

파이썬 웹 개발부터 배포까지!

# 점프 투 장고



## 장고의 기본 요소 익히기!

# INDEX

- 02-1 주소와 화면을 연결하는 URL과 뷰
- 02-2 데이터를 관리하는 모델
- 02-3 개발 편의를 제공하는 장고 Admin
- 02-4 질문 목록과 질문 상세 기능 구현하기
- 02-5 URL 더 똑똑하게 사용하기



장고의 기본 요소 익히기!

# INDEX

- 02-6 답변 등록 기능 만들기
- 02-7 화면 예쁘게 꾸미기
- 02-8 부트스트랩으로 더 쉽게 화면 꾸미기
- 02-9 표준 HTML과 템플릿 상속 사용해 보기
- 02-10 질문 등록 기능 만들기



## 장고의 기본 요소 익히기!

### 이장의 목표

- ✓ urls.py 파일을 이용해 URL과 매핑되는 뷰 함수를 관리한다.
- ✓ 장고 ORM을 이용해 데이터베이스를 제어한다.
- ✓ 파이보 게시판에 질문 목록과 질문 상세 기능을 만든다.

## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

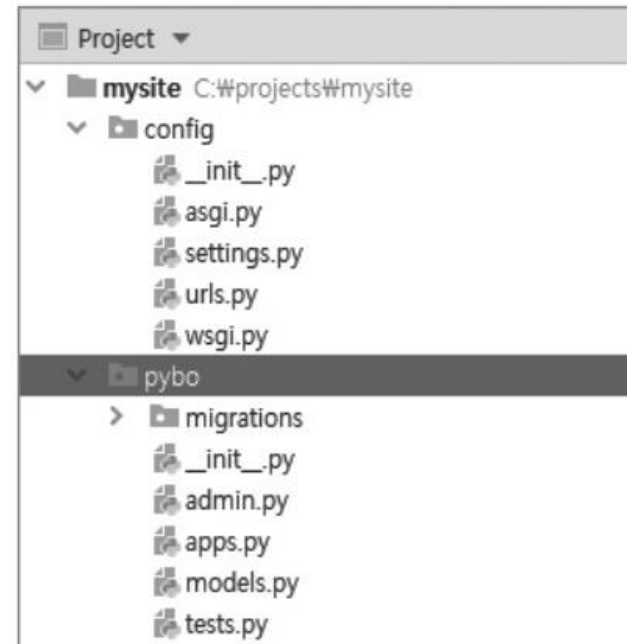
### 앱 생성하고 확인하기

#### 01단계 pybo 앱 생성하기

```
C:\> 명령 프롬프트
(mysite) C:\projects\mysite>django-admin startapp pybo
(mysite) C:\projects\mysite>
```

- ▶ 명령 프롬프트에서 django-admin의 startapp 명령을 이용하여 pybo 앱을 생성하자.

#### 02단계 생성된 앱 확인하기



pybo 디렉터리를 펼친 모습

## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

안녕하세요 파이보?

### 01단계 개발 서버 구동하기

C:\ 명령 프롬프트

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

### 02단계 localhost:8000/pybo에 접속하기

localhost:8000/pybo

- ▶ 앞으로 이 과정을 '페이지를 요청한다' 또는 '/pybo/를 요청한다'라고 할 것이다.

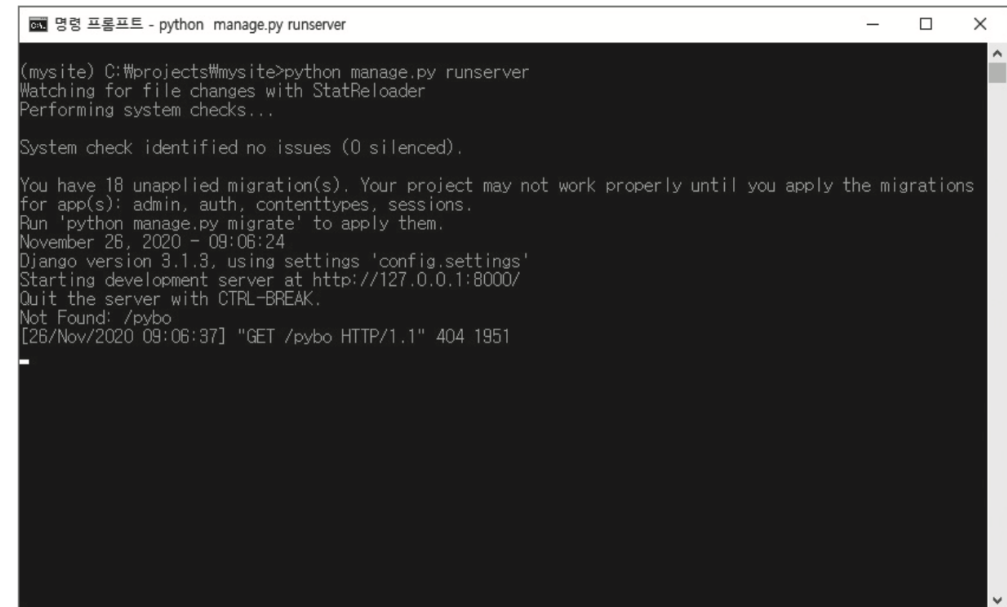
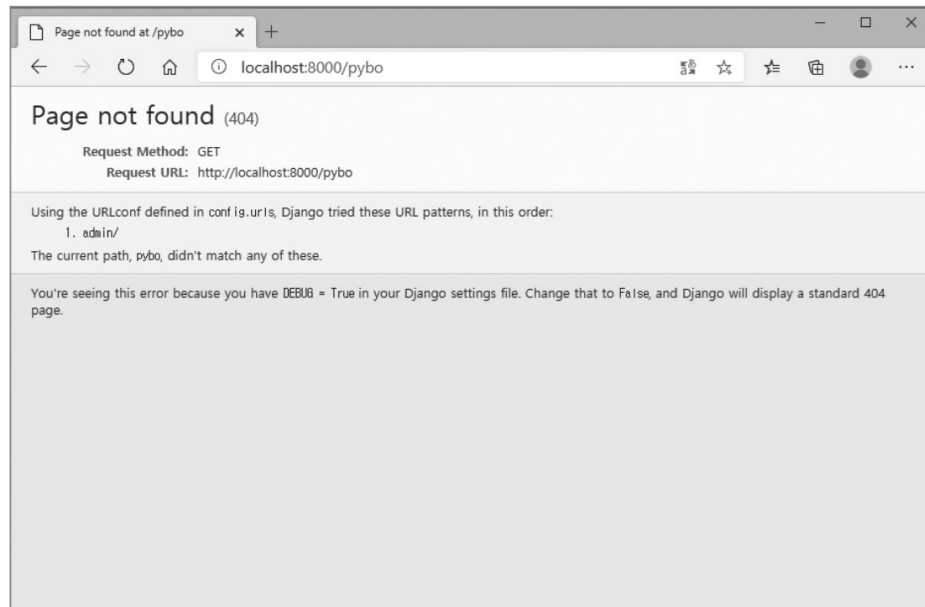
## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

안녕하세요 파이보?

