

# 챕터1

## Conda 가상환경 설정하기

Anaconda를 환경변수에 설정한 상태에서,

Conda create --name askcompany python=3.7

으로 가상환경을 생성한다.

사용할 때는 conda activate askcompany 로 활성화

Where python 명령어로 현제 파이썬이 어떤 경로의 파이썬인지 알 수 있다. Where pip도

설치되어 있는 conda 가상환경 리스트 확인하는 단축키

Conda env list

Pip install “django~=3.0.0”으로 장고 설치

Django-admin –version 3.0.4

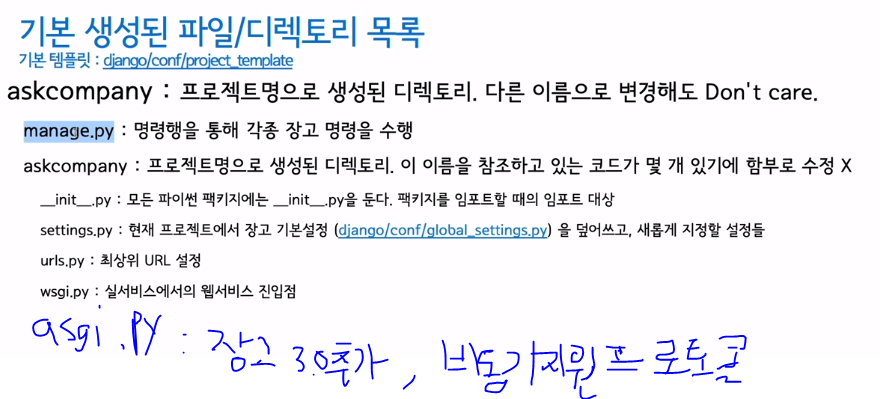
Python 3.7.6

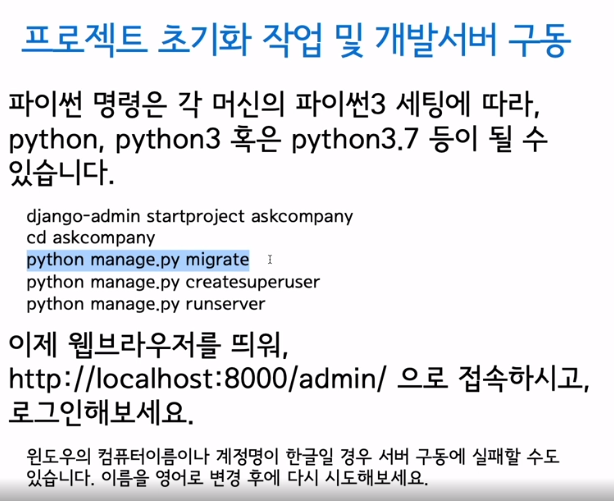
Pip verision 20.0.2

비쥬얼 스튜디오 코드에서 ‘보기/명령 팔레트 설정/select interpreter’을 선택하면 아까 생성한 가상환경을 사용할 수 있다.

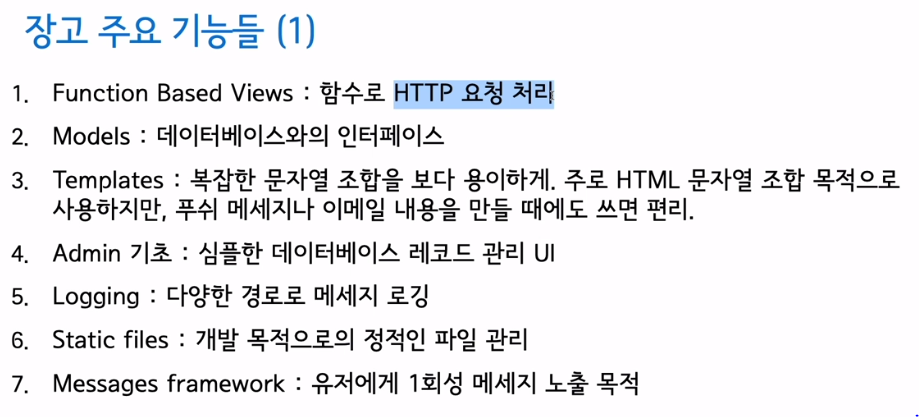
‘보기/명령 팔레트 설정/select default shell’ 클릭해서 어떤 것을 디폴트 쉘로 쓸 것인지 지정 가능하다.

