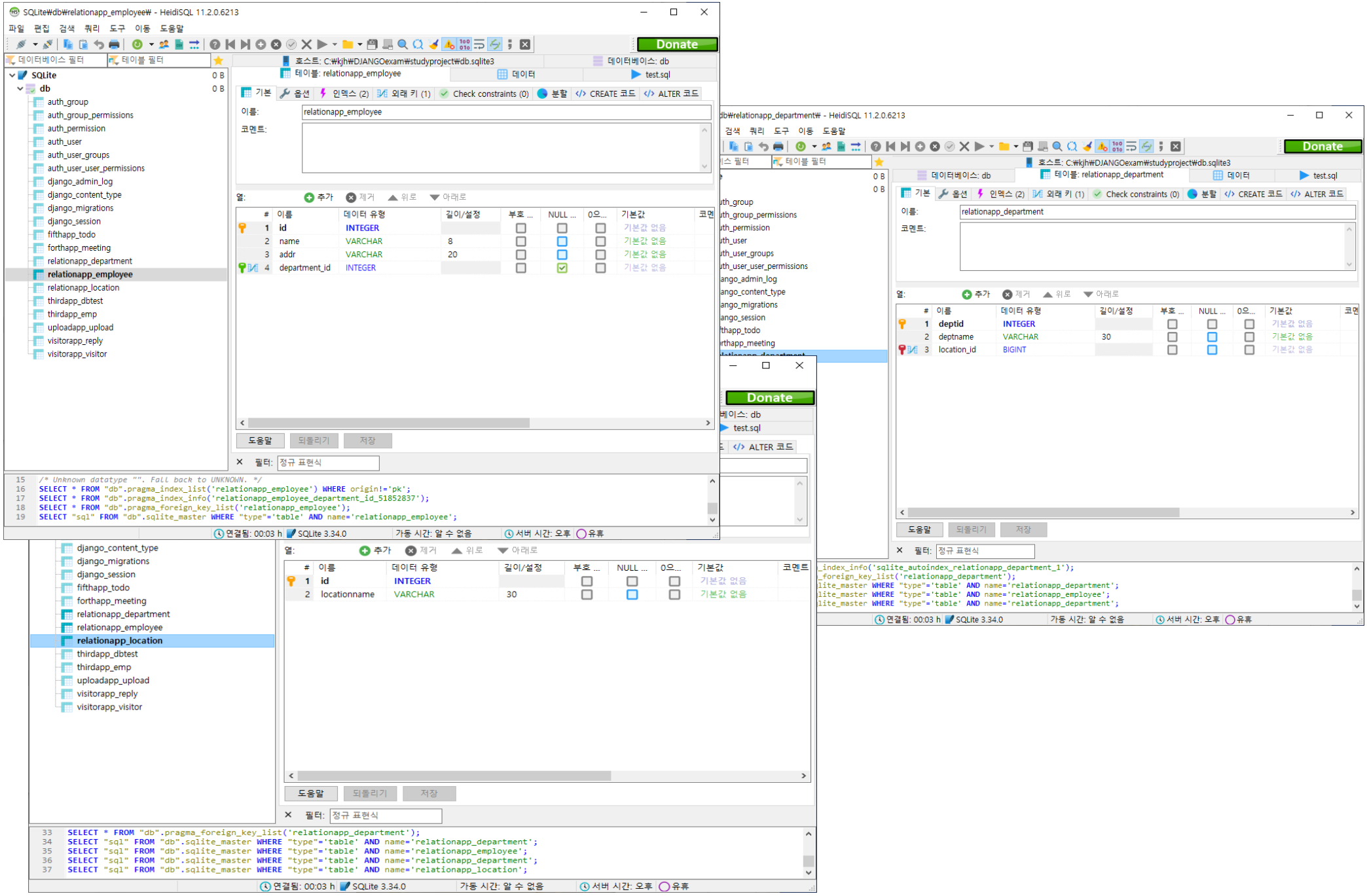
[모델 릴레이션(DB 테이블 릴레이션) 테스트]

(1) 제공된 relationapp을 studyproject에 붙여넣고 settings.py 에 등록한다.

