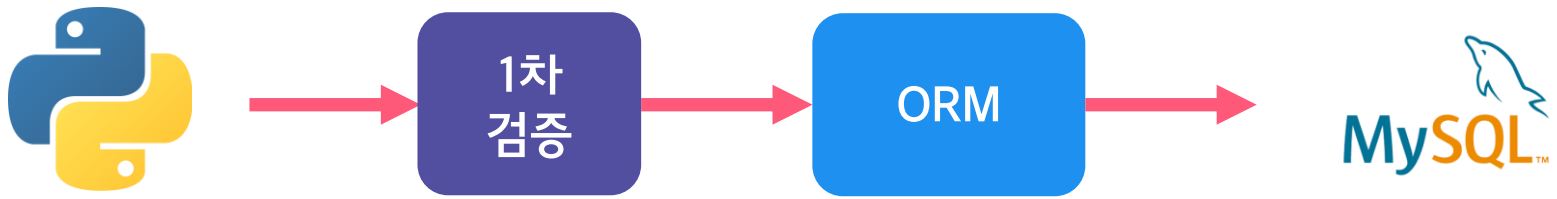
제약 조건이 빡빡하다보니…



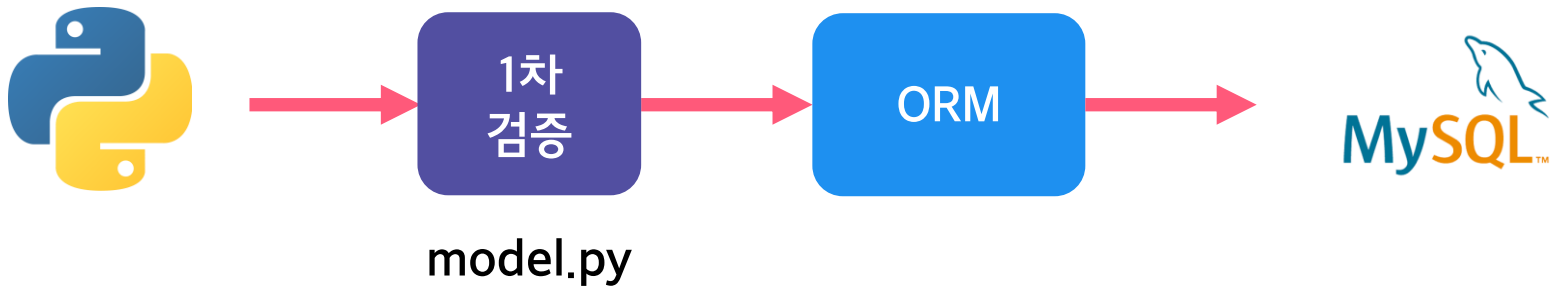
아무 데이터나

넣어버리면…

파이썬에서 미리 조건을 걸자!



파이썬에서 미리 조건을 걸자!



model.py

#1

request.form

and_, or_

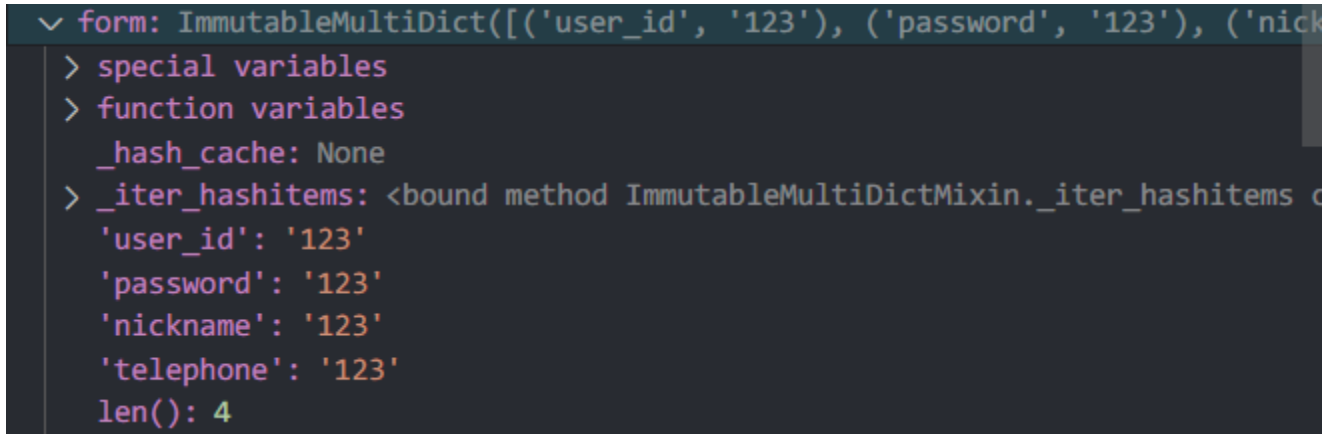
클라이언트가 request를 보내면,

URL로 request 요청을 보냄

flask를 이걸 받아요.

```
from flask import Blueprint, render_template, request, url_for, session, flash
from rabbit_delivery.models import *
from werkzeug.utils import redirect
```

form에는 html form 정보가 담깁니다.