## 03 장고 프로젝트 생성

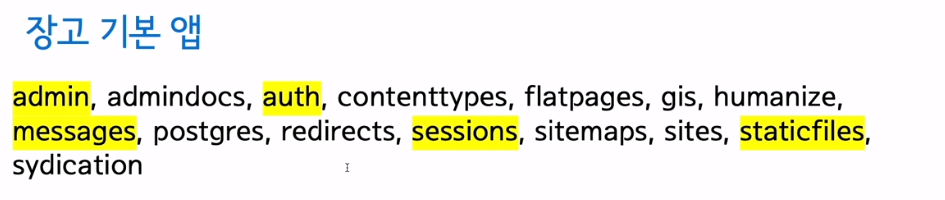


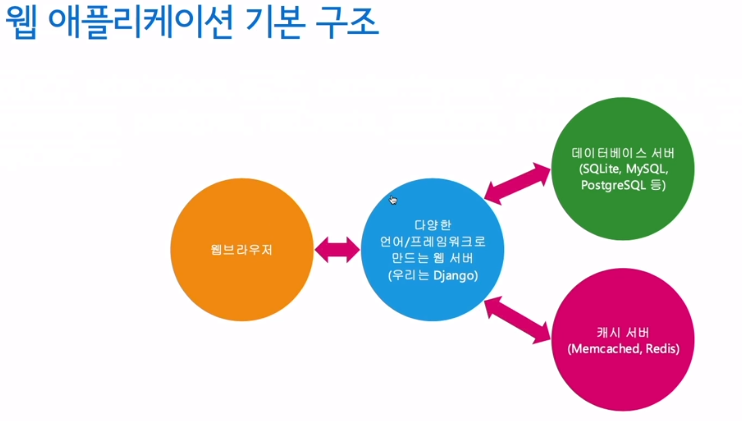


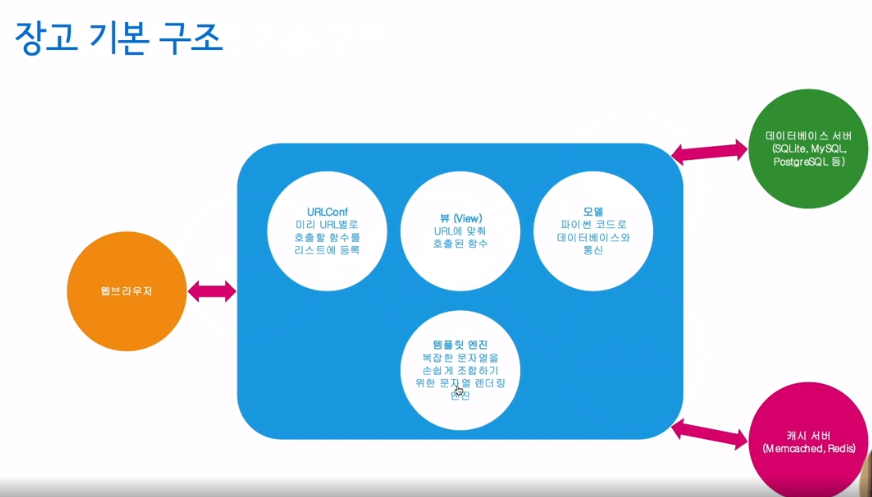
## 04 장고 주요 구성 요소



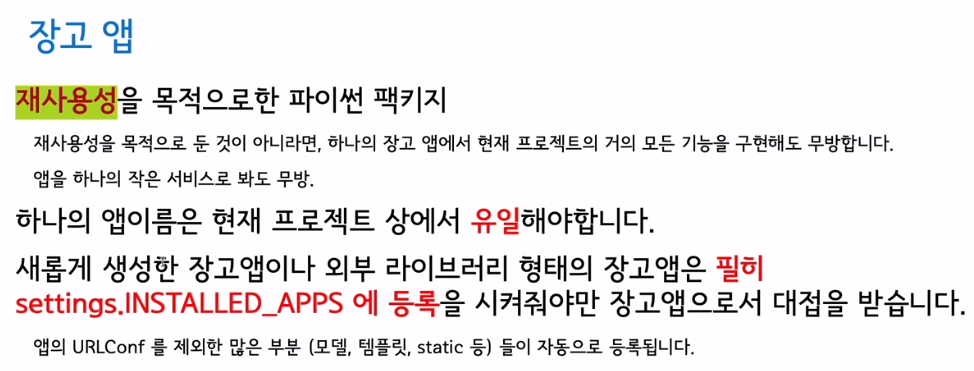