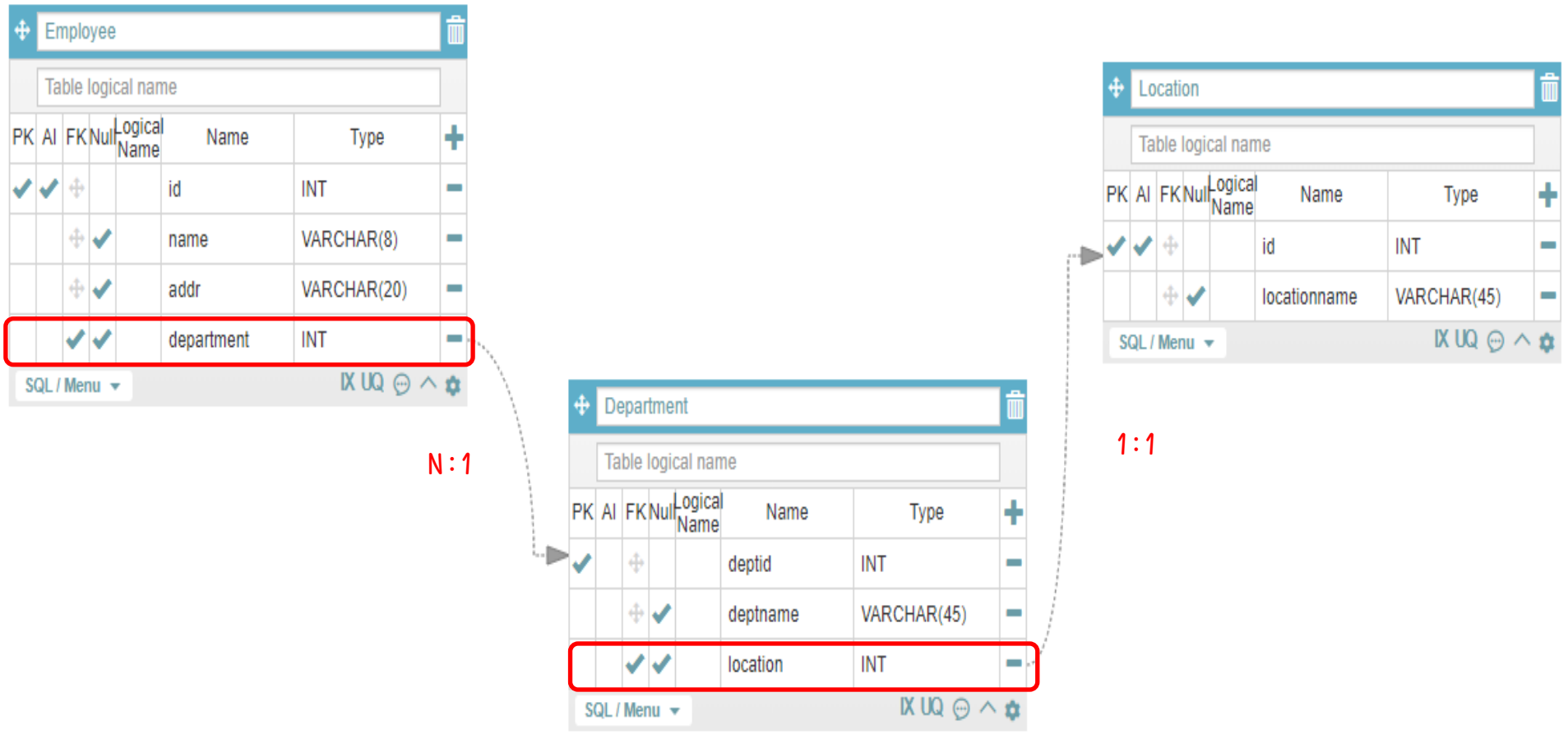
(2) 다음과 같이 makemigrations 명령과 migrate 명령을 실행하고 DB 테이블을 생성한다.

