

공개SW (Open Source SW)를 중심으로 하는

공간정보 빅데이터 분석 및 실습

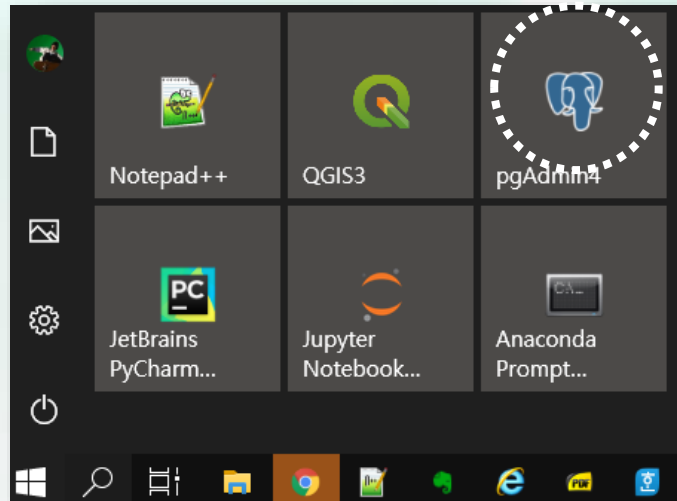
05. PostGIS 기반 분석



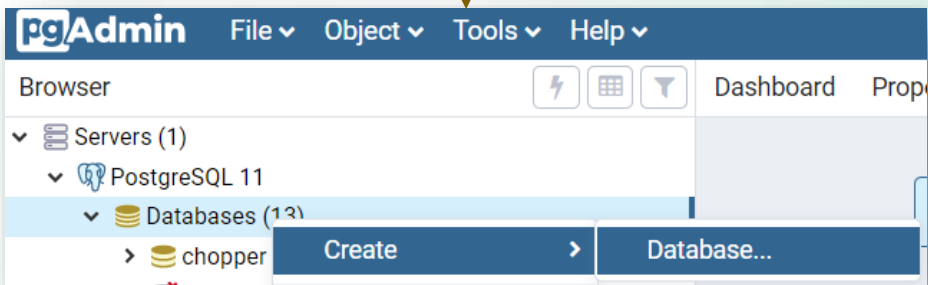
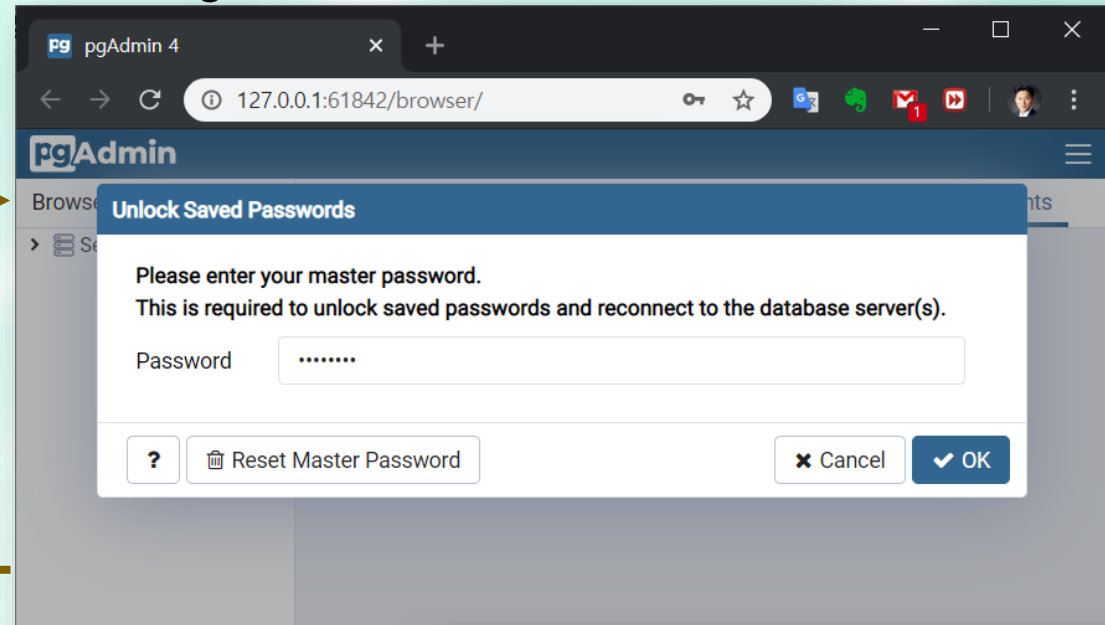
이동훈 (thlee33@gmail.com)

pgAdmin 실행 및 신규 데이터베이스 생성

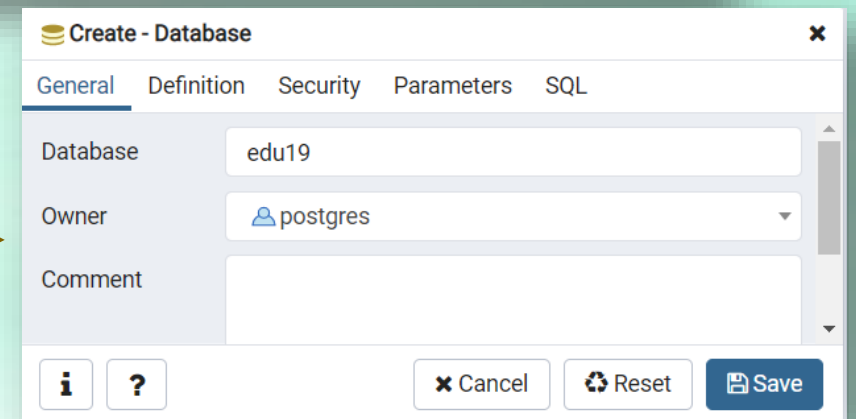
- 일반적인 PostgreSQL 사용 방법과 동일



PostgreSQL admin(관리자) 비밀번호 입력하고 OK



Database 컨텍스트 메뉴 > Create > Database 클릭

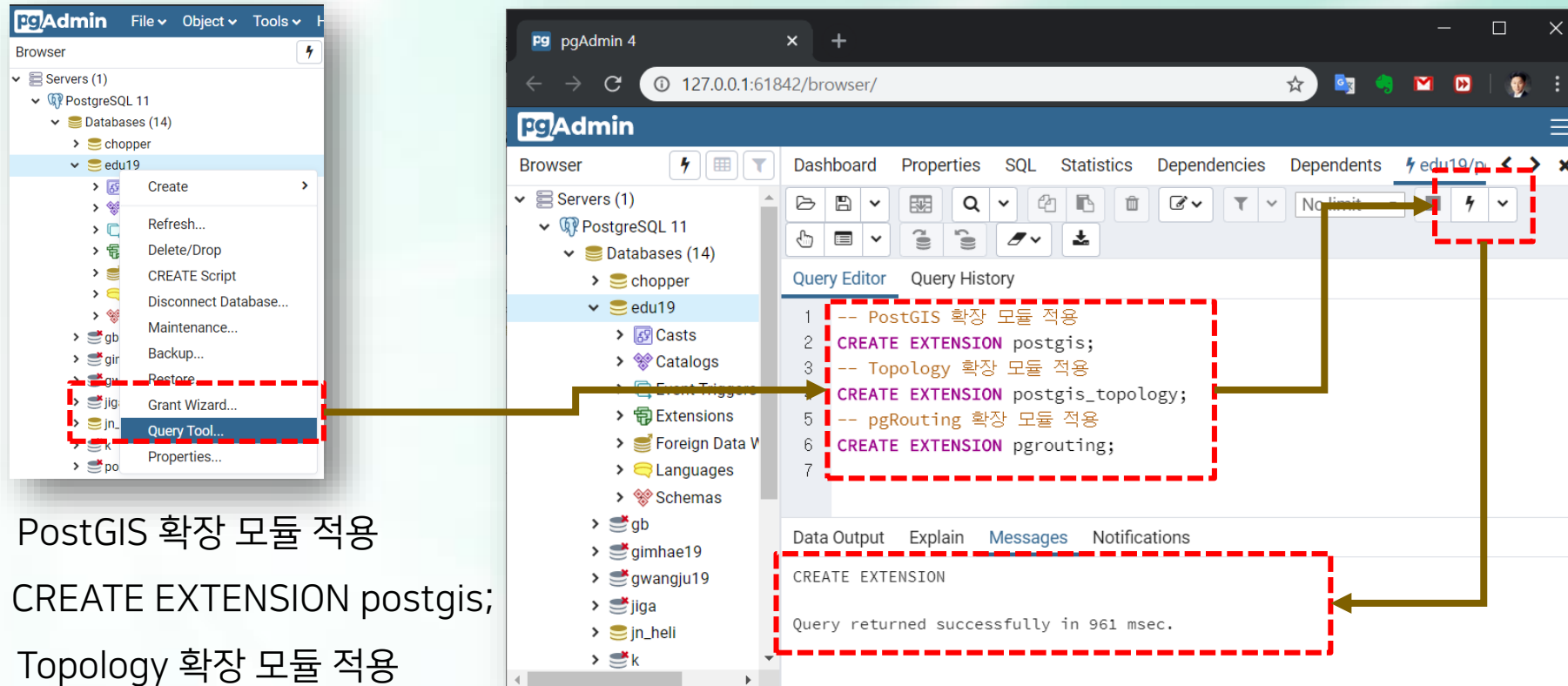


Database 이름을 입력하고 저장 (보안/권한 등 세부설정)