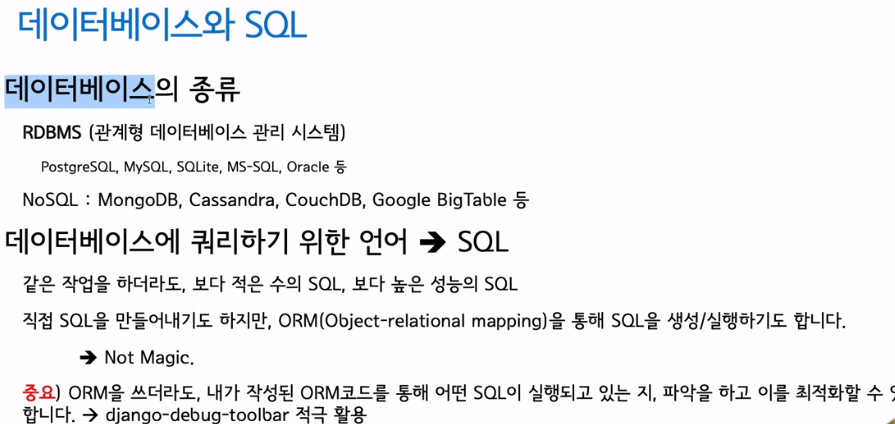


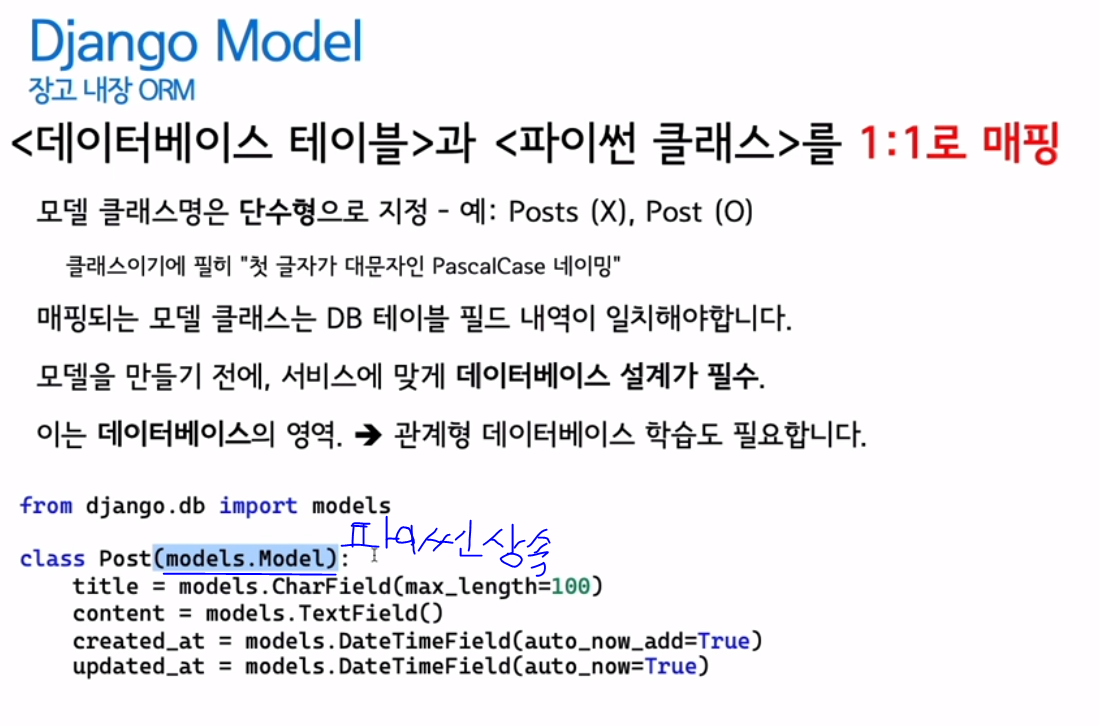
## 05 장고앱과 블로그 코딩쇼



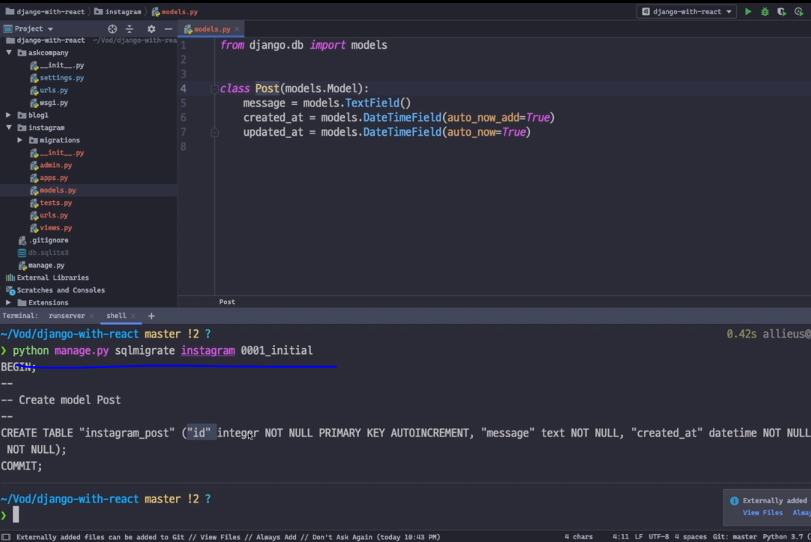
# 챕터2

## 1장고 모델 (ORM) 소개





클래스안에 필드명과 필드 타입 정의



실제로 데이터베이스에 들어간 쿼리를 볼 수 있다.

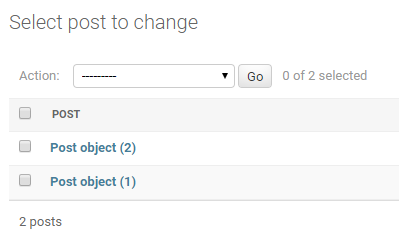
Python manage.py sqlmigrate instagram 0001\_initial

## 2 장고 모델 필드

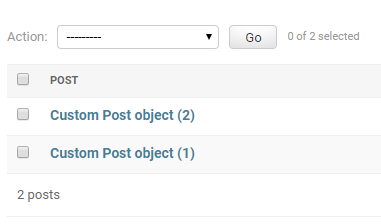
## 3 장고 admin을 통한 데이터 관리

어떤 객체에 대한 문자열 표현이 필요할 때 아래처럼 \_\_str\_\_ 함수로 구현 할 수 있다.

