

2장. 데이터를 관리하는 모델



목 차

1

migrate와 모델 만들기

2

데이터 만들고 저장하고 조회하기

3

장고 관리자(Admin) 사용하기

데이터베이스와 테이블 생성

◆ 장고 서버 구동시 경고 메시지

```
(mysite) c:\webproject\polls>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly
contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
```

◆ migrate 명령으로 앱 설정

```
(mysite)C:\projects\polls>python manage.py migrate
```

데이터베이스와 테이블 생성

◆ migrate 명령으로 앱이 필요로 하는 테이블 생성

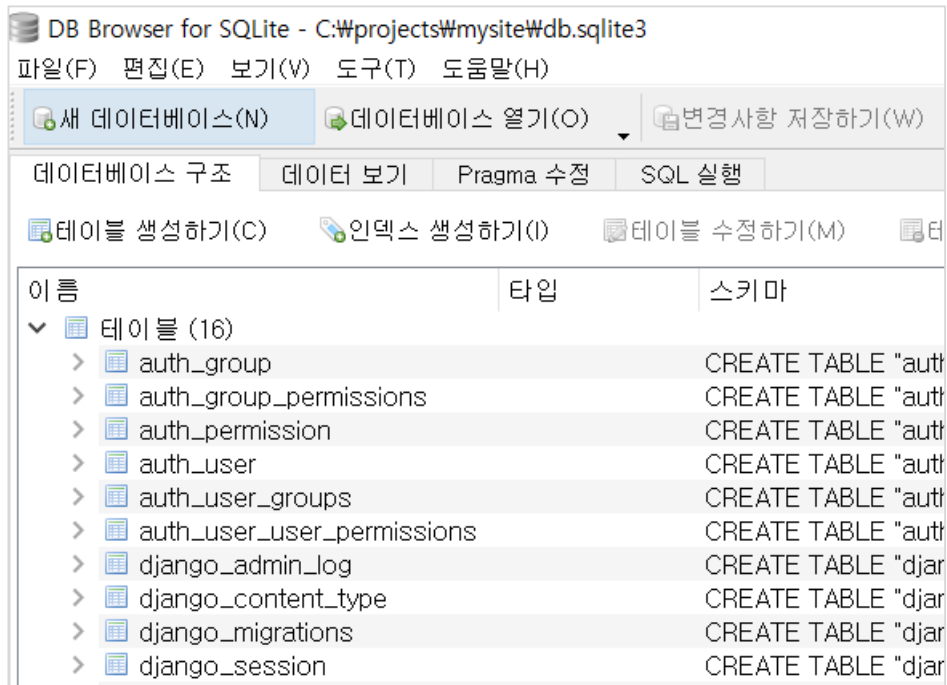
```
(mysite) c:\Webproject\polls>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
```

config/setting.py 에 'poll'앱 등록하기

```
INSTALLED_APPS = [
    'poll.apps.PollConfig',
    'django.contrib.admin',
    'django.contrib.auth',
]
```

데이터베이스와 테이블 생성

◆ DB Browser for SQLite로 테이블 살펴보기



C:\projects\polls\db.sqlite3

ORM(Object Relational Mapping)

◆ ORM(Object Relational Mapping) – 객체 관계 매핑 방식

파이썬은 현재 가장 많이 사용하고 있는 관계형 데이터베이스 방식을 개선한 ORM 방식을 사용한다.