### 03단계 오류 메시지 확인하기



- ▶ 아직 `/pybo/` 페이지에 해당하는 URL 매핑을 장고에 등록하지 않아서 `/pybo/` 페이지를 찾을 수 없다고 메시지를 보낸 것이다.

## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

안녕하세요 파이보?

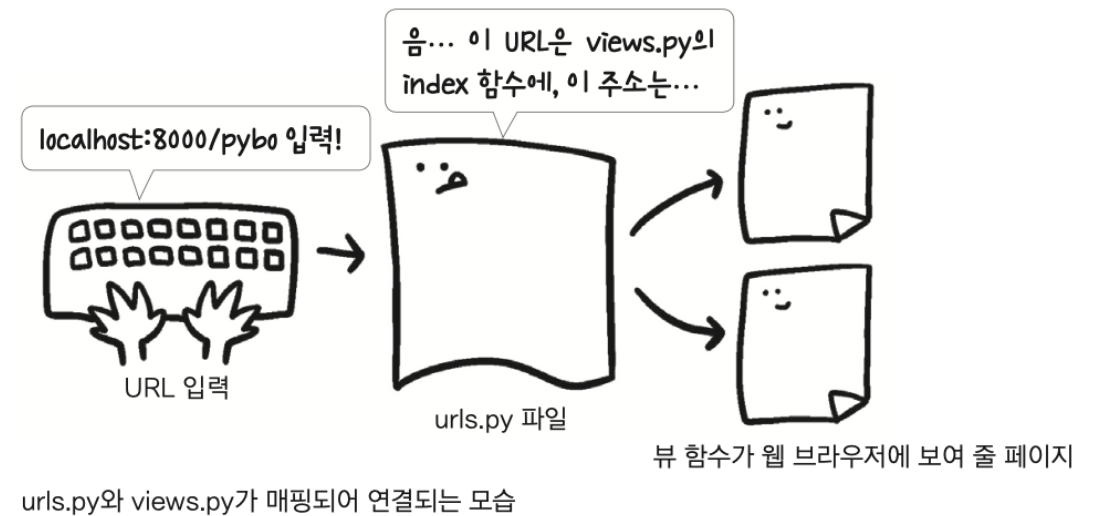
### 04단계 config/urls.py 수정하기

파일 이름 C:/projects/mysite/config/urls.py

```
from django.contrib import admin
from django.urls import path
from pybo import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('pybo/', views.index),
]
```

- ▶ 앞으로 이 과정을 'URL 매핑을 추가한다'고 말할 것이다.
- ▶ path 함수를 사용하여 pybo/URL과 views.index를 매핑했다.





## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

안녕하세요 파이보?

### 05단계 config/urls.py 다시 살펴보기

Urls.py 파일의 urlpatterns

```
urlpatterns = [  
    path('admin/', admin.site.urls),  
    path('pybo/', views.index),  
]
```

- ▶ urlpatterns에는 호스트명과 포트를 입력하지 않는다.
- ▶ 슬래시 /를 붙이면 사용자가 슬래시 없이 주소를 입력해도 장고가 자동으로 슬래시를 붙여 준다.

### 06단계 오류 메시지 확인하기

C:\ 명령 프롬프트

```
(... 생략 ...)  
File "C:\projects\mysite\config\urls.py", line 23, in <module>  
    path('pybo/', views.index),  
AttributeError: module 'pybo.views' has no attribute 'index'
```

- ▶ config/urls.py 파일을 수정했음에도 이런 오류가 발생하는 이유는 URL 매핑에 추가한 뷰 함수 `views.index`가 없기 때문이다.

## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

안녕하세요 파이보?

### 07단계 pybo/views.py 작성하기

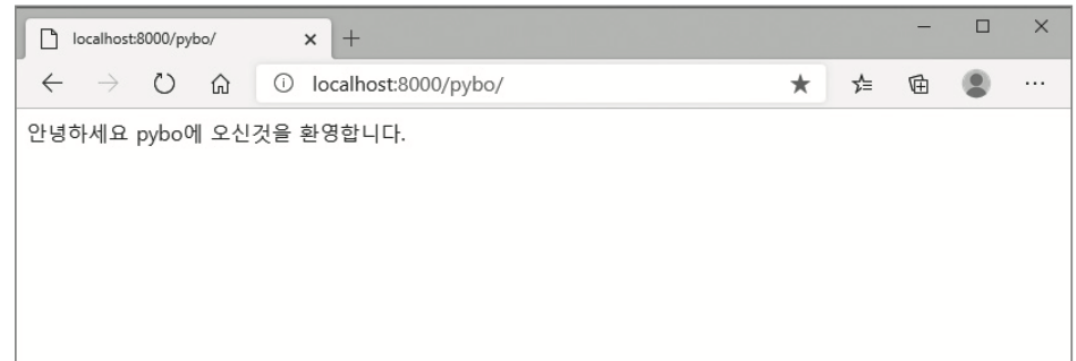
```
파일 이름 C:/projects/mysite/pybo/views.py

from django.http import HttpResponse

def index(request):
    return HttpResponse("안녕하세요 pybo에 오신것을 환영합니다.")
```

- ▶ return문에 사용된 HttpResponse는 페이지 요청에 대한 응답을 할 때 사용하는 장고 클래스이다.