PostGIS 확장 모듈 적용

○ 데이터베이스 Query Tool > PostGIS 확장 모듈 적용 (데이터베이스별로 처음 한 번만)

- Query Editor 화면에 생성 구문을 입력하고, F5 또는 실행 버튼을 클릭
- 우측 하단의 Message에 성공 결과 확인



- PostGIS 확장 모듈 적용
CREATE EXTENSION postgis;
- Topology 확장 모듈 적용
CREATE EXTENSION postgis_topology;
CREATE EXTENSION pgrouting; -- pgRouting 확장 모듈 적용

PostGIS에 한국 좌표계 적용

- 변환계수 적용이 필요한 예전 한국측지계(Bessel 타원체) 좌표계(EPSG 5174 등)에 대한 좌표계 정보를 수정하여 정상적인 좌표 변환 지원
- <https://www.osgeo.kr/205> 에서 제공하는 .sql 파일을 다운로드
- 텍스트 편집기로 열어서 복사하고 pgadmin의 Query Editor에서 실행

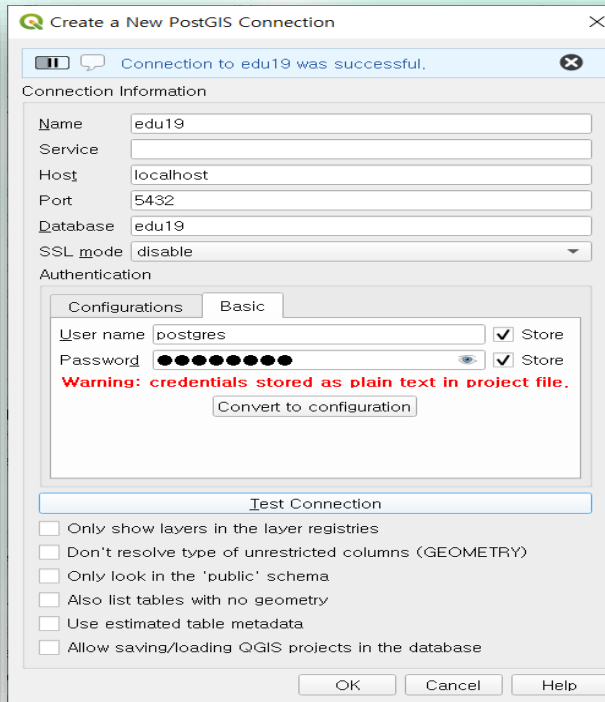
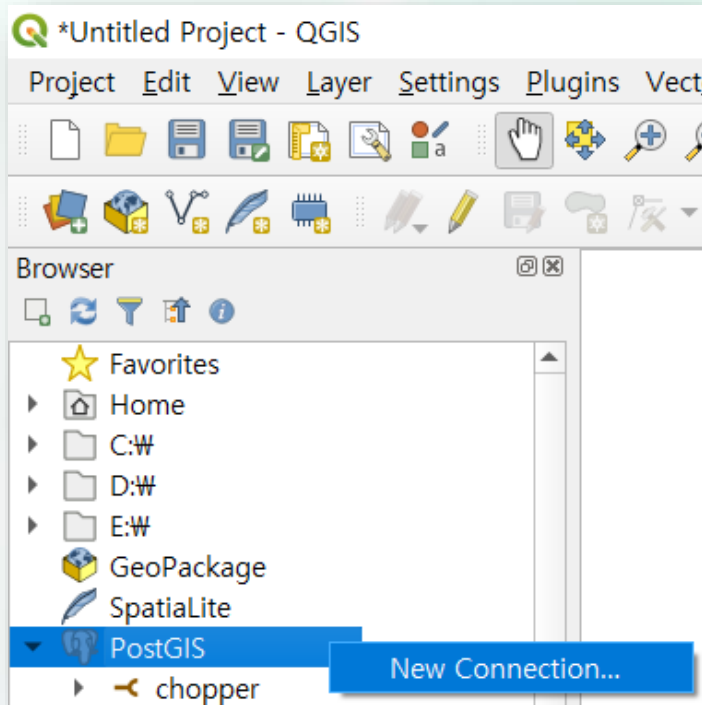
The screenshot shows the pgAdmin 4 web interface. The left sidebar displays the database structure, including Schemas (2) and Tables (6). The main area is the Query Editor, which contains a SQL script for applying Korean coordinate systems to PostGIS. The script includes comments in Korean and SQL commands to create or replace spatial reference systems (SRIDs) for various Korean coordinate systems, including Bessel, WGS 84, and Korean Datum 1985.

```
1 /*=====
2 작성자 : Minpa Lee, mapplus@gmail.com, OSGeo Korea
3 설 명 : PostGIS Korean 좌표계 등록 스크립트 예 2 - TOWGS84 파라미터 적용
4 proj : +towgs84=-115.80,474.99,674.11,1.16,-2.31,-1.63,6.43
5 WKT : TOWGS84[-115.80,474.99,674.11,1.16,-2.31,-1.63,6.43]
6
7 2096=PROJCS["Korean 1985 / East Belt", GEOGCS["Korean 1985", DATUM["Korean Datum 1985", SPHEROID["Bessel 1841", 6377397.155, 299.1
8 2097=PROJCS["Korean 1985 / Central Belt", GEOGCS["Korean 1985", DATUM["Korean Datum 1985", SPHEROID["Bessel 1841", 6377397.155, 29
9 2098=PROJCS["Korean 1985 / West Belt", GEOGCS["Korean 1985", DATUM["Korean Datum 1985", SPHEROID["Bessel 1841", 6377397.155, 299.1
10 4162=GEOGCS["Korean 1985", DATUM["Korean Datum 1985", SPHEROID["Bessel 1841", 6377397.155, 299.1528128, AUTHORITY["EPSG","7004"]],
11 4166=GEOGCS["Korean 1995", DATUM["Korean Datum 1995", SPHEROID["WGS 84", 6378137.0, 298.257223563, AUTHORITY["EPSG","7030"]], TOW
12 4326=GEOGCS["WGS 84", DATUM["World Geodetic System 1984", SPHEROID["WGS 84", 6378137.0, 298.257223563, AUTHORITY["EPSG","7030"]],
13 4326=PROJCS["Korean 1995", DATUM["Korean Datum 1995", SPHEROID["WGS 84", 6378137.0, 298.257223563, AUTHORITY["EPSG","7030"]],
```

srid	auth_name	auth_srid	srtext	proj4text
[PK] integer	character varying (256)	integer	character varying (2048)	character varying (2048)
1	2096	EPSG	2096	PROJCS["Korean 1985 / East B...
2	2097	EPSG	2097	PROJCS["Korean 1985 / Centra...
3	2098	EPSG	2098	PROJCS["Korean 1985 / West B...
4	4927	EPSG	4927	GEOGCS["Korea 2000", DATUM[...
5	5167	EPSG	5167	PROJCS["Korean 1985 / East S...
6	5168	EPSG	5168	PROJCS["Korean 1985 / Centra...
7	5173	EPSG	5173	PROJCS["Korean 1985 / Modifi...
8	5174	EPSG	5174	PROJCS["Korean 1985 / Modifi...
9	5175	EPSG	5175	PROJCS["Korean 1985 / Modifi...
10	5176	EPSG	5176	PROJCS["Korean 1985 / Modifi...
11	5177	EPSG	5177	PROJCS["Korean 1985 / Modifi...