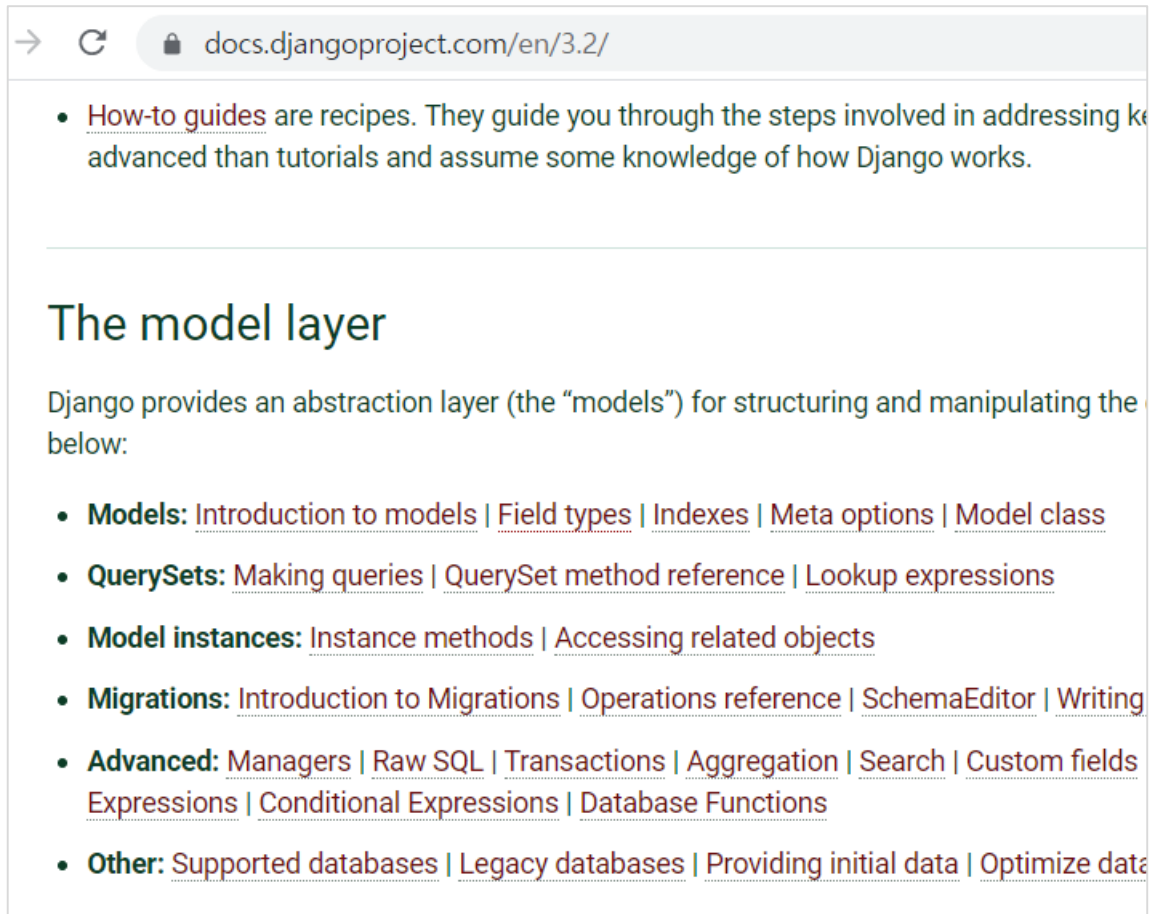
테이블을 모델화(클래스의 객체)하여 DB를 획기적으로 관리할 수 있으며, 특히 SQL 언어의 문법을 사용하지 않는다.

SQL을 사용했을 경우 단점은

- SQL 쿼리문은 개발자마다 다양하게 작성되므로 일관성이나 유지 보수 측면에서 심플하지 않고 복잡할 수 있다.
- 또한, 개발자가 쿼리문 자체를 잘못 작성하여 시스템의 성능이 저하 될 수 있다.

ORM(Object Relational Mapping)

◆ Django Documentation



The screenshot shows a web browser window with the URL `docs.djangoproject.com/en/3.2/`. The page content includes a list of links for 'How-to guides' and a section titled 'The model layer' which provides an overview of Django's abstraction layer and lists various topics for further exploration.

→ ↻ 🔒 docs.djangoproject.com/en/3.2/

- [How-to guides](#) are recipes. They guide you through the steps involved in addressing key tasks. They are more advanced than tutorials and assume some knowledge of how Django works.

The model layer

Django provides an abstraction layer (the “models”) for structuring and manipulating the database. The following sections provide more information about the model layer:

- **Models:** [Introduction to models](#) | [Field types](#) | [Indexes](#) | [Meta options](#) | [Model class](#)
- **QuerySets:** [Making queries](#) | [QuerySet method reference](#) | [Lookup expressions](#)
- **Model instances:** [Instance methods](#) | [Accessing related objects](#)
- **Migrations:** [Introduction to Migrations](#) | [Operations reference](#) | [SchemaEditor](#) | [Writing migrations](#)
- **Advanced:** [Managers](#) | [Raw SQL](#) | [Transactions](#) | [Aggregation](#) | [Search](#) | [Custom fields](#) | [Expressions](#) | [Conditional Expressions](#) | [Database Functions](#)
- **Other:** [Supported databases](#) | [Legacy databases](#) | [Providing initial data](#) | [Optimize data](#)

ORM(Object Relational Mapping)

◆ Django Documentation

This example model defines a **Person**, which has a **first_name** and **last_name**:

```
from django.db import models

class Person(models.Model):
    first_name = models.CharField(max_length=30)
    last_name = models.CharField(max_length=30)
```

first_name and **last_name** are **fields** of the model. Each field is specified as a class column.