### 08단계 첫 번째 장고 프로그램 완성!

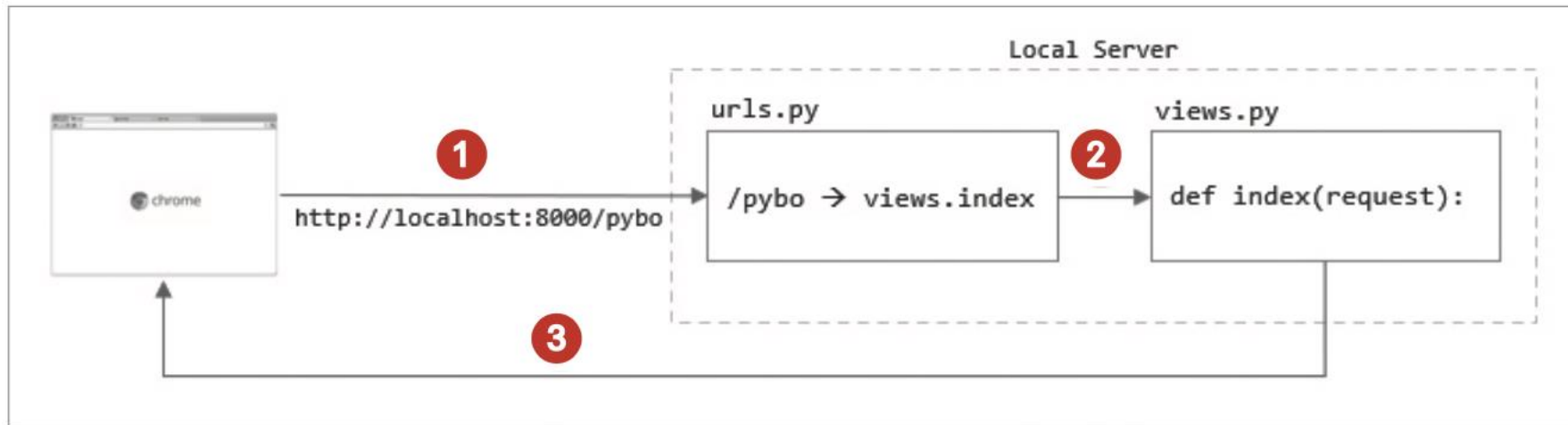


## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

### 장고 개발 흐름 정리하기



- 1 웹 브라우저 주소창에 localhost:8000/pybo 입력(장고 개발 서버에 /pybo/ 페이지 요청).
- 2 config/urls.py 파일에서 URL을 해석해 pybo/views.py 파일의 index 함수 호출.
- 3 pybo/views.py 파일의 index 함수를 실행해 함수 실행 결과를 웹 브라우저에 전달.

## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

### URL 분리하기

#### 01단계 config/urls.py 다시 살펴보기

파일 이름 C:/projects/mysite/config/urls.py

```
from django.contrib import admin
from django.urls import path
from pybo import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('pybo/', views.index),
]
```

#### 02단계 config/urls.py 수정하기

파일 이름 C:/projects/mysite/config/urls.py

```
from django.contrib import admin
from django.urls import path, include
from pybo import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('pybo/', include('pybo.urls')),
]
```

config 디렉터리의 urls.py 파일명

- ▶ `path('pybo/', include('pybo.urls'))`는 `pybo/`로 시작되는 페이지 요청은 모두 `pybo/urls.py` 파일에 있는 URL 매핑을 참고하여 처리하라는 의미이다.

## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

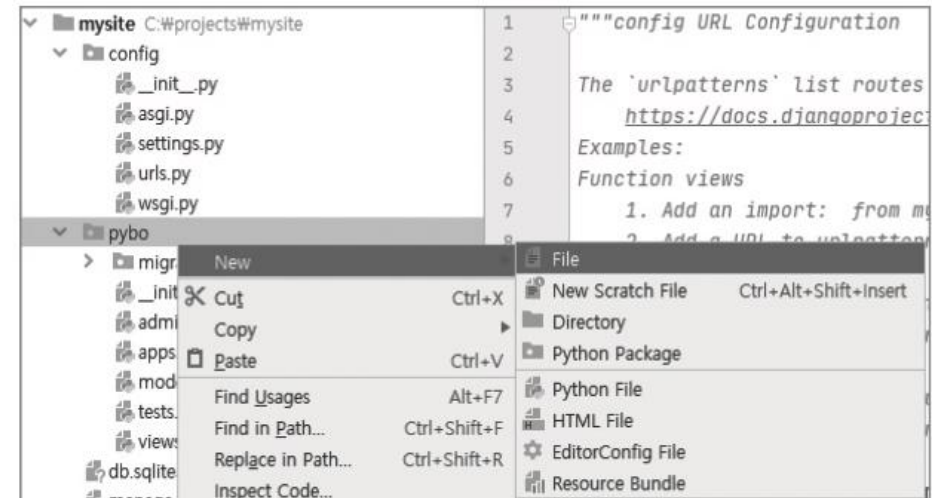
Do it!  
실습

### URL 분리하기

URL 요청	요청 처리 파일
pybo/question/create/	pybo/urls.py
pybo/answer/create/	pybo/urls.py
그외	config/urls.py

- ▶ 다음 예와 같이 pybo/로 시작하는 요청은 config/urls.py 파일이 아닌 pybo/urls.py 파일을 통해 처리하게 된다.

### 03단계 pybo/urls.py 생성하기



## 02-1 주소와 화면을 연결하는 URL과 뷰

• 완성 소스 [github.com/pahkey/djangobook/tree/2-01](https://github.com/pahkey/djangobook/tree/2-01)

Do it!  
실습

### URL 분리하기

#### 04단계 pybo/urls.py 수정하기

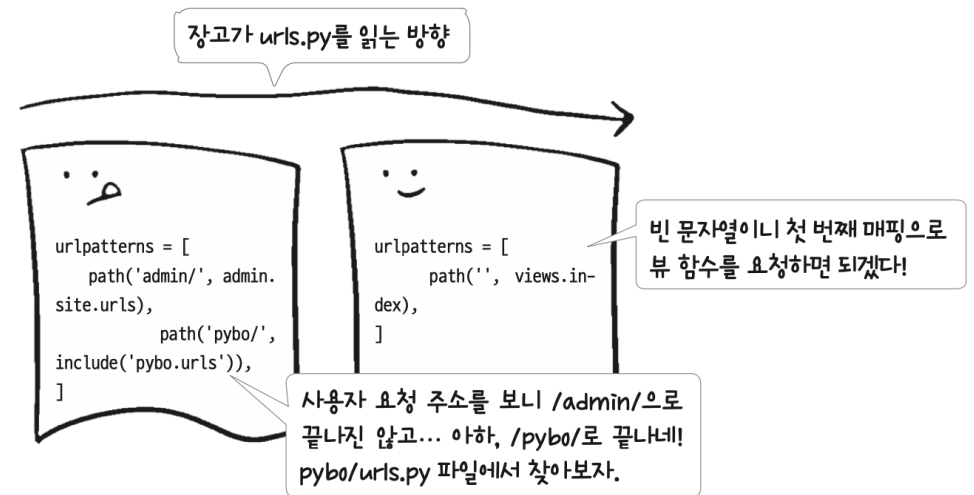
파일 이름 C:/projects/mysite/pybo/urls.py

```
from django.urls import path

from . import views

urlpatterns = [
    path('', views.index),
]
```

- ▶ pybo/에 대한 처리를 한 상태에서 pybo/urls.py 파일이 실행되므로 첫 번째 매개변수에 pybo/가 아닌 빈 문자열('')을 인자로 넘겨준 것이다.