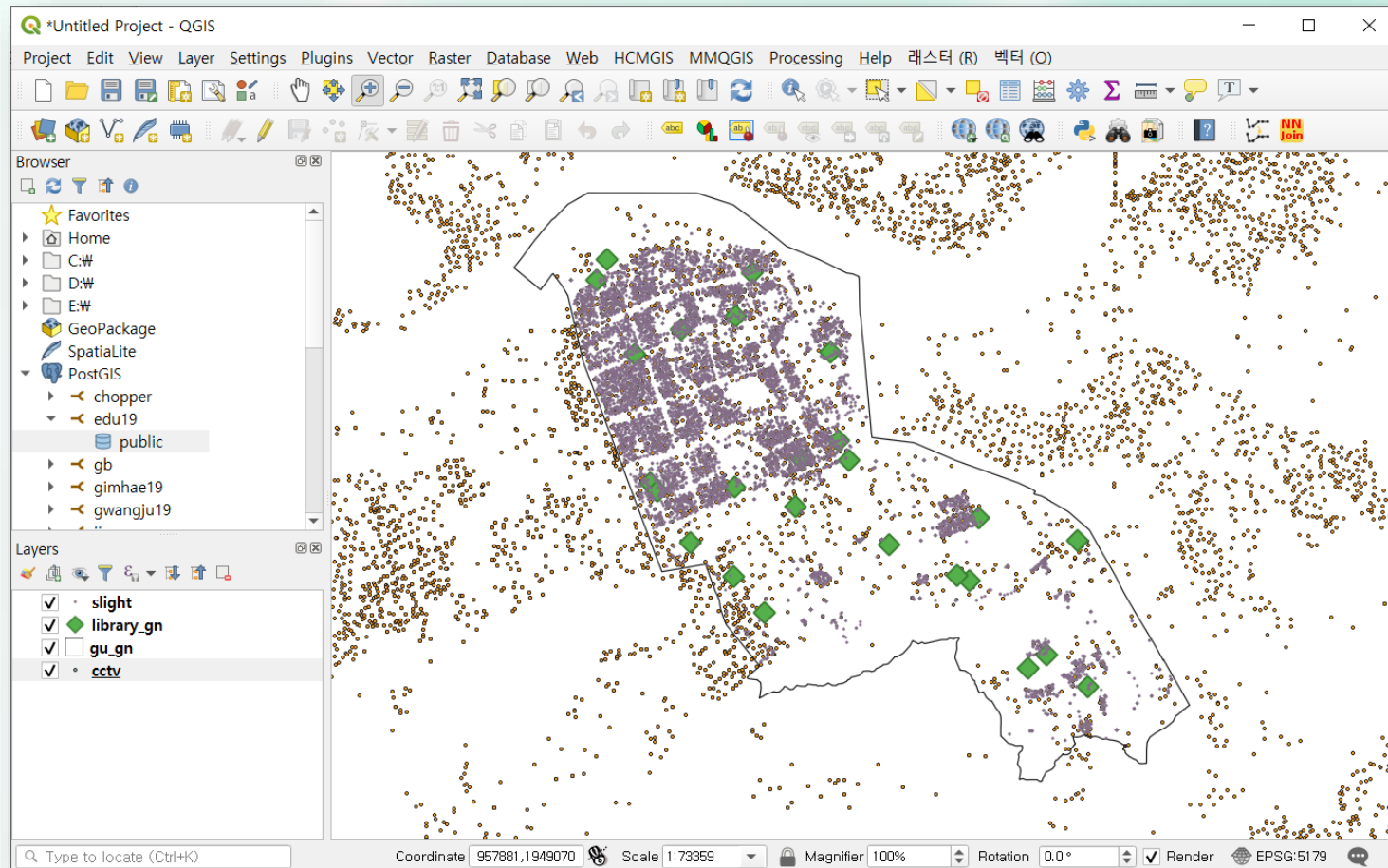
QGIS에서의 PostGIS 연결

- PostGIS는 공간데이터 저장 및 분석 위주 툴로서 시각화를 위한 QGIS 등의 별도의 도구가 필요
- QGIS의 Browser 패널에 있는 PostGIS 컨텍스트 메뉴 > New Connection
(한 번 등록된 데이터베이스는 목록에 나타남)
- 하단 오른쪽과 같이 접속할 데이터베이스의 Host, Port, 데이터베이스명, 계정 정보 입력
- Test Connection 버튼 눌러서 성공 메시지 확인 후, OK



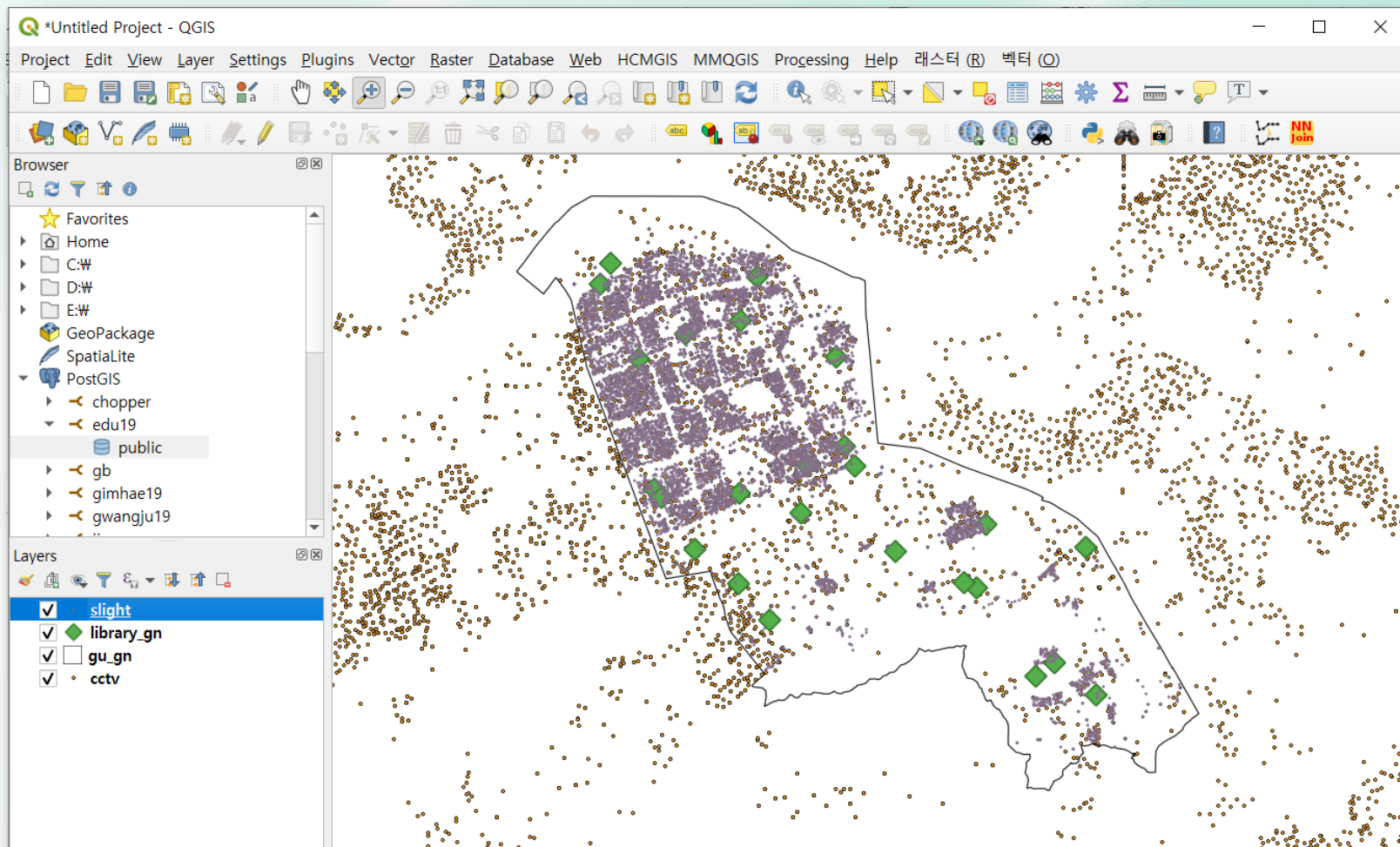
PostGIS에 로딩할 데이터 준비

- 3차시 QGIS 실습시 정제된 보안등(slight), 도서관(library), CCTV(cctv), 강남구 행정구역 데이터를 QGIS에 로딩
 - 4개 데이터 모두 EPSG 5179로 좌표계 통일
 - CCTV만 강남구 외부 데이터를 포함한 버전이고, 나머지는 강남구 내에 있는 정제 데이터