```
(djangovenv) C:\kjh\DJANGOexam\studyproject>python manage.py makemigrations relationapp
Migrations for 'relationapp':
  relationapp\migrations\0001_initial.py
    - Create model Department
    - Create model Location
    - Create model Employee
    - Add field location to department

(djangovenv) C:\kjh\DJANGOexam\studyproject>

(djangovenv) C:\kjh\DJANGOexam\studyproject>python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, forthapp, relationapp, sessions, thirdapp
Running migrations:
  Applying relationapp.0001_initial... OK

(djangovenv) C:\kjh\DJANGOexam\studyproject>
```

```
from django.db import models

class Location(models.Model) :
    locationname = models.CharField(max_length=30)
    def __str__(self):
        return    f"id={self.id}, locationname={self.locationname}"

class Department(models.Model) :
    deptid = models.IntegerField(primary_key=True)
    deptname = models.CharField(max_length=30)
    location = models.OneToOneField(Location, on_delete=models.CASCADE)
    def __str__(self):
        return f"deptid={self.deptid}, deptname={self.deptname}, location={self.location}"

class Employee(models.Model) :
    name = models.CharField(max_length=8)
    addr = models.CharField(max_length=20)
    department = models.ForeignKey(Department, on_delete=models.SET_NULL, null=True)
    def __str__(self):
        return    f"name={self.name}, addr={self.addr}, department={self.department}"
```


SQLite¥db¥relationapp_employee¥ - HeidiSQL 11.2.0.6213
파일 편집 검색 쿼리 도구 이동 도움말
Donate
데이터베이스 필터
테이블 필터
호스트: C:¥kjh¥DJANGOexam¥studyproject¥db.sqlite3
데이터베이스: db
테이블: relationapp_employee
데이터
test.sql
SQLite 0 B
db 0 B
auth_group
auth_group_permissions
auth_permission
auth_user
auth_user_groups
auth_user_user_permissions
django_admin_log
django_content_type
django_migrations
django_session
fifthapp_todo
forthapp_meeting
relationapp_department
relationapp_employee
relationapp_location
thirdapp_dbtest
thirdapp_emp
uploadapp_upload
visitorapp_reply
visitorapp_visitor
기본 옵션 인덱스 (2) 외래 키 (1) Check constraints (0) 분할 CREATE 코드 ALTER 코드
이름: relationapp_employee
코멘트:
열: 추가 제거 위로 아래로
이름 데이터 유형 길이/설정 부호 ... NULL ... 0으... 기본값 코멘
1 id INTEGER 기본값 없음
2 name VARCHAR 8 기본값 없음
3 addr VARCHAR 20 기본값 없음
4 department_id INTEGER 기본값 없음
도움말 되돌리기 저장
필터: 정규 표현식
/* Unknown datatype "". Fall back to UNKNOWN. */
SELECT * FROM "db".pragma_index_list('relationapp_employee') WHERE origin!='pk';
SELECT * FROM "db".pragma_index_info('relationapp_employee_department_id_51852837');
SELECT * FROM "db".pragma_foreign_key_list('relationapp_employee');
SELECT "sql" FROM "db".sqlite_master WHERE "type"='table' AND name='relationapp_employee';
연결됨: 00:03 h SQLite 3.34.0 가동 시간: 알 수 없음 서버 시간: 오후 유휴
db¥relationapp_department¥ - HeidiSQL 11.2.0.6213
테이블: relationapp_department
relationapp_department
이름 데이터 유형 길이/설정 부호 ... NULL ... 0으... 기본값 코멘
1 deptid INTEGER 기본값 없음
2 deptname VARCHAR 30 기본값 없음
3 location_id BIGINT 기본값 없음
_index_info('sqlite_autoindex_relationapp_department_1');
_foreign_key_list('relationapp_department');