URL 매핑을 확인하는 과정

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### migrate와 테이블 알아보기

#### 01단계 장고 개발 서버 구동 시 나오는 경고 메시지 살펴보기

```
C:\ 명령 프롬프트

(mysite) C:\projects\mysite>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 until you apply
the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
May 06, 2020 - 09:49:37
Django version 3.1.3, using settings 'config.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

아직 적용되지 않은 18개의 migration이 있음!

#### 02단계 config/settings.py 열어 기본으로 설치된 앱 확인하기

```
파일 이름 C:/projects/mysite/config/settings.py

(... 생략 ...)
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
(... 생략 ...)
```

▶ INSTALLED\_APPS는 현재 장고 프로젝트에 설치된 앱이다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### migrate와 테이블 알아보기

#### 03단계 config/settings.py에서 데이터베이스 정보 살펴보기

파일 이름 C:/projects/mysite/config/settings.py

```
(... 생략 ...)
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}
(... 생략 ...)
```

- ▶ config/settings.py파일에는 설치된 앱뿐만 아니라 사용하는 데이터베이스에 대한 정보도 정의되어 있다.

#### 04단계 migrate 명령으로 앱이 필요로 하는 테이블 생성하기

C:\ 명령 프롬프트

```
(mysite) C:\projects\mysite>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

Applying admin.0003\_logentry\_add\_action\_flag\_choices... OK

Applying contenttypes.0002\_remove\_content\_type\_name... OK

Applying auth.0002\_alter\_permission\_name\_max\_length... OK

Applying auth.0003\_alter\_user\_email\_max\_length... OK

Applying auth.0004\_alter\_user\_username\_opts... OK

Applying auth.0005\_alter\_user\_last\_login\_null... OK

Applying auth.0006\_require\_contenttypes\_0002... OK

Applying auth.0007\_alter\_validators\_add\_error\_messages... OK

Applying auth.0008\_alter\_user\_username\_max\_length... OK

Applying auth.0009\_alter\_user\_last\_name\_max\_length... OK

Applying auth.0010\_alter\_group\_name\_max\_length... OK

Applying auth.0011\_update\_proxy\_permissions... OK

Applying sessions.0001\_initial... OK



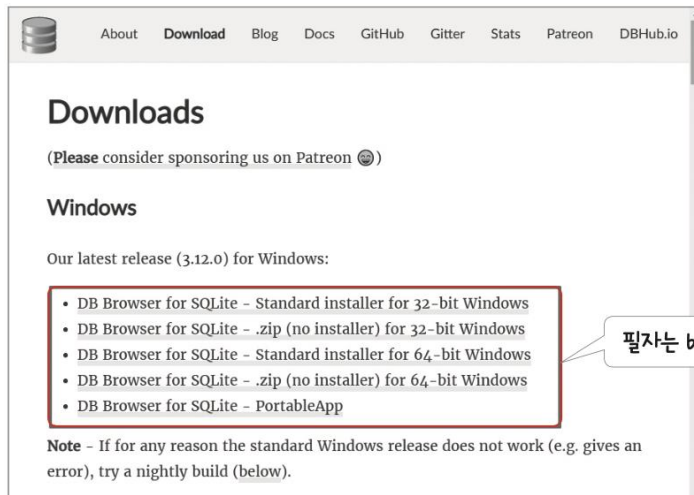
## 02-2 데이터를 관리하는 모델

Do it!  
실습

### migrate와 테이블 알아보기

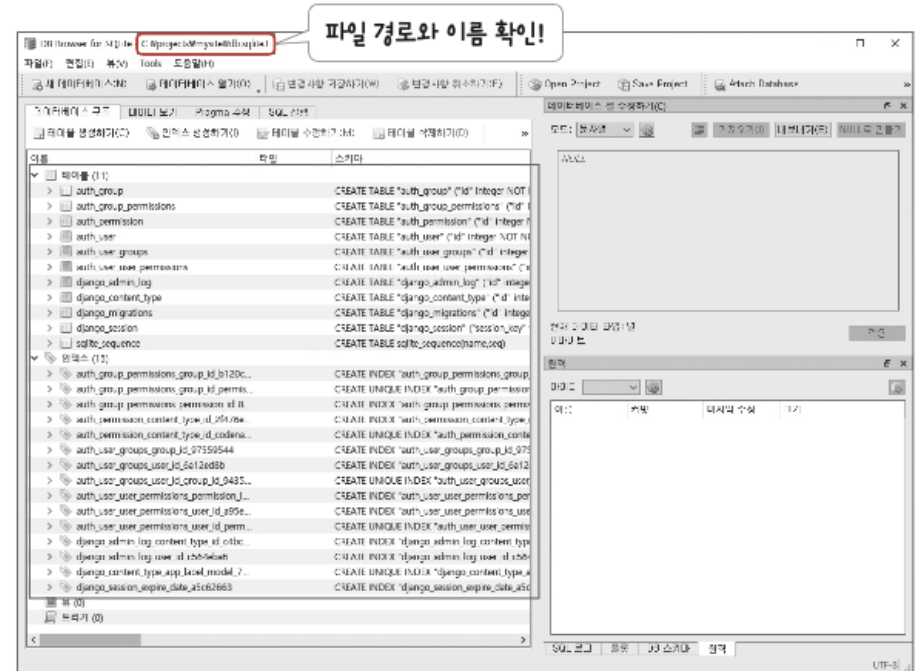
• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

#### 05단계 DB Browser for SQLite로 테이블 살펴보기



- ▶ 'DB Browser for SQLite'를 설치하면 데이터베이스의 테이블을 확인할 수 있다.