SHP 등의 GIS 공간데이터 로딩/ 조회

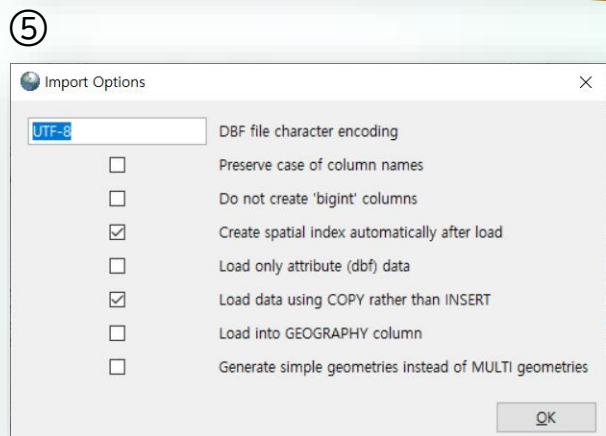
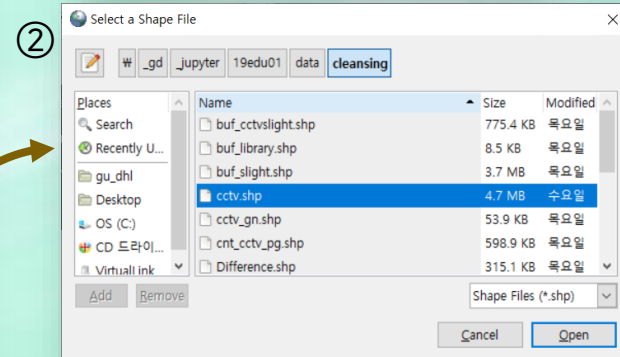
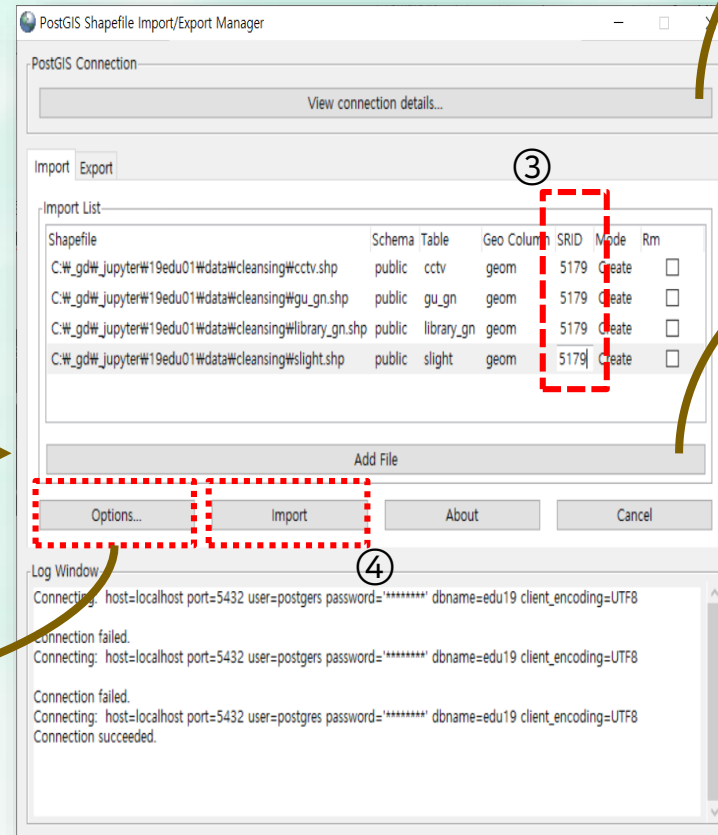
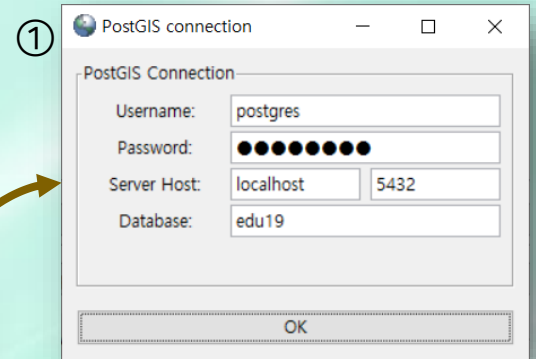
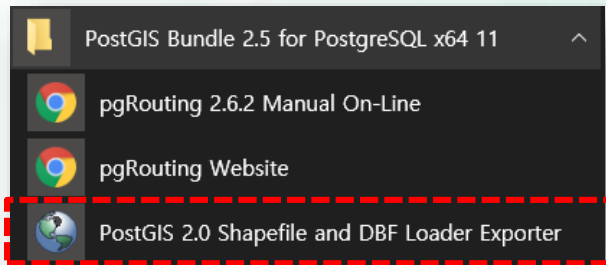
- 준비된 레이어를 PostGIS의 해당 데이터베이스에 Drag & Drop하면 로딩됨
- PostGIS의 공간데이터를 QGIS에서 보려면, 반대로 PostGIS상의 레이어(공간데이터)를 더블클릭하거나 Layers 창에 Drag & Drop하면 됨



PostGIS에서 지원하는 공간데이터 Import/ Export

- 시작 메뉴의 PostGIS 내에 있는 PostGIS Shape and DBF Loader Exporter를 이용

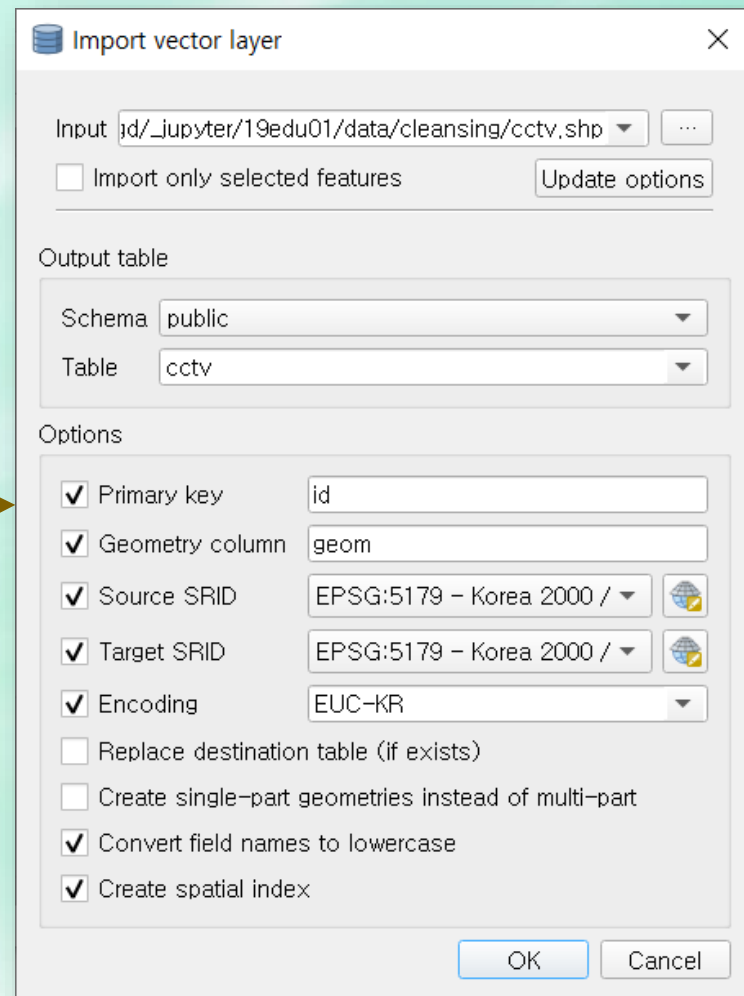
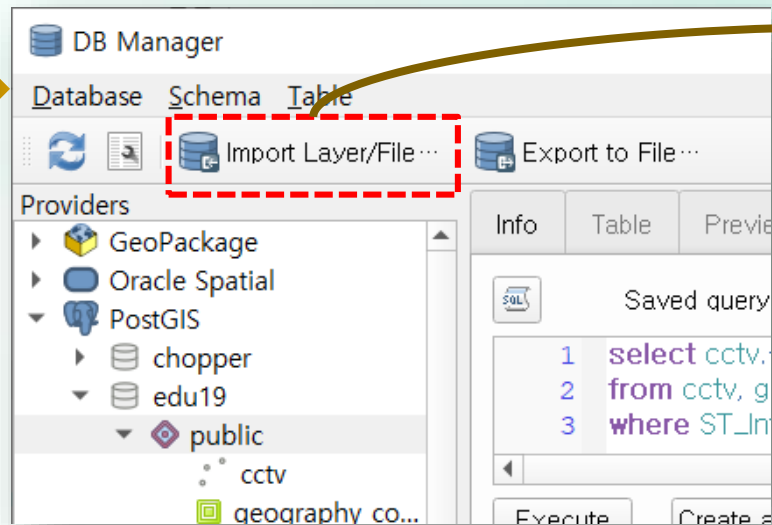
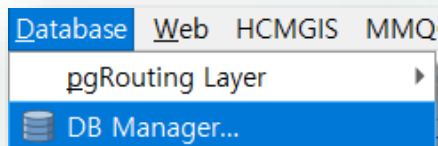
- ① 먼저 데이터베이스 연결 설정
- ② Add File로 로딩할 공간데이터(shp 등)를 선택(ctrl키 누르면 여러 개 파일 선택 가능)
- ③ Import List 확인 및 SRID(좌표계) 입력
- ④ Import 버튼 클릭
- ⑤ 인코딩 변경 시에는 Option 설정 이용