(자바의 toString()과 유사)



위는 기본 문자열 표현



위는 \_\_str\_\_함수로 재정의한 오브젝트의 문자열 표현

# Create your models here.

class Post(models.Model):

    message = models.TextField()

    created\_at = models.DateTimeField(auto\_now\_add=True)

    updated\_at = models.DateTimeField(auto\_now=True)

    # Java의 toString

    def \_\_str\_\_(self):

        # 파이썬 3.7부터 지원되는 문법

        #return f"Custom Post object ({self.id})"

        # 파이썬 3.7 이전에서 사용하려면 아래와 같이

        return "Custom Post object ({})".format(self.id)

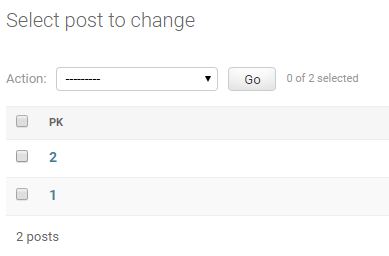
@admin.register(Post)

class PostAdmin(admin.ModelAdmin):

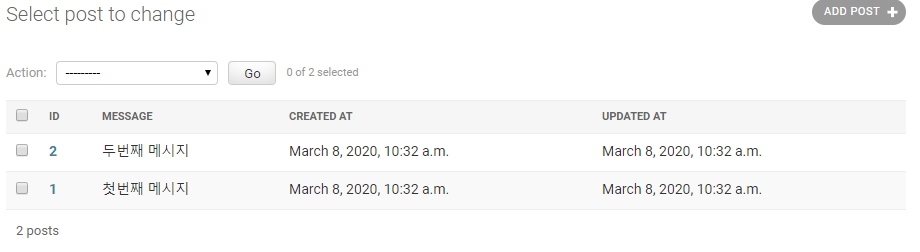
    list\_display = ['pk']

admin페이지에서 해당 테이블에서 보여줄 field 목록을 list\_display에 구성하여 보여줄 수 있다.

primarykey의 별칭으로 ‘pk’를 사용할 수 있다. (default로 id 사용하고 있는 상태에서)



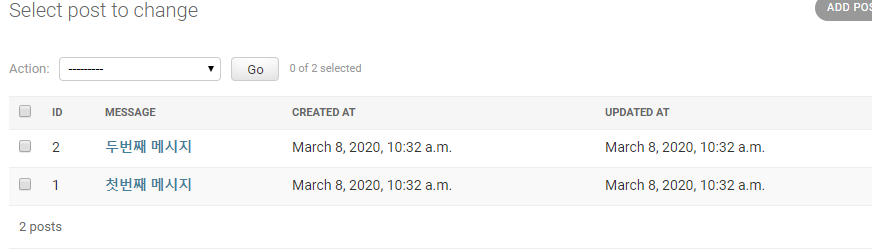
그러면 위처럼 나온다.



@admin.register(Post)

class PostAdmin(admin.ModelAdmin):

    list\_display = ['id', 'message', 'created\_at', 'updated\_at']



@admin.register(Post)

class PostAdmin(admin.ModelAdmin):

    list\_display = ['id', 'message', 'created\_at', 'updated\_at']

    list\_display\_links = ['message']

list\_display\_links 리스트로 링크를 연결할 field값을 지정할 수 있다. (여러 개 입력 가능)

# Create your models here.

class Post(models.Model):

    message = models.TextField()

    created\_at = models.DateTimeField(auto\_now\_add=True)

    updated\_at = models.DateTimeField(auto\_now=True)

    # Java의 toString

    def \_\_str\_\_(self):

        # 파이썬 3.7부터 지원되는 문법

        #return f"Custom Post object ({self.id})"

        # 파이썬 3.7 이전에서 사용하려면 아래와 같이

        #return "Custom Post object ({})".format(self.id)

        return self.message

    def message\_length(self):

        return len(self.message)