The above **Person** model would create a database table like this:

```
CREATE TABLE myapp_person (
    "id" serial NOT NULL PRIMARY KEY,
    "first_name" varchar(30) NOT NULL,
    "last_name" varchar(30) NOT NULL
);
```


ORM(Object Relational Mapping)

◆ poll 모델 만들기

1. poll/models.py에 Question 모델 생성

```
from django.db import models

class Question(models.Model):
    question_text = models.CharField(max_length=200)
    pub_date = models.DateTimeField('date published')

    def __str__(self):
        return self.question_text
```

ORM(Object Relational Mapping)

◆ Poll 모델 만들기

2. poll/models.py에 Choice모델 생성

```
from django.db import models

class Choice(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    choice_text = models.CharField(max_length=200)
    votes = models.IntegerField(default=0)

    def __str__(self):
        return self.choice_text
```

데이터베이스와 테이블 생성

◆ makemigrations 명령으로 테이블 관리 시작하기

```
(mysite)C:\projects\polls>python manage.py makemigrations
```

◆ makemigrations 후 다시 migrate 명령 실행

```
(mysite)C:\projects\polls>python manage.py migrate
```

```
operations = [  
    migrations.CreateModel(  
        name='Question',  
        fields=[  
            ('id', models.BigAutoField(auto_created=True, primary_key=True,  
            ('subject', models.CharField(max_length=200)),  
            ('content', models.TextField()),  
            ('create_date', models.DateTimeField()),  
        ],  
    ),  
]
```

migrations/0001.initial.py

데이터 저장 및 조회하기

◆ 장고 셸 실행하기

```
(mysite)C:\projects\polls>python manage.py shell
```

◆ 데이터 입력 – Question 모델의 객체 생성

```
>>> from poll.models import Question, Choice
>>> from django.utils import timezone
>>> Question.objects.all()
<QuerySet []>
>>> q = Question(question_text= " 접종한 백신은 무엇인가요?", pub_date=timezone.now())
>>> q.save()
>>> q.id
1
>>> q.question_text
"What's up?"
>>> q.pub_date
datetime.datetime(2021, 7, 16, 13, 13, 45, 965555, tzinfo=<UTC>)
```

데이터 저장 및 조회하기

◆ 데이터 조회 – Question 모델의 객체 생성

```
>>> Question.objects.all() : 모두 조회(리스트)
```

```
>>> Question.objects.filter(id=1) : 1개 조회(리스트)
```

```
>>> Question.objects.get(id=1) : 1개의 데이터 조회
```

```
>>> quit() # 셸을 나갈때 사용
```

데이터 변경 및 삭제하기

◆ 데이터 수정

```
>>>q = Question.objects.get(id=1)
>>>q.subject = '접종받은 백신의 종류는?'
>>>q.save()
```

◆ 데이터 삭제

```
>>>q = Question.objects.get(id=1)
>>>q.delete()
```

데이터 변경 및 삭제하기

◆ 데이터 입력 – Choice 모델의 객체 생성

`q = Question.objects.get(id=1)` : 1개의 질문 가져오기

`a = Choice(question=q, choice_text='화이자', votes=0)`

`a.save()`

`a = Choice(question=q, choice_text='얀센', votes=0)`

`a.save()`

`a = Choice(question=q, choice_text='모더나', votes=0)`

`a.save()`

`q.choice_set.all()` : 연결된 데이터로 조회하기

`<QuerySet [<Choice: 화이자>, <Choice: 얀센>, <Choice: 모더나>]>`

장고 Admin

◆ 슈퍼유저 생성하기

```
(mysite)C:\projects\polls>python manage.py createsuperuser
```

사용자 이름 : admin

이메일 주소 :

Password:*****

Password(again):*****

장고 Admin

◆ localhost:8000/admin 에 접속하기

Django 관리

사용자 이름:

비밀번호:

로그인

admin / 12345

Django 관리

사이트 관리

POLL

Choices + 추가 ✎ 변경

Questions + 추가 ✎ 변경

인증 및 권한

그룹 + 추가 ✎ 변경

사용자(들) + 추가 ✎ 변경

장고 Admin

- ◆ 장고 Admin에서 모델 관리하기
 - poll/admin.py 에 등록

```
from django.contrib import admin
from .models import Question, Choice

admin.site.register(Question)
admin.site.register(Choice)
```

장고 Admin

◆ Question 추가하기

변경할 question 선택

액션: 3 중 아무것도 선택되지 않았습니다.



<input type="checkbox"/>	QUESTION
<input type="checkbox"/>	당신이 좋아하는 과일은 무엇인가요?
<input type="checkbox"/>	코로나 백신 중 원하는 제품을 고르세요.
<input type="checkbox"/>	What's up?

3 questions

장고 Admin

◆ Choice 추가하기

choice 추가

Question:	<div>-----</div> <div>What's up? 코로나 백신 중 원하는 제품을 고르세요. 당신이 좋아하는 과일은 무엇인가요?</div>	 
Choice text:		
Votes:	<div>0</div>	

choice 추가

Question:	<div>코로나 백신 중 원하는 제품을 고르세요. ▾</div>  	
Choice text:	<div>1. 얀센 2. 화이자 3. 모더나</div>	
Votes:	<div>0</div>	