# API 測試 新增自定義查詢集

原本只有繼承自models.Manager這個預設的管理器類別 objects.get()、objects.filter()等等
如果沒有指定objects的話，Django預設會使用models.Manager來做為該模型的管理器
(就是沒有使用自定義的objects)

如果平時有幾個很常用的查詢條件
那可以建立那些查詢條件的查詢器，這樣就不用每次查詢都要輸入條件式

假設 很常查詢student資料中，誰是男生
那麼條件式就會是student.objects.filter( `cSex__contains = "M"` )

假設 很常查詢student資料中，誰住台北
那麼條件式就會是student.objects.filter( `cAddr__contains = '台北'` )

每次都要輸入一堆，很麻煩

==========

那來定義一個專門用來查詢性別(cSex)、查詢住址(cAddr)的查詢指令

C:\Users\226083\django_test\myweb\myapp\models.py

新增在 class student(models.Model): 的前面
第8~23行

```
# 自定義查詢 search_cSex
# get_queryset 是繼承models.Manager內的指令，不能亂改
# super()呼叫父類別search_cSex
# 後面再銜接get_queryset.filter(條件)
class search_cSex(models.Manager):
    def get_queryset(self):
        return super(search_cSex,self).get_queryset().filter(cSex__contains = "M")
```

```
# 自定義查詢 search_cAddr
# get_queryset 是繼承models.Manager內的指令，不能亂改
# super()呼叫父類別search_cAddr
# 後面再銜接get_queryset.filter(條件)
class search_cAddr(models.Manager):
    def get_queryset(self):
        return super(search_cAddr,self).get_queryset().filter(cAddr__contains = '台北')
```

原始查詢集是我們利用全查詢得到的查詢集 “objects.all()”

不過只要透過get_queryset方法就能改變原始查詢集，使用多個不同管理器時，這樣才不會跟原始查詢集的指令衝突

第41~43行

```
# Create your models here.
class student(models.Model):
    SEX_CHOICES = [
        ('M', '男'),
        ('F', '女'),
    ]
    cName = models.CharField('姓名',max_length=20, null=False)
    cSex = models.CharField('性別',max_length=1, choices=SEX_CHOICES, default='', null=Fals
e)
    cBirthday = models.DateField('生日',null=False)
    cEmail = models.EmailField('Email',max_length=100, blank=True, default='')
    cPhone = models.CharField('手機',max_length=50, blank=True, default='')
    cAddr = models.CharField('地址',max_length=255, blank=True, default='')
    last_modified = models.DateTimeField('最后修改日期', auto_now = True)
    created = models.DateTimeField('保存日期',default = timezone.now)
    objects = models.Manager()
    obj_cSex = search_cSex()
    obj_cAddr = search_cAddr()
```

注意：

如果有用到自定義管理器，原始的objects一定要賦值，不然會導致該模型不存在objects管理器 (objects = models.Manager())

==========

進入 Python 的互動模式進行測試

(django) C:\Users\226083\django_test\myweb>py manage.py shell

```
(django) C:\Users\226083\django_test\myweb>py manage.py shell
Python 3.7.0 (default, Jun 28 2018, 08:04:48) [MSC v.1912 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)

>>> from myapp.models import student
>>> from django.utils import timezone
>>> student.objects.all()
<QuerySet [<student: Kevin>, <student: evan>, <student: lu>, <student: linc>, <student: ee>]>
>>>
```

先看看有哪些資料

```
>>> student.objects.all()
<QuerySet [<student: Kevin>, <student: evan>, <student: laby>, <student: lu>, <student: winnie>, <student: hata>]>
```

動作: -------- 去 6 中 0 個被選

| ID 1 | 姓名 2 | 性別 | 生日 | EMAIL | 手機 | 地址 |
|---|---|---|---|---|---|---|
| 1 | Kevin | 男 | 1995年5月23日 | kevin@student.ncut.edu.tw | 0900-123456 | 台中 |
| 2 | evan | 男 | 1996年3月23日 | evan@student.ncut.edu.tw | 0901-123456 | 台北 |
| 3 | laby | 女 | 2000年7月29日 | laby@student.ncut.edu.tw | 0902-123456 | 台北 |
| 4 | lu | 男 | 1995年10月13日 | lu@student.ncut.edu.tw | 0903-123456 | 台北 |
| 5 | winnie | 女 | 1996年1月29日 | winnie@student.ncut.edu.tw | 0904-123456 | 台中 |
| 6 | hata | 男 | 2001年5月15日 | hata@student.ncut.edu.tw | 0906-123456 | 台中 |

6 students

發現有一筆資料有錯，改

```
# 先找出內部資料有問題的那筆資料
>>> s6 = student.objects.get(cName="hata")

# 確認有誤的資料
>>> s6.cSex
'M'

# 改成正確的
>>> s6.cSex = "F"

# 記得存檔
>>> s6.save()

# 確認改變
>>> s6.cSex
'F'
```

==========

輸入

```
student.obj_cSex.all()
```

```
>>> student.obj_cSex.all()
<QuerySet [<student: Kevin>, <student: evan>, <student: lu>]>
```

顯示的資料，內部cSex資料都是”F”(男性)

==========

輸入

```
student.obj_cAddr.all()
```

```
>>> student.obj_cAddr.all()
<QuerySet [<student: evan>, <student: laby>, <student: lu>]>
```

顯示的資料，內部cAddr資料都是”台北”