#### 06단계 DB Browser for SQLite 살펴보기



DB Browser for SQLite에서 mysite/db.sqlite3 파일을 열면 나타나는 화면

## 02-2 데이터를 관리하는 모델

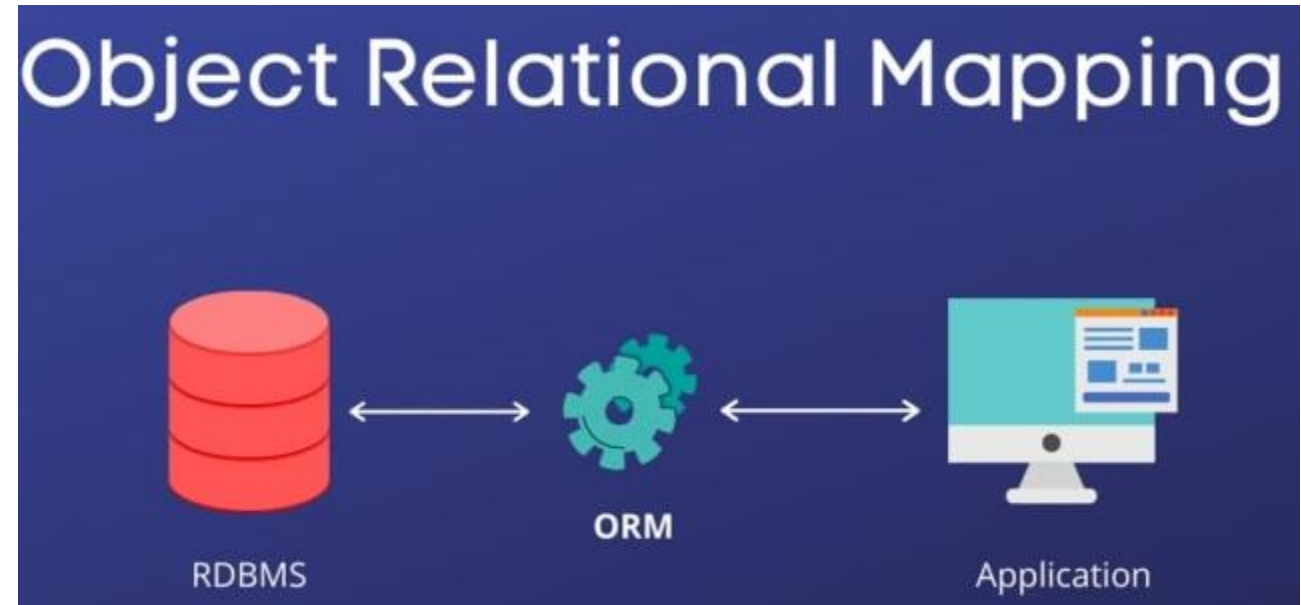
• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

SQLite? ORM?



- ▶ RDBMS의 한 종류로 설치하지 않고, 파일로 데이터베이스 데이터들을 관리(**SQL + Light**의미로 가벼운 SQL 기반 DB를 의미)



- ▶ '객체 - 관계 매핑을 통해 RDBMS와 같은 데이터베이스에 데이터를 저장할 때 SQL 쿼리문이 아닌 일반적인 프로그래밍 언어를 통해 저장 (내부적으로는 변환)

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 모델 만들기

#### 01단계 모델 속성 구상하기

##### ▶ 질문 모델에서 필요한 속성

속성명	설명
subject	질문의 제목
content	질문의 내용
create_date	질문을 작성한 일시

##### ▶ 답변 모델에서 필요한 속성

속성명	설명
question	질문(어떤 질문의 답변인지 알아야 하므로 질문 속성이 필요함)
content	답변의 내용
create_date	답변을 작성한 일시

#### 02단계 pybo/models.py에 질문 모델 작성하기

파일 이름 C:/projects/mysite/pybo/models.py

```
from django.db import models

class Question(models.Model):
    subject = models.CharField(max_length=200)
    content = models.TextField()
    create_date = models.DateTimeField()
```

- ▶ 질문 모델은 Question 클래스(모델)로 만든다.
- ▶ CharField : 글자 수 제한이 있는 데이터  
TextField : 글자 수 제한이 없는 데이터  
DateTimeField : 날짜, 시간 관련 속성

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 모델 만들기

#### 03단계 pybo/models.py에 답변 모델 작성하기

```
파일 이름 C:/projects/mysite/pybo/models.py

from django.db import models

class Question(models.Model):
    subject = models.CharField(max_length=200)
    content = models.TextField()
    create_date = models.DateTimeField()

class Answer(models.Model):
    question = models.ForeignKey(Question, on_delete=models.CASCADE)
    content = models.TextField()
    create_date = models.DateTimeField()
```

- ▶ 어떤 모델이 다른 모델을 속성으로 가지면 ForeignKey를 이용한다.

#### 04단계 config/settings.py를 열어 pybo 앱 등록하기

```
파일 이름 C:/projects/mysite/config/settings.py

(... 생략 ...)
INSTALLED_APPS = [
    'pybo.apps.PyboConfig',
    'django.contrib.admin',
    'django.contrib.auth',
    (... 생략 ...)
]
(... 생략 ...)
```

- ▶ 테이블 생성을 하려면 config/settings.py파일에서 INSTALLED\_APPS 항목에 pybo 앱을 추가해야 한다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 모델 만들기

#### 05단계 pybo/apps.py 열어 살펴보기

파일 이름 C:/projects/mysite/pybo/apps.py

```
from django.apps import AppConfig

class PyboConfig(AppConfig):
    name = 'pybo'
```

- ▶ PyboConfig 클래스가 config/settings.py 파일의 INSTALLED\_APPS 항목에 추가되지 않으면 장고는 pybo 앱을 인식하지 못하고 데이터베이스 관련 작업도 할 수 없다.