QGIS에서의 또 다른 PostGIS 공간데이터 로딩 방법

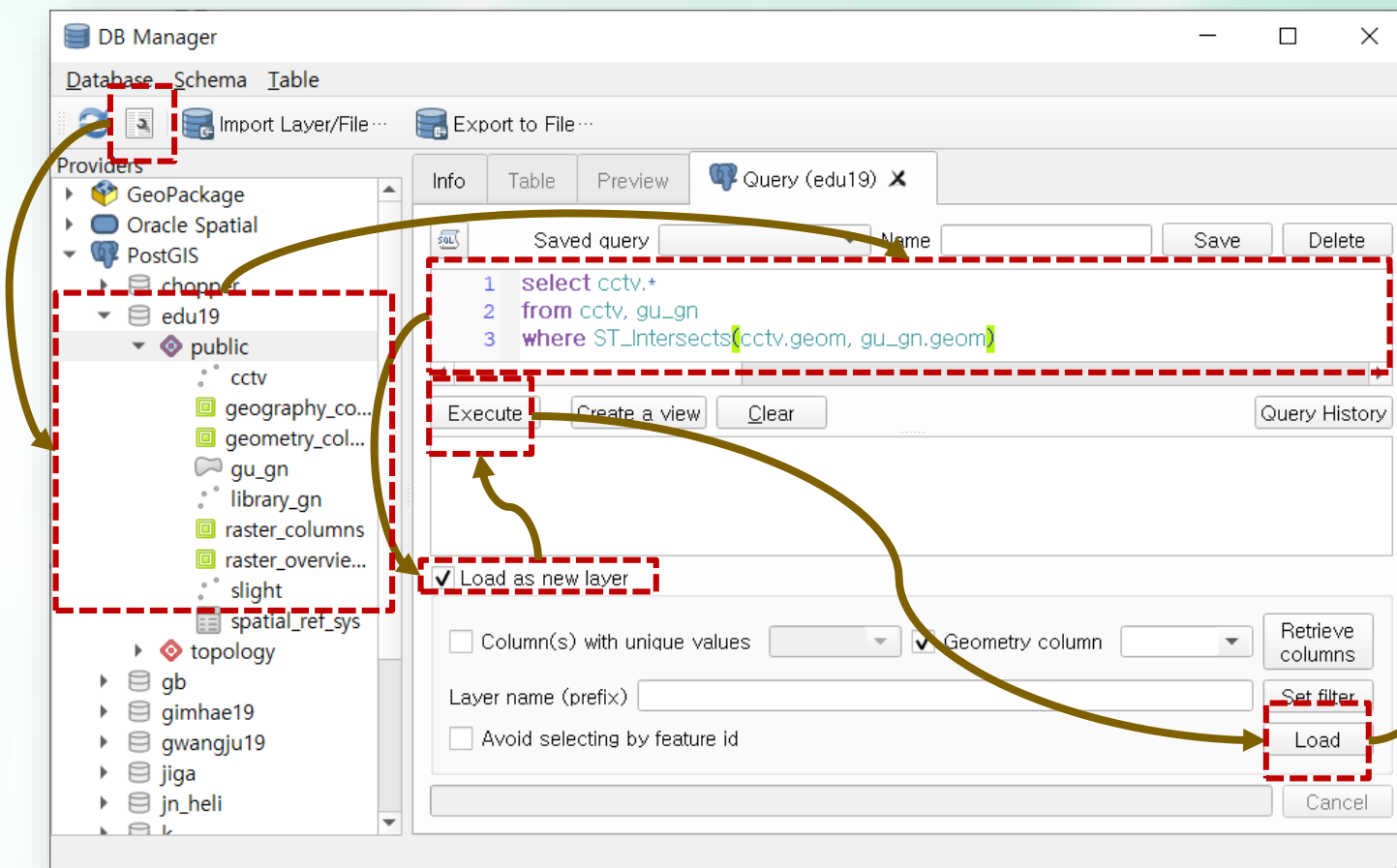
- 다음과 같은 상황에서 유리

- QGIS 상에 로딩되어 있는 레이어를 PostGIS 상에 로딩
- 선택된 데이터만 올릴 때
- 테이블명을 변경
- PK 지정
- 공간 인덱싱을 생성
- 필드명을 소문자로 변환
- 로딩하면서 좌표계를 변환
- 인코딩 문제가 있을 때 캐릭터 셋을 지정

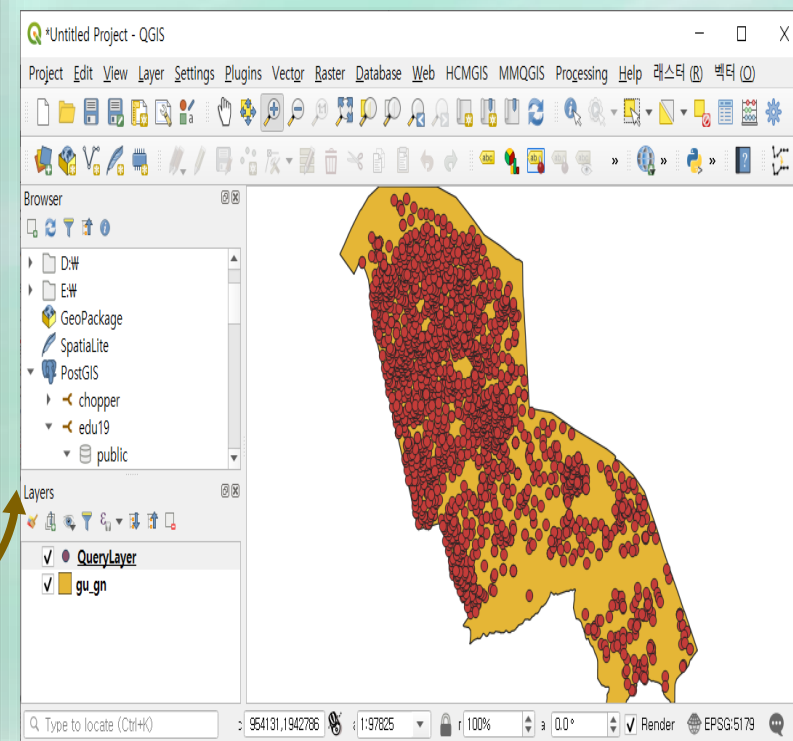


QGIS 상에서의 공간쿼리 및 View 조회

- DB Manager > 원하는 PostGIS 데이터베이스 연결 (더블클릭)
- SQL Window  버튼 클릭하고, 공간 SQL을 작성
- Load as a new layer 체크 > Excute로 SQL 실행 >성공하면 Load 버튼 클릭 > SQL을 만족하는 공간데이터가 지도창에 로딩됨