참고: name 속성의 이름으로 저장됩니다.

#2, #3

db.session

flask 첫 코드를 보면…

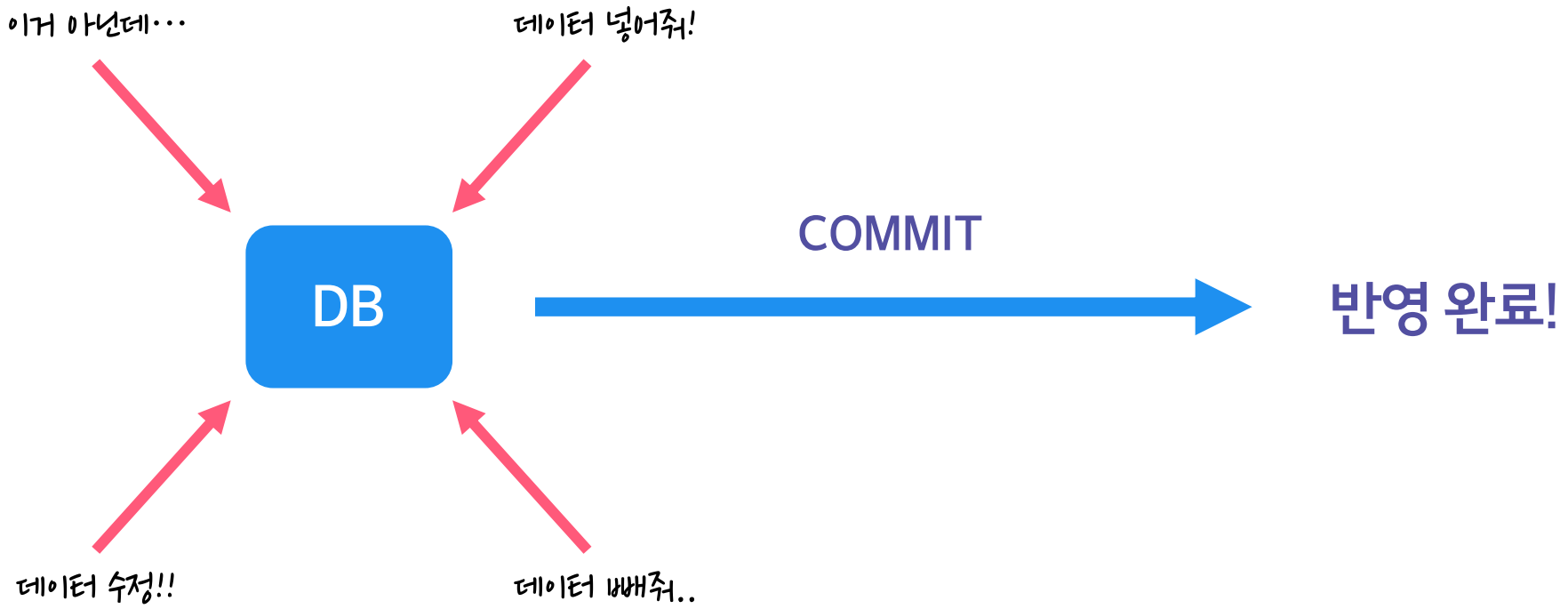
```
db = SQLAlchemy()
db.init_app(app)
```

db는 안 죽고 살아있습니다.

혹시… 트랜잭션 기억하세요?

트랜잭션(Transaction)은 데이터베이스의 상태를 변환시키는
하나의 **논리적 기능**을 수행하기 위한 **작업의 단위** 또는
한꺼번에 모두 수행되어야 할 **일련의 연산**들을 의미한다.

db.session도 트랜잭션 단위로!



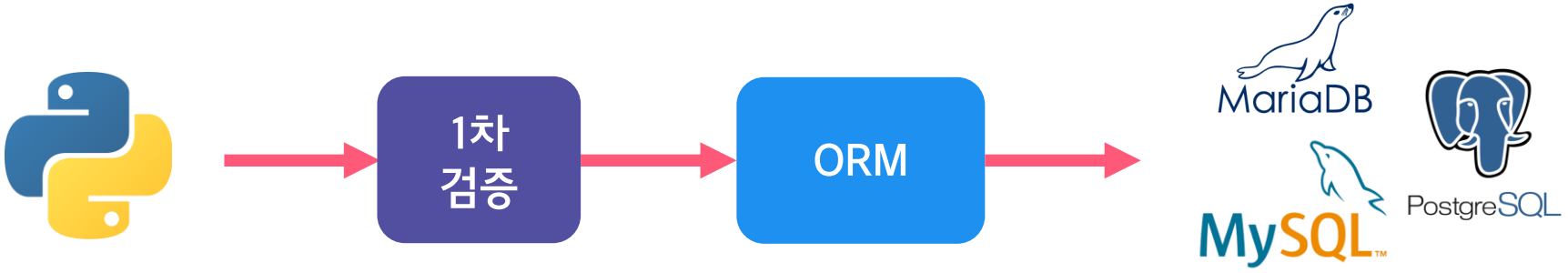
ORM의 약자가 뭐게요?