lite_master WHERE "type"='table' AND name='relationapp_department';
lite_master WHERE "type"='table' AND name='relationapp_employee';
lite_master WHERE "type"='table' AND name='relationapp_department';
relationapp_location
열: 추가 제거 위로 아래로
이름 데이터 유형 길이/설정 부호 ... NULL ... 0으... 기본값 코멘트
1 id INTEGER 기본값 없음
2 locationname VARCHAR 30 기본값 없음
SELECT * FROM "db".pragma_foreign_key_list('relationapp_department');
SELECT "sql" FROM "db".sqlite_master WHERE "type"='table' AND name='relationapp_department';
SELECT "sql" FROM "db".sqlite_master WHERE "type"='table' AND name='relationapp_employee';
SELECT "sql" FROM "db".sqlite_master WHERE "type"='table' AND name='relationapp_department';
SELECT "sql" FROM "db".sqlite_master WHERE "type"='table' AND name='relationapp_location';

[Employee, Department, Location 테이블의 ERD]



```
(djangovenv) C:\kjh\DJANGOexam\studyproject>python manage.py shell
Python 3.8.7 (tags/v3.8.7:6503f05, Dec 21 2020, 17:59:51) [MSC v.1928 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>> from relationapp.models import Employee, Department, Location
>>> Employee.objects.all()
<QuerySet []>
>>> Department.objects.all()
<QuerySet []>
>>> Location.objects.all()
<QuerySet []>



>>> l1 = Location(locationname='서울')
>>> l2 = Location(locationname='부산')
>>> l3 = Location(locationname='대전')
>>> l4 = Location(locationname='광주')
>>> l5 = Location(locationname='제주도')
>>> l1.save();l2.save();l3.save();l4.save();l5.save()
>>> Location.objects.all()
<QuerySet [<Location: id=1, locationname=서울>, <Location: id=2, locationname=부산>, <Loc
ation: id=3, locationname=대전>, <Location: id=4, locationname=광주>, <Location: id=5, lo
cationname=제주도>]>
```

```
>>> d1 = Department(deptid=10, deptname='영업부', location=l1)
>>> d2 = Department(deptid=20, deptname='총무부', location=l2)
>>> d3 = Department(deptid=30, deptname='기획부', location=l3)
>>> d4 = Department(deptid=40, deptname='개발부', location=l4)
>>> d5 = Department(deptid=50, deptname='마케팅부', location=l1)
>>> d1.save();d2.save();d3.save();d4.save()
```

```
>>> d5.save()
Traceback (most recent call last):
  File "C:\kjh\python_venv\djangovenv\lib\site-packages\django\db\backends\utils.py", lin
e 84, in _execute
    return self.cursor.execute(sql, params)
  File "C:\kjh\python_venv\djangovenv\lib\site-packages\django\db\backends\sqlite3\base.p
y", line 423, in execute
    return Database.Cursor.execute(self, query, params)
sqlite3.IntegrityError: UNIQUE constraint failed: relationapp_department.location_id
```

```
>>> Department.objects.all()
<QuerySet [<Department: deptid=10, deptname=영업부, location=id=1, locationname=서울>, <D
epartment: deptid=20, deptname=총무부, location=id=2, locationname=부산>, <Department: de
ptid=30, deptname=기획부, location=id=3, locationname=대전>, <Department: deptid=40, dept
name=개발부, location=id=4, locationname=광주>]>
```

```
>>> e1 = Employee(name="둘리", addr="쌍문동", department=d1)
>>> e2 = Employee(name="또치", addr="아프리카", department=d2)
>>> e3 = Employee(name="도우너", addr="깐따삐아별", department=d3)