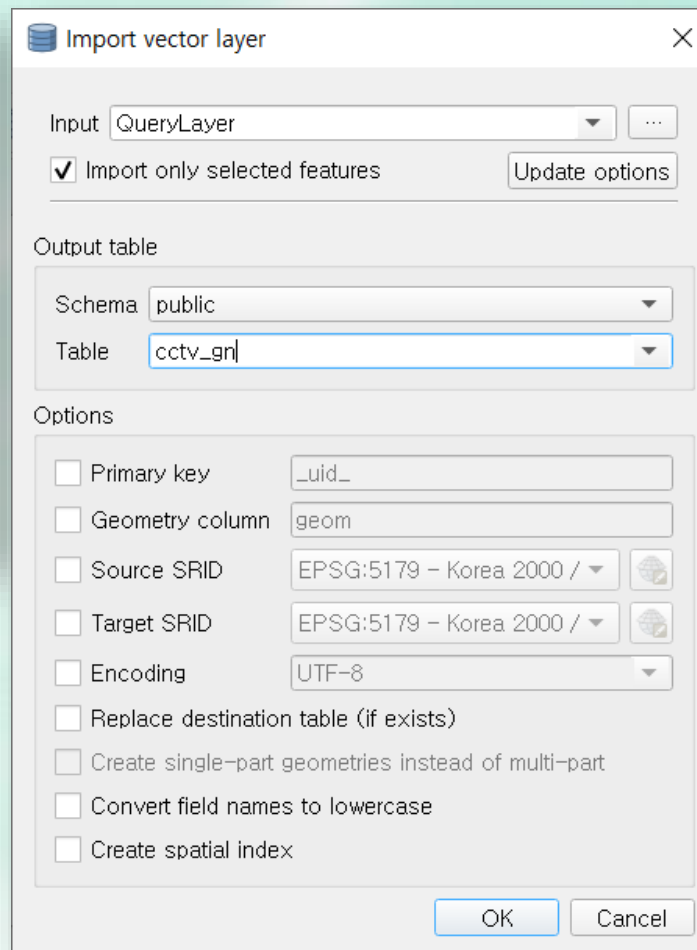
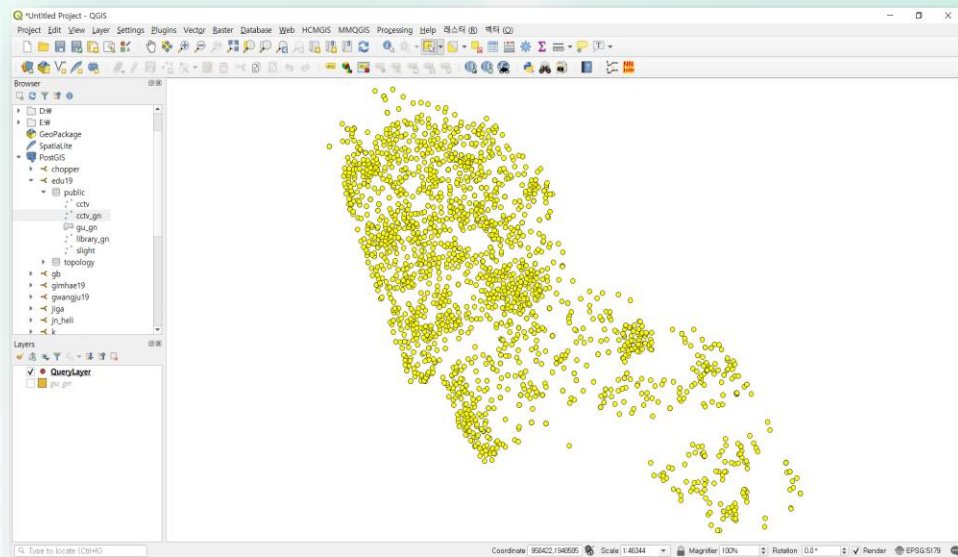


```
select cctv.*
from cctv, gu_gn
where ST_Intersects(cctv.geom, gu_gn.geom)
```



공간쿼리 결과 View를 Table로 저장

- 공간SQL 결과 임시 레이어를 선택
- DB Manager의 Import에서 선택 개체 옵션을 체크하고 로딩



다수의 shp을 로딩하려면

- 아래의 명령을 윈도우 명령프롬프트에서 실행
- `shp2pgsql -I -s 5179 -W cp949 TL_SCCO_SIG.shp | psql -U postgres -d edu19`
 - -I: 인덱스 생성
 - -s: 좌표계 정의
 - -U: DB 계정명
 - -d: 데이터베이스명
- 윈도우 명령프롬프트에서 shp2pgsql을 사용하려면 시스템 환경변수 Path에 postgresql(psql.exe)의 경로를 추가
 - C:\Program Files\PostgreSQL\11\bin(11버전 64bit일 때의 기본 설치 경로)

좌표가 포함된 CSV를 PostGIS에서 공간 테이블로 로딩

- 빈 테이블을 생성(csv와 항목 순서, 데이터 타입 등이 일치해야 함)
- 불필요한 컬럼은 텍스트에디터, QGIS, 스프레드시트, PANDAS 등으로 미리 정리

```
CREATE TABLE  
table_a(pnu char(20), x_coord numeric(13, 6), y_coord numeric(13, 6), 기타 속성 컬럼명 numeric(7, 2) );
```

- CSV 파일을 해당 테이블에 복제, 구분자 및 헤더 등 확인

```
COPY table_a FROM 'C:\data\test.csv' DELIMITERS '|' CSV HEADER;
```

- 테이블에 geom이라는 지오메트리(도형) 컬럼을 EPSG 5179 좌표계로 생성

```
ALTER TABLE table_a  
ADD COLUMN geom geometry(POINT, 5179);
```

- 좌표 컬럼 2개를 이용하여 포인트 도형정보를 geom에 생성

```
UPDATE table_a  
SET geom = ST_SetSRID(ST_MakePoint(X_COORD, Y_COORD) ,5179);
```

※ 저용량일 경우 QGIS에서 공간데이터화 한 후, 로딩해도 동일 효과

- <https://postgis.net/docs/reference.html> 에서 지원되는 공간함수 목록 및 각 목록 클릭 시 설명/예시 등 제공
- 구글 등에서 PostGIS 와 관련 기능(함수명)을 넣어서 검색하는 방법도 가능
- 공간함수를 적용하는 공간테이블은 좌표계가 정의되어 있고, 모두 동일한 좌표계이어야 함
 - 공간함수를 적용할 때 좌표변환 함수를 같이 써서 좌표계를 맞추는 방법도 가능하나 처리 시간이 더 오래 걸림

8.13. Geometry Processing

`ST_Buffer` — (T) Returns a geometry covering all points within a given distance from the input geometry.

`ST_BuildArea` — Creates an areal geometry formed by the constituent linework of given geometry

`ST_Centroid` — Returns the geometric center of a geometry.

`ST_ClipByBox2D` — Returns the portion of a geometry falling within a rectangle.

`ST_ConcaveHull` — The concave hull of a geometry represents a possibly concave geometry that encloses all geometries within the set. You can think of it as shrink wrapping.

`ST_ConvexHull` — Computes the convex hull of a geometry.

`ST_CurveToLine` — Converts a CIRCULARSTRING/CURVEPOLYGON/MULTISURFACE to a LINESTRING/POLYGON/MULTIPOLYGON

`ST_DelaunayTriangles` — Return a Delaunay triangulation around the given input points.

`ST_Difference` — Returns a geometry that represents that part of geometry A that does not intersect with geometry B.

`ST_FlipCoordinates` — Returns a version of the given geometry with X and Y axis flipped. Useful for people who have built latitude/longitude features and need to fix them.

`ST_GeneratePoints` — Converts a polygon or multi-polygon into a multi-point composed of randomly location points within the original areas.