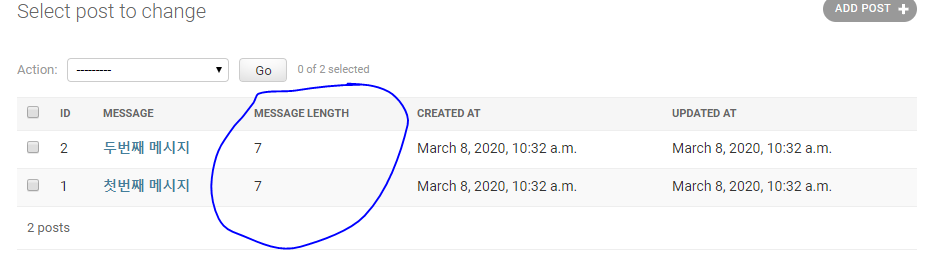
@admin.register(Post)

class PostAdmin(admin.ModelAdmin):

    list\_display = ['id', 'message', 'message\_length', 'created\_at', 'updated\_at']

    list\_display\_links = ['message']

인자가 없는 메서드를 모델에 생성하고, list\_display 리스트 안에 넣어서 어드민 페이지에서 보여줄 수 있다.



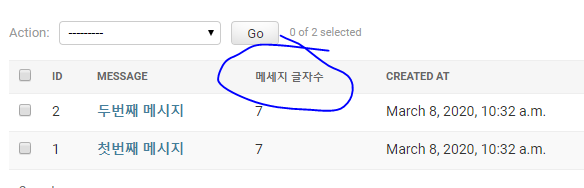
    def message\_length(self):

        return len(self.message)

    message\_length.short\_description = "메세지 글자수"

위 3번째 라인은 함수에 설명을 넣는 메서드인 것 같다.

아래처럼 admin 페이지에서 field 명을 바꿀 수 있다.



@admin.register(Post)

class PostAdmin(admin.ModelAdmin):

    list\_display = ['id', 'message', 'message\_length', 'created\_at', 'updated\_at']

    list\_display\_links = ['message']

    def message\_length(self, post):

        return len(post.message)

자주 쓰는 메서드라면 models.py에 정의해서 사용해야 하겠지만, 단발성으로 사용하는 경우라면 admin에 위 처럼 메서드를 정의하여 사용할 수 있다.

정의된 Message\_length(self, post) 함수는 admin이 알아서 호출해주고, 이 함수는 post 객체가 넘어온다.

@admin.register(Post)

class PostAdmin(admin.ModelAdmin):

    list\_display = ['id', 'message', 'message\_length', 'created\_at', 'updated\_at']

    list\_display\_links = ['message']

    def message\_length(self, kkkk):

        #return len(kkkk.message)

        return f"{len(kkkk.message)} 글자"

이런식으로 2번째 인자 이름을 막 넣어도 되는 것을 보니, admin에 정의된 메서드 2번째 인자 위치에 model의 인스턴스 주소가 넘어오는듯 하다. (추정)

### Search\_fields 속성 정의 : admin내 검색UI를 통해, DB를 통한 where 쿼리 대상 필드 리스트

python manage.py

shell

Python 3.7.6 (default, Jan 8 2020, 20:23:39) [MSC v.1916 64 bit (AMD64)] on win32

Type "help", "copyright", "credits" or "license" for more information.

(InteractiveConsole)

>>> from instagram.models import Post

>>> Post.objects.all()

<QuerySet [<Post: 첫번째 메시지>, <Post: 두번째 메시지>]>

>>> post.objects.all().filter(message\_\_icontains='첫번째')

Traceback (most recent call last):

File "<console>", line 1, in <module>

NameError: name 'post' is not defined

>>> Post.objects.all().filter(message\_\_icontains='첫번째')

<QuerySet [<Post: 첫번째 메시지>]>

* Filter랑 필드명\_\_icontains=’찾을문자열’ 입력하여 해당 문자열이 포함된 단어만 추출할 수 있다.