#### 06단계 migrate로 테이블 생성하기

C:\ 명령 프롬프트

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

Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  No migrations to apply.
  Your models have changes that are not yet reflected in a migration, and so won't be applied.
  Run 'manage.py makemigrations' to make new migrations, and then re-run 'manage.py migrate' to apply them.
```

manage.py makemigrations 명령을 실행 후  
manage.py migrate 명령을 실행하세요

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 모델 만들기

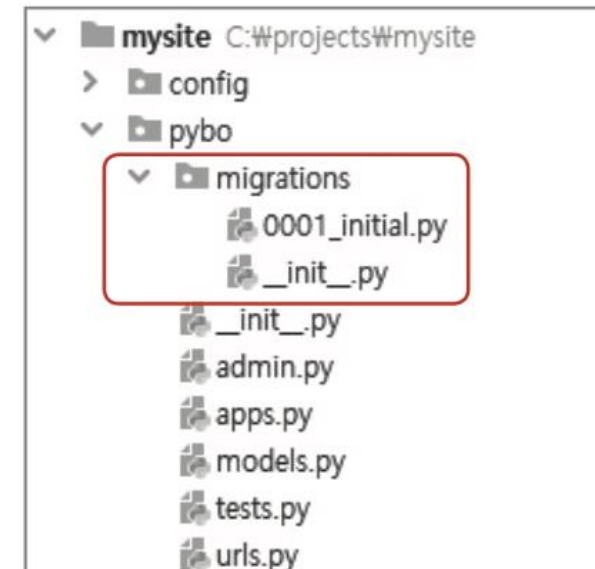
#### 07단계 makemigrations로 테이블 작업 파일 생성하기

```
C:\> 명령 프롬프트

(mysite) C:\projects\mysite>python manage.py makemigrations
Migrations for 'pybo':
  pybo\migrations\0001_initial.py
    - Create model Question
    - Create model Answer
```

- ▶ makemigrations 명령은 장고가 테이블 작업을 수행하기 위한 파일들을 생성한다.
- ▶ 실제 테이블 생성 명령은 migrate이다.

#### 08단계 makemigrations로 생성된 파일 위치 살펴보기



## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 모델 만들기

#### 09단계 makemigrations 한 번 더 실행해 보기

```
C:\> 명령 프롬프트
(mysite) C:\projects\mysite>python manage.py makemigrations
No changes detected
```

- ▶ 모델의 변경사항이 없다면 '모델 변경 사항 없음'이라고 알려 주는 것이다.

#### 10단계 migrate 실행하기

```
C:\> 명령 프롬프트
(mysite) C:\projects\mysite>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, pybo, sessions
Running migrations:
  Applying pybo.0001_initial... OK
```

- ▶ migrate 명령을 실행하면 장고는 등록된 앱에 있는 모델을 참조하여 실제 테이블을 생성한다.

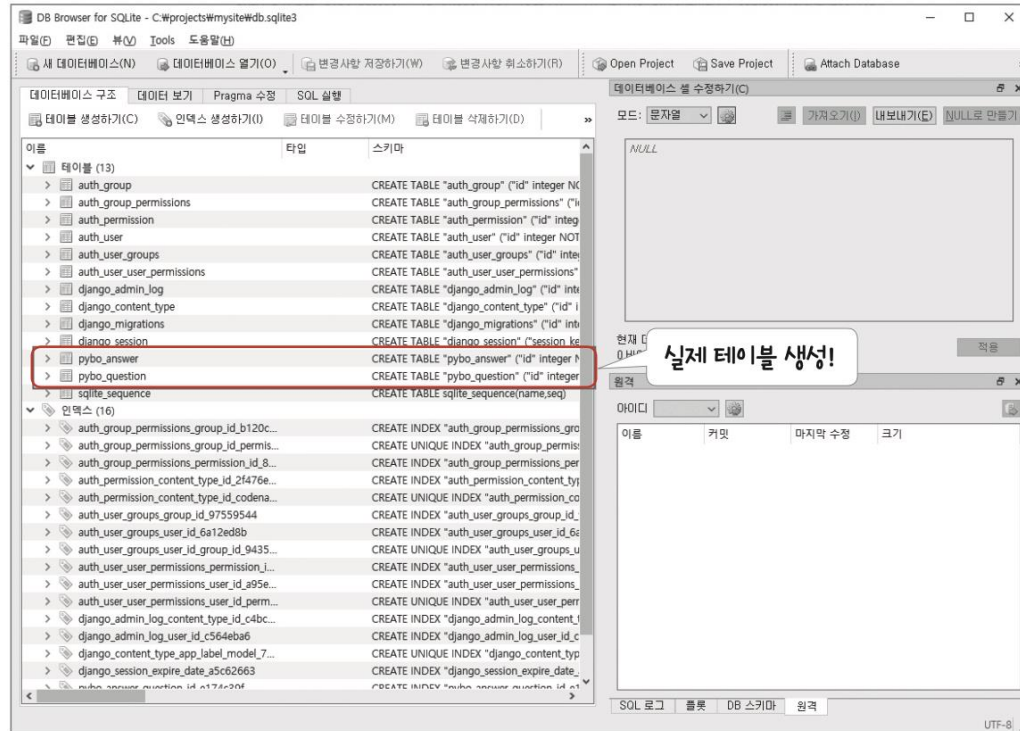
## 02-2 데이터베이스를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 모델 만들기

#### 11단계 DB Browser for SQLite로 생성된 테이블 확인하기



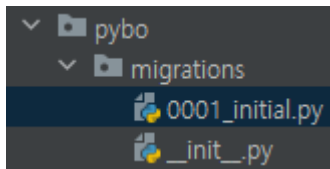
- ▶ 실제 테이블명을 보면 Question 모델은 pybo\_question으로, Answer 모델은 pybo\_answer로 테이블 이름이 지어졌음을 확인할 수 있다.



## 02-2 데이터를 관리하는 모델

Do it!  
실습

모델 확인



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

```
C:\> 명령 프롬프트  
  
(mysite) C:\projects\mysite>python manage.py sqlmigrate pybo 0001  
BEGIN;  
--  
-- Create model Question  
(mysite) C:\projects\mysite>python manage.py sqlmigrate pybo 0001  
BEGIN;  
--  
-- Create model Question  
--  
CREATE TABLE "pybo_question" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,  
"subject" varchar(200) NOT NULL, "content" text NOT NULL, "create_date" datetime  
NOT NULL);  
--  
-- Create model Answer  
--  
CREATE TABLE "pybo_answer" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "con-  
tent" text NOT NULL, "create_date" datetime NOT NULL, "question_id" integer NOT NULL  
REFERENCES "pybo_question" ("id") DEFERRABLE INITIALLY DEFERRED);  
CREATE INDEX "pybo_answer_question_id_e174c39f" ON "pybo_answer" ("question_id");  
COMMIT;
```

- ▶ 'pybo'는 makemigrations 명령을 실행할 때 생성된 pybo/migrations/0001\_initial.py의 마이그레이션명을 의미
- ▶ '0001'은 생성된 파일의 일련번호

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 데이터 만들고 저장하고 조회하기

#### 01단계 장고 셸 실행하기