- Object Relation Mapping의 약자.
- 객체와 관계형 데이터베이스의 데이터를 자동으로 매핑 해주는 것
- ORM을 통해 객체의 관계를 인식하여 SQL을 자동 생성함.

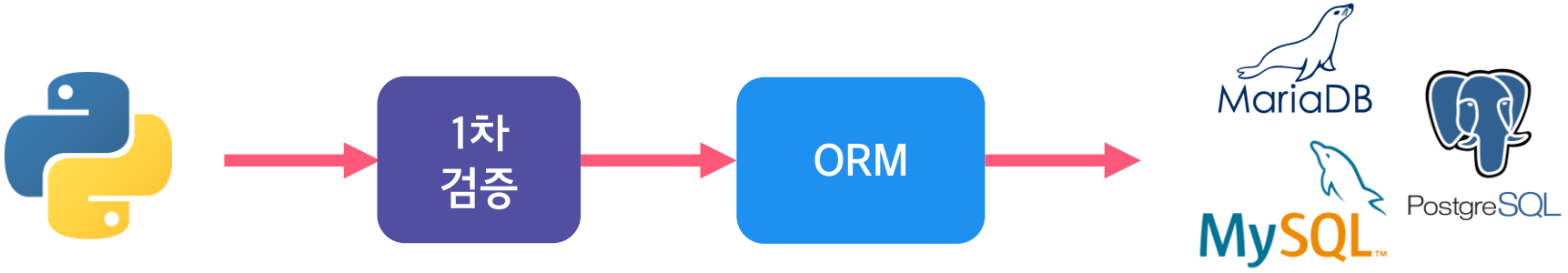
mongoDB는 ORM이 없다!

- Object Relation Mapping의 약자.
- 객체와 관계형 데이터베이스의 데이터를 자동으로 매핑 해주는 것
- ORM을 통해 객체의 관계를 인식하여 SQL을 자동 생성함.

pymongo VS SQLAlchemy



pymongo VS SQLAlchemy



- SQLAlchemy의 메서드는 실제 SQL 문법과는 많은 차이가 있음.
- pymongo는 mongoDB와 명령어가 매우 유사함.

#4

ObjectId

find_one, update_one,
delete_one

find_one 과 delete_one

```
query = {"_id" : ObjectId(_id)}
movie_info = col.find_one(query)
```

mongoDB랑 똑같이 동작한다!

ObjectId

기본 키

???

없어도 되냐고요?

기본 키

```
id INTEGER PRIMARY KEY (UNIQUE NOT NULL),
```

UNIQUE 덕분에!

실수로 똑같은 데이터를 여러 번 올림

UNIQUE 덕분에!

실수로 똑같은 데이터를 여러 번 올림

UNIQUE 제한 조건 때문에 막힘!

따라서 ObjectId 라는게 있어요.

Pymongo에서 ObjectId 가져오기

```python
# Netflix 작품 하나의 상세 정보를 출력합니다.
@app.route("/netflix/<_id>", methods=['GET'])
def show(_id):
    netflix = col.find({"_id": ObjectId(_id)})[0]
    return render_template('show.html', netflix=netflix)
```

update_one

바꿀 값을 찾고, 원하는 값으로 변경한다!

update_one

바꿀 값을 찾고, 원하는 값으로 변경한다!

find_one과 동일한 방식

이런식으로 코드를 짜면?

```
prev_data = {
    "_id" : ObjectId(대충 아이디),
    "name" : "토끼",
    "like" : "당근",
    "age" : 22
}
```

```
def update(_id):
    query = {"_id" : ObjectId(_id)}
    change_query = {"name" : "주먹도끼"}
    col.update_one(query, change_query)
```

짠!

```
prev_data = {
    "name" : "주먹도끼"
}
```

```
def update(_id):
    query = {"_id" : ObjectId(_id)}
    change_query = {"name" : "주먹도끼"}
    col.update_one(query, change_query)
```

그래서 안 날아가게 하기 위해서…

```
def update(_id):
    query = {"_id" : ObjectId(_id)}
    change_query = {"name" : "주먹도끼"}
    col.update_one(query, {“$set" : change_query})
```

#5

count