>>> qs = Post.objects.all().filter(message\_\_icontains='첫번째')

>>> print(qs)

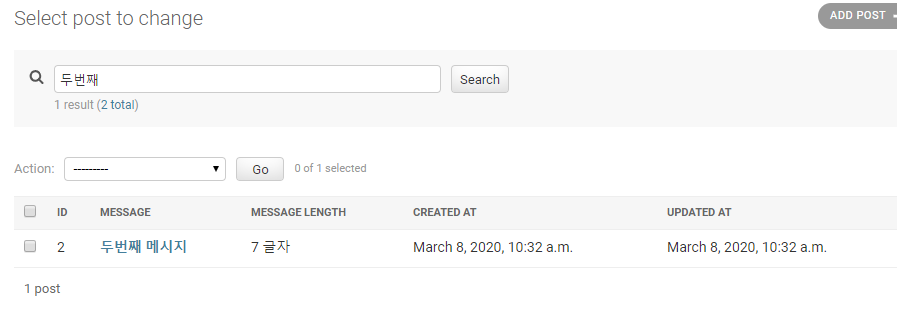
<QuerySet [<Post: 첫번째 메시지>]>

>>> print(qs.query)

SELECT "instagram\_post"."id", "instagram\_post"."message", "instagram\_post"."created\_at", "instagram\_post"."updated\_at" FROM "instagram\_post" WHERE "instagram\_post"."message" LIKE %첫번째% ESCAPE '\'

처럼 .query 를 붙여서 db로 전달되는 쿼리문을 볼 수 있다.

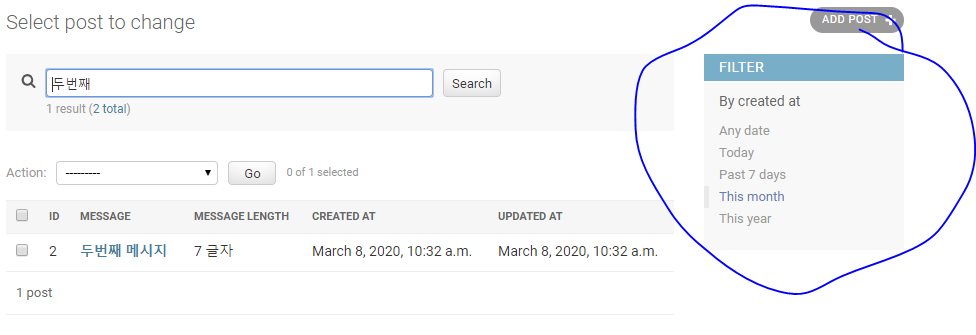
    search\_fields = ['message']



위처럼 search\_fields를 만들어서 쉽게 검색기능을 만들 수 있다.

### List\_filter 속성 정의 : 지정 필드값으로 필터링 옵션 제공

list\_filter = ['created\_at']



is\_public = models.BooleanField(default=False, verbose\_name='공개여부')

list\_display = ['id', 'message', 'message\_length', 'is\_public', 'created\_at', 'updated\_at']

    list\_filter = ['message', 'is\_public']

옵션을 넣어서 공개 여부를 지정하고, 필터링을 걸 수 있다.

## 4장고가 media 파일을 다루는 방법

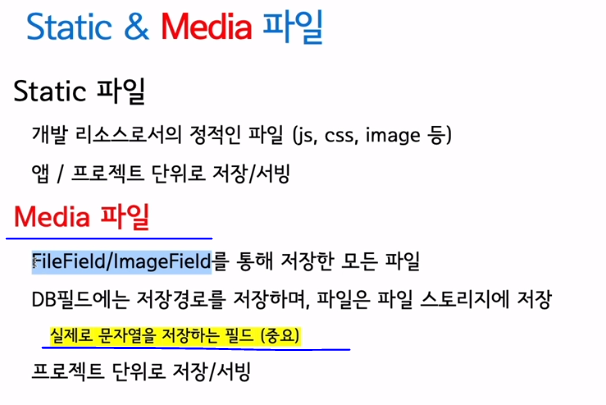


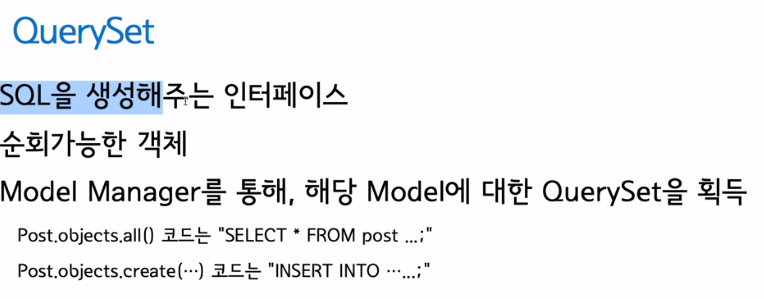
photo = models.ImageField(blank=True)