```
C:\> 명령 프롬프트

(mysite) C:\projects\mysite>python manage.py shell
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
```

- ▶ 장고 셸은 장고에 필요한 환경들이 자동으로 설정되어 실행되므로 파이썬 셸과는 약간의 차이가 있다.

#### 02단계 Question, Answer 모델 임포트하기

```
C:\> 명령 프롬프트

>>> from pybo.models import Question, Answer
```

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

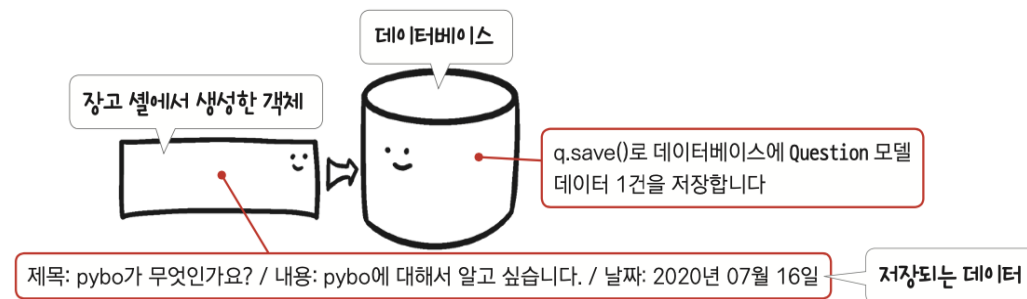
### 데이터 만들고 저장하고 조회하기

#### 03단계 Question 모델로 Question 모델 데이터 만들기

C:\ 명령 프롬프트

```
>>> from django.utils import timezone
>>> q = Question(subject='pybo가 무엇인가요?', content='pybo에 대해서 알고 싶습니다.',
create_date=timezone.now())
>>> q.save()
```

- ▶ create\_date 속성은 DateTimeField이므로 timezone.now()로 현재 일시를 입력한다.
- ▶ 객체 q가 생성된 다음 q.save()를 입력하면 Question 모델 데이터 1건이 데이터베이스에 저장된다.



## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

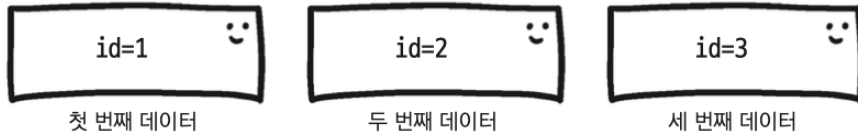
### 데이터 만들고 저장하고 조회하기

#### 04단계 Question 모델 데이터의 id값 확인하기

C:\ 명령 프롬프트

```
>>> q.id  
1
```

- ▶ 장고는 데이터 생성 시 데이터에 id값을 자동으로 넣어준다.



모델 데이터에 자동으로 부여된 id값

#### 05단계 Question 모델로 Question 모델 데이터 1개 더 만들기

C:\ 명령 프롬프트

```
>>> q = Question(subject='장고 모델 질문입니다.', content='id는 자동으로 생성되나요?', create_
date=timezone.now())  
>>> q.save()  
>>> q.id  
2
```

- ▶ 두 번째로 생성한 Question 모델 데이터의 id는 2이다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 데이터 만들고 저장하고 조회하기

#### 06단계 Question 모델 데이터 모두 조회하기

```
C:\ 명령 프롬프트
>>> Question.objects.all()
<QuerySet [<Question: Question object (1)>, <Question: Question object (2)>]>
```

- ▶ `Question.objects.all()`은 `Question`에 저장된 모든 데이터를 조회하는 함수이다.
- ▶ `<Question object (1)>`, `<Question object (2)>`의 1, 2가 바로 장고에서 `Question` 모델 데이터에 자동으로 입력해 준 `id`이다.

#### 07단계 Question 모델 데이터 조회 결과에 속성값 보여 주기

```
파일 이름 C:/projects/mysite/pybo/models.py

(... 생략 ...)

class Question(models.Model):
    subject = models.CharField(max_length=200)
    content = models.TextField()
    create_date = models.DateTimeField()

    def __str__(self):
        return self.subject

(... 생략 ...)
```

- ▶ `Question` 모델에 `__str__` 메서드를 추가하면 데이터 조회 시 `id`가 아닌 제목을 표시해 준다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 데이터 만들고 저장하고 조회하기

#### 08단계 Question 모델 데이터 다시 조회해 보기

```
C:\> 명령 프롬프트

(mysite) C:\projects\mysite>python manage.py shell
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from pybo.models import Question, Answer
>>> Question.objects.all()
<QuerySet [<Question: pybo가 무엇인가요?>, <Question: 장고 모델 질문입니다.>]>
```

제목이 표시됨!

- ▶ makemigrations, migrate 명령은 모델의 속성이 추가되거나 변경된 경우에 실행해야 하는 명령이다.

#### 09단계 조건으로 Question 모델 데이터 조회하기

```
C:\> 명령 프롬프트

>>> Question.objects.filter(id=1)
<QuerySet [<Question: pybo가 무엇인가요?>]>
```

- ▶ filter 함수는 조건에 해당하는 데이터를 모두 찾아준다.
- ▶ filter 함수는 반환값이 리스트 형태인 QuerySet이다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 데이터 만들고 저장하고 조회하기

#### 10단계 Question 모델 데이터 하나만 조회하기

```
C:\ 명령 프롬프트
>>> Question.objects.get(id=1)
<Question: pybo가 무엇인가요?>
```

- ▶ get 함수를 사용하면 리스트가 아닌 데이터 하나만 조회할 수 있다.
- ▶ filter 함수는 여러 건의 데이터를 반환하지만, get 함수는 단 한 건의 데이터를 반환한다.

#### 11단계 get으로 조건에 맞지 않는 데이터 조회하기

```
C:\ 명령 프롬프트
>>> Question.objects.get(id=3)
Traceback (most recent call last):
  File "<console>", line 1, in <module>
  File "C:\venvs\mysite\lib\site-packages\django\db\models\manager.py", line 82, in manager_method
    return getattr(self.get_queryset(), name)(*args, **kwargs)
  File "C:\venvs\mysite\lib\site-packages\django\db\models\query.py", line 415, in get
    raise self.model.DoesNotExist(
pybo.models.Question.DoesNotExist: Question matching query does not exist.
```

- ▶ 조건에 맞는 데이터 1개를 반환해야 하는데 조건에 맞는 데이터가 없으니 오류가 발생한 것이다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 데이터 만들고 저장하고 조회하기

#### 12단계 filter로 조건에 맞지 않는 데이터 조회하기