>>> e4 = Employee(name="마이콜", addr="미국", department=d4)
>>> e5 = Employee(name="올라프", addr="겨울왕국", department=d1)
>>> e6 = Employee(name="듀크", addr="삼성동", department=d1)
>>> e1.save();e2.save();e3.save();e4.save();e5.save();e6.save()
```

```
>>> Employee.objects.all()
<QuerySet [<Employee: name=둘리, addr=쌍문동, department=deptid=10, deptname=
영업부, location=id=1, locationname=서울>, <Employee: name=또치, addr=아프리
카, department=deptid=20, deptname=총무부, location=id=2, locationname=부산>,
 <Employee: name=도우너, addr=깐따삐아별, department=deptid=30, deptname=기획
부, location=id=3, locationname=대전>, <Employee: name=마이콜, addr=미국, dep
artment=deptid=40, deptname=개발부, location=id=4, locationname=광주>, <Emplo
yee: name=올라프, addr=겨울왕국, department=deptid=10, deptname=영업부, locat
ion=id=1, locationname=서울>, <Employee: name=듀크, addr=삼성동, department=d
eptid=10, deptname=영업부, location=id=1, locationname=서울>]>
```

```
>>> Employee.objects.get(id=1)
<Employee: name=둘리, addr=쌍문동, department=deptid=10, deptname=영업부, loc
ation=id=1, locationname=서울>
>>> emp = Employee.objects.get(id=1)
>>> emp.name
'둘리'
>>> emp.addr
'쌍문동'
>>> emp.department.deptid
10
>>> emp.department.deptname
'영업부'
>>> emp.department.location.id
1
>>> emp.department.location.locationname
'서울'



>>> dept = Department.objects.get(deptid=20)
>>> dept
<Department: deptid=20, deptname=총무부, location=id=2, locationname=부산>
>>> dept = Department.objects.get(pk=20)
>>> dept
<Department: deptid=20, deptname=총무부, location=id=2, locationname=부산>
>>>
>>> dept.deptname
'총무부'
>>> dept.location.locationname
'부산'
```

```
>>> dept = Department.objects.get(pk=10)
>>> dept.deptname
'영업부'
>>> dept.location.locationname
'서울'
>>> dept.employee_set.all()
<QuerySet [<Employee: name=둘리, addr=쌍문동, department=deptid=10, deptname=
영업부, location=id=1, locationname=서울>, <Employee: name=올라프, addr=겨울
왕국, department=deptid=10, deptname=영업부, location=id=1, locationname=서울
>, <Employee: name=듀크, addr=삼성동, department=deptid=10, deptname=영업부,
location=id=1, locationname=서울>]>

>>> result = dept.employee_set.all()
>>> for d in result:
...     print(d)
...
name=둘리, addr=쌍문동, department=deptid=10, deptname=영업부, location=id=1, locationname=서울
name=올라프, addr=겨울왕국, department=deptid=10, deptname=영업부, location=id=1, locationname=서울
name=듀크, addr=삼성동, department=deptid=10, deptname=영업부, location=id=1, locationname=서울



>>> lo = Location.objects.get(locationname='서울')
>>> lo
<Location: id=1, locationname=서울>
>>> lo.department
<Department: deptid=10, deptname=영업부, location=id=1, locationname=서울>
>>> lo.department.employee_set
<django.db.models.fields.related_descriptors.create_reverse_many_to_one_manager.<locals>.RelatedMana
ger object at 0x0000020DF8AF90A0>
>>> lo.department.employee_set.all()
<QuerySet [<Employee: name=둘리, addr=쌍문동, department=deptid=10, deptname=영업부, location=id=1,
locationname=서울>, <Employee: name=올라프, addr=겨울왕국, department=deptid=10, deptname=영업부, lo
cation=id=1, locationname=서울>, <Employee: name=듀크, addr=삼성동, department=deptid=10, deptname=
영업부, location=id=1, locationname=서울>]>
```