`ST_GeometricMedian` — Returns the geometric median of a MultiPoint.

`ST_Intersection` — (T) Returns a geometry that represents the shared portion of geomA and geomB.

`ST_LineToCurve` — Converts a LINESTRING/POLYGON to a CIRCULARSTRING, CURVEPOLYGON

`ST_MakeValid` — Attempts to make an invalid geometry valid without losing vertices.

`ST_MemUnion` — Same as `ST_Union`, only memory-friendly (uses less memory and more processor time).

`ST_MinimumBoundingCircle` — Returns the smallest circle polygon that can fully contain a geometry. Default uses 48 segments per quarter circle.

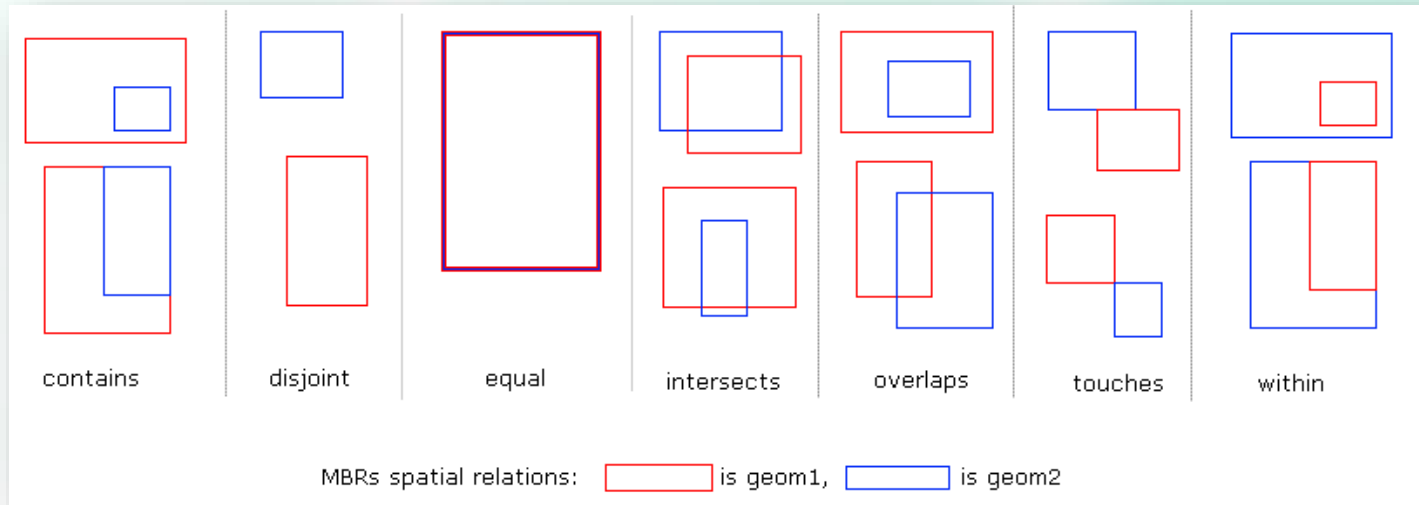
`ST_MinimumBoundingRadius` — Returns the center point and radius of the smallest circle that can fully contain a geometry.

`ST_OrientedEnvelope` — Returns a minimum rotated rectangle enclosing a geometry.

<PostGIS 공간함수 목록 중 일부 화면>

PostGIS 공간함수

- <https://postgis.net/docs/reference.html> 에서 지원되는 공간함수 목록 및 각 목록 클릭 시 설명/예시 등 제공
- 구글 등에서 PostGIS 와 관련 기능(함수명)을 넣어서 검색하는 방법도 가능
- 공간함수를 적용하는 공간테이블은 좌표계가 정의되어 있고, 모두 동일한 좌표계이어야 함
 - 공간함수를 적용할 때 좌표변환 함수를 같이 써서 좌표계를 맞추는 방법도 가능하나 처리 시간이 더 오래 걸림



Equals: 두 개의 Geometry가 동일한가?
Disjoint: 두 개의 Geometry가 서로 격리되었는가?
Intersects: 두 개의 Geometry가 교차하는가?
Touches: 두 개의 Geometry가 접촉하는가?
Crosses: 두 개의 Geometry가 횡단하는가?
Within: 하나의 Geometry가 다른 하나에 포함되는가?
Contains: 하나의 Geometry가 다른 하나를 포함하는가?
Overlaps: 두 개의 Geometry가 부분적으로 겹치는가?

<PostGIS 공간함수 목록 중 일부 화면>

공간테이블의 좌표계 확인

- select * from geometry_columns;

Query Editor

Query History

1

select * from geometry_columns;

2

3차원 데이터도 저장 가능

테이블명

지오메트리 컬럼

차원

좌표계

도형 타입

Data Output

Explain

Messages

Notifications

	f_table_catalog	f_table_schema	f_table_name	f_geometry_column	coord_dimension	srid	type
	character varying	name	name	name	integer	integer	character varying (30)
1	edu19	public	gu_gn	geom	2	5179	MULTIPOLYGON
2	edu19	public	cctv	geom	2	5179	MULTIPOINT
3	edu19	public	library_gn	geom	2	5179	POINT
4	edu19	public	slight	geom	2	5179	POINT
5	edu19	public	cctv_gn	geom	2	5179	MULTIPOINT

점 : POINT(0 0)
선 : LINESTRING(0 0,1 1,1 2)
면: POLYGON((0 0,4 0,4 0,0 0),(1 1, 2 1, 2 2, 1 2,1 1))
MULTIPOINT((0 0),(1 2))
MULTILINESTRING((0 0,1 1,1 2),(2 3,3 2,5 4))
MULTIPOLYGON(((0 0,4 0,4 0,0 0),(1 1,2 1,2 2,1 2,1 1)), ((-1 -1,-1 -2,-2 -2,-2 -1,-1 -1)))
GEOMETRYCOLLECTION(POINT(2 3),LINESTRING(2 3,3 4))

“MULTI”가 붙은 것은 하나의 레코드에 2개 이상의 도형을 담는 방식으로 예를 들어 수백 개의 섬과 육지 영역을 하나의 전라남도 행정구역으로 묶을 수 있는 방식

좌표계 변환 방법

○ 공간테이블의 좌표계를 변환

alter table 테이블명 -변경할 테이블 지정

alter column geom type geometry(POINT, 4326) - 도형(geometry)이 들어있는 geom 컬럼에 변경될 좌표계를 지정 (POINT는 기존 도형타입과 동일해야 함)

using ST_Transform(geom, 4326) --좌표변환 공간함수로 실제 변환이 이뤄짐

○ 좌표계 정보 적용/업데이트

- SELECT UpdateGeometrySRID('public', gu_gn ', 'geom', 5179);

○ geom 도형을 이용하여 좌표 속성을 생성

- alter table 테이블명 add column x double precision; → x 컬럼 생성
- alter table 테이블명 add column y double precision; → y 컬럼 생성
- update 테이블명 set x = ST_X(geom), y=ST_Y(geom); → x, y 컬럼에 좌표 속성 입력
- 위의 SQL을 응용해서 4326 좌표계로 변환해서 속성으로 넣을 경우 →
update 테이블명 set x = ST_X(ST_Transform(geom, 4326)), y = ST_Y(ST_Transform(geom, 4326));

➤ 공간 쿼리를 하기 전 좌표계 부여, 대상 테이블들은 동일한 좌표계로 통일

CCTV Spatial SQL

- 서울시 실폭도로(Z_KAIS_TL_SPRD_RW_11000.shp)를(EPSG:5181로 PostGIS에 로딩하고), EPSG:5179로 변환하면서 강남구 영역만 추출

```
create table rd_gn as
select rd_sl.*
from rd_sl, gu_gn
where ST_Intersects(ST_Transform(rd_sl.geom, 5179),
gu_gn.geom);
```

```
alter table rd_gn
alter column geom type geometry(MultiPolygon, 5179)
using ST_Transform(geom, 5179);
```

- 도서관(500m) 버퍼 테이블 생성

```
create table library_buf as
select
  library_gn.id, library_gn.name, library_gn.type, ST_Buffer(library_gn.geom, 500)
as geom
from
  library_gn;
```