```
C:\ 명령 프롬프트
>>> Question.objects.filter(id=3)
<QuerySet []>
```

- ▶ 조건에 맞는 데이터가 없으면 빈 QuerySet을 반환한다.

#### 13단계 제목의 일부를 이용하여 데이터 조회하기

```
C:\ 명령 프롬프트
>>> Question.objects.filter(subject__contains='장고')
<QuerySet [<Question: 장고 모델 질문입니다.>]>
```

- ▶ `subject__contains='장고'`의 의미는 'subject 속성에 '장고'라는 문자열이 포함되어 있는가?'이다.



## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 데이터 수정하기

#### 01단계 Question 모델 데이터 수정하기

```
C:\> 명령 프롬프트

>>> q = Question.objects.get(id=2)
>>> q
<Question: 장고 모델 질문입니다.>
```

#### 02단계 subject 속성 수정하기

```
C:\> 명령 프롬프트

>>> q.subject = 'Django Model Question'
```

#### 03단계 수정한 Question 모델 데이터 데이터베이스에 저장하기

```
C:\> 명령 프롬프트

>>> q.save()
>>> q
<Question: Django Model Question>
```

- ▶ 반드시 save 함수를 실행해야 변경된 Question 모델 데이터가 데이터베이스에 반영된다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 데이터 삭제하기

#### 01단계 Question 모델 데이터 삭제하기

```
C:\> 명령 프롬프트

>>> q = Question.objects.get(id=1)
>>> q.delete()
(1, {'pybo.Question': 1})
```

- ▶ delete 함수를 수행하면 해당 데이터가 데이터베이스에서 즉시 삭제되며, 삭제된 데이터의 추가 정보가 반환된다.

(1, {'pybo.Question': 1}) → 앞에 1 : id // 뒤에 1 : 삭제된 개수


- ▶ 눈으로만 확인하세요.

```
>>> a = Answer(question=q, content='네 자동으로 생성됩니다 22222222222', create_date=timezone.now())
>>> a
<Answer: Answer object (None)>
>>> a.save()
>>> a.id
2
>>> q.delete()
(3, {'pybo.Answer': 2, 'pybo.Question': 1})
```

#### 02단계 삭제 확인하기

```
C:\> 명령 프롬프트

>>> Question.objects.all()
<QuerySet [<Question: Django Model Question>]>
```

 Answer 모델을 만들 때 ForeignKey로 Question 모델과 연결한 것이 기억나는가? 만약 삭제한 Question 모델 데이터에 2개의 Answer 모델 데이터가 등록된 상태라면 (1, {'pybo.Answer': 2, 'pybo.Question': 1})와 같이 삭제된 답변 개수도 함께 반환될 것이다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 연결된 데이터 알아보기

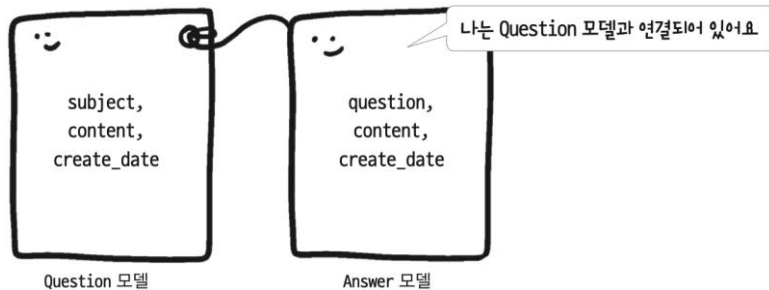
#### 01단계 Answer 모델 데이터 만들기

```
C:\ 명령 프롬프트
>>> q = Question.objects.get(id=2)
>>> q
<Question: Django Model Question>
>>> from django.utils import timezone
>>> a = Answer(question=q, content='네 자동으로 생성됩니다.', create_date=timezone.now())
>>> a.save()
```

#### 02단계 id 확인하기

```
C:\ 명령 프롬프트
>>> a.id
1
```

▶ Answer 모델 데이터에도 id가 있다.



## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

### 연결된 데이터 알아보기

#### 03단계 Answer 모델 데이터 조회하기

```
C:\> 명령 프롬프트

>>> a = Answer.objects.get(id=1)
>>> a
<Answer: Answer object (1)>
```

- ▶ get 함수로 조회할 때 조건은 id를 사용한다.

#### 04단계 연결된 데이터로 조회하기: 답변에 있는 질문 조회하기

```
C:\> 명령 프롬프트

>>> a.question
<Question: Django Model Question>
```

- ▶ Answer 모델 데이터에는 Question 모델 데이터가 연결되어 있으므로 Answer 모델 데이터에 연결된 Question 모델 데이터를 조회할 수 있다.

## 02-2 데이터를 관리하는 모델

• 완성 소스 [github.com/pahkey/djangobook/tree/2-02](https://github.com/pahkey/djangobook/tree/2-02)

Do it!  
실습

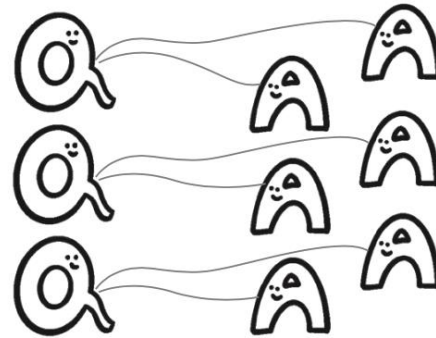
### 연결된 데이터 알아보기

#### 05단계 연결된 데이터로 조회하기: 질문을 통해 답변 찾기

C:\ 명령 프롬프트

```
>>> q.answer_set.all()  
<QuerySet [Answer: Answer object (1)]>
```

- ▶ Question 모델과 Answer 모델처럼 서로 연결되어 있으면 연결모델명\_set과 같은 방법으로 연결된 데이터를 조회할 수 있다.
- ▶ 질문 1개에는 1개 이상의 답변이 달릴 수 있으므로 질문에 달린 답변은 q.answer\_set으로 조회해야 한다. (답변 세트를 조회)



나는 Question 모델 데이터에 연결되어 있어!

Question 모델 데이터에 연결된 Answer 모델 데이터의 모습

**Thank You**