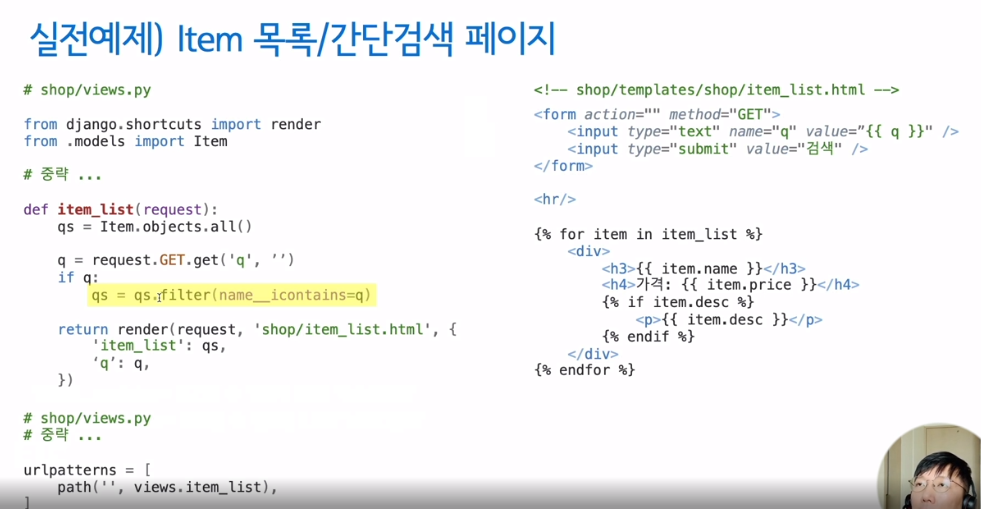
models.ImageField를 사용하기 위하여 아래 라이브러리 설치 필요 에러가 나왔었다. (파이썬 이미징 라이브러리)

python -m pip install Pillow

!@#$ : 지금 당장 필요한부분은 아니니 다른 영상부터 시청 후 컴백 예정 ( ~ 3:28 시청)

## 6 모델을 통한 조회(기초)





*# GET request의 인자중에 q 값이 있으면 가져오고, 없으면 빈 문자열 넣기*

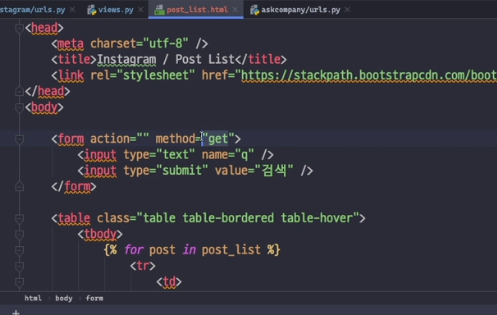
*# 제목에 q가 포함되어 있는 레코드만 필터링*

Q의 의미 아래 블로그에서 찾았음. url에서 query가 존재하면 이라는 의미같음

<https://wayhome25.github.io/django/2017/05/04/django-queryset-search/>

## 7 Queryset을 통한 간단 검색 구현

### form을 통하여 값 넣어서 특정 url에 request 날리는 방법



입력된 것을 get방식으로 전달

Form 의 Action=”” 이 비어있다는 것은 현제 Form이 있는 같은 주소를 의미한다는 뜻이다.

## 8. queryset의 정렬 및 범위 조건

Pip install Django-extensions

그리고 Django-extensions을 사용하려면 settings.py에서 INSTALLED\_APPS = []에 추가해야 한다.