CCTV Spatial SQL

- CCTV(50m), 보안등(20m) 버퍼 테이블 생성

```
create table cctv_buf as
select
  1 as id,
  ST_Union( ST_Buffer(cctv_gn.geom, 50)) as geom
from cctv_gn;
```

```
create table slight_buf as
select
  1 as id,
  ST_Union( ST_Buffer(slight.geom, 20)) as geom
from slight;
```

CCTV Spatial SQL 2

- 도서관 버퍼에서 실폭도로 부분만 추출

```
create table lib_rd as
select lib.id, lib.name, ST_Intersection(rd.geom, lib.geom) AS geom
from rd_gn as rd, library_buf as lib
where ST_Intersects(lib.geom, rd.geom);
```

- CCTV 버퍼와 보안등 버퍼 데이터를 병합

```
create table cctv_light as
select
  1 as id,
  ST_Union(cctv.geom, light.geom) as geom
from
  cctv_buf as cctv, slight_buf as light;
```

CCTV Spatial SQL 2

- 1 영역에서 2 영역을 제외

```
create table becareful as
select
  lib_rd.id, lib_rd.name,
  ST_Difference(lib_rd.geom, cctv_light.geom) as geom
from
  lib_rd, cctv_light;
```

```
create table becareful2 as
select
  becareful.id, becareful.name,
  ST_Union(becareful.geom) as geom
from
  becareful
group by
  becareful.id, becareful.name;
```

CCTV Spatial SQL 3

- 도서관별 주의구간 면적 정보를 생성

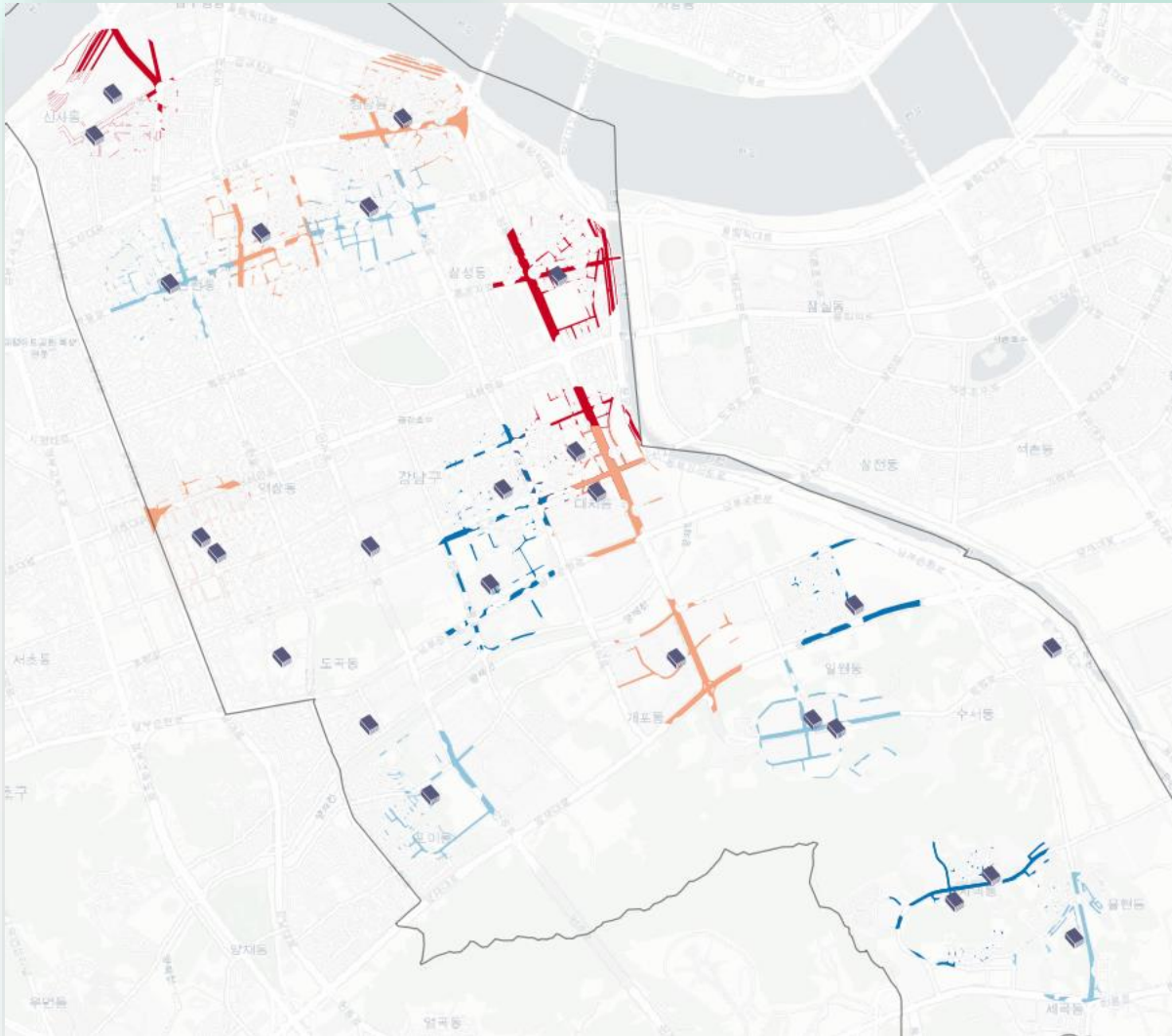
```
create table becareful3 as
select
  becareful2.id, becareful2.name, becareful2.geom,
  ST_Area(geom) as area
from
  becareful2;
```

- 면적이 큰 순위를 부여

```
create table becareful4 as
select
  becareful3.*,
  rank() OVER (order by area desc) AS rank
from becareful3
```

CCTV Spatial SQL 3

- QGIS에서 시각화



주요 공간 SQL

- OSGeo 한국어지부 PostGIS 교육자료

- <https://www.slideshare.net/mapplus/postgis-open-source-gis>

- road_link 테이블에서 차선(lanes)이 6차선 이상인 도로 도형의 합계 길이(km 단위)

- `select sum(ST_Length(geom)) / 1000 as sum_len from road_link where lanes >= 6;`

- 읍면동 행정구역(admin_emd)에서 면적 상위 10개 순대로의 읍면동 이름과 면적

- `select emd_nm, ST_Area(geom) as area from admin_emd order by ST_Area(geom) desc limit 10;`

- 특정 좌표에서 1km 반경 내의 매장 ST_Distance/ ST_Dwithin

- `select * from stores where ST_Distance(geom, ST_GeomFromText('point(197215 447711)', 5174)) < 1000;`

- 영등포구 내에 있는 매장

- `select stores.* from stores, admin_sgg where ST_Intersects(stores.geom, admin_sgg.geom) and admin_sgg.sgg_nm = '영등포구'`

주요 공간 SQL

- 중구의 경위도 중심좌표

- select ST_AsText(ST_Transform(ST_Centroid(admin_sgg.geom), 4326)) from admin_sgg where admin_sgg.sgg_nm = '중구';

- 소방서(firestation)에서 500미터 반경 내의 도로 중 가장 가까운 도로(road_link2)와 거리

- SELECT DISTINCT ON(f.nam) f.nam, r.roadname_a, r.lanes, ST_Distance(r.geom, f.geom) As dist
FROM firestation AS f LEFT JOIN road_link2 As r ON ST_DWithin(r.geom, f.geom, 500)
ORDER BY f.nam, ST_Distance(r.geom, f.geom)
LIMIT 1;

- PostGIS 워크숍

- <https://postgis.net/workshops/postgis-intro/>

PostGIS와 Geopandas 연동

- http://geopandas.org/reference.html?highlight=from_postgis#GeoDataFrame.from_postgis
- psycopg 등 PostgreSQL 연동 라이브러리 필요
 - 아래 소스코드 참조
 - https://github.com/thlee33/geopandas_basic_20191231/blob/master/geopandas_postgis_20191231.ipynb

```
classmethod GeoDataFrame.from_postgis(sql, con, geom_col='geom', crs=None, index_col=None,  
coerce_float=True, parse_dates=None, params=None)
```

Examples

```
>>> sql = "SELECT geom, highway FROM roads"  
Spatialite  
>>> sql = "SELECT ST_Binary(geom) AS geom, highway FROM roads"  
>>> df = geopandas.GeoDataFrame.from_postgis(sql, con)
```