





SGIS 소지역 통계 분석사례집



통계청 통계지리정보서비스(SGIS)는 인구주택총조사, 전국사업체조사 등 센서스 통계의 활용도를 높이기 위해 소지역 통계를 시각화 콘텐츠와 분석용 데이터로 제공합니다. 행정동의 약 1/30 크기인 집계구와 1km, 100m 격자 등 소지역 단위로 집계한 통계를 지도 위에 표현해보면 기존의 집계표로는 파악하기 어려운 새로운 의미를 발견할 수 있습니다.

.....

통계를 지도 위에 표현하기 위해서는 공간데이터 분석 경험이 필요합니다. 공간 속성을 포함한 데이터 구조와 익숙하지 않은 분석도구는 입문자에게는 큰 어려움으로 다가올 수 있습니다. 본 사례집은 공간 분석을 시작하는 이용자가 SGIS 소지역 통계를 활용한 지도 시각화 분석 과정을 단계별로 따라 하면서 공간데이터의 개념을 익히고 QGIS, R 등 대표적인 분석 도구의 기본 사용법을 쉽게 배울 수 있도록 구성되었습니다.

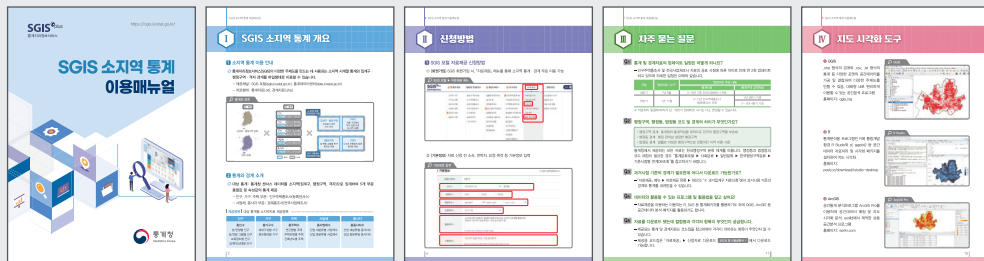
.....

SGIS는 분석사례에 소개된 통계 이외에도 다양한 소지역 통계를 2000년부터 최신 연도까지 시계열로 제공하고 있습니다. 자료제공 메뉴에서 통계와 경계 데이터를 무료로 다운로드한 후, 분석 목적에 맞게 지도 위에 멋지게 시각화하여 다양하게 활용해 보시기 바랍니다.

CONTENTS

Ⅰ 전국 행정동별 교육 서비스업 분석(QGIS)	8
Ⅱ 물금읍 교육 서비스업 격차 분석(QGIS)	20
Ⅲ 서울시 행정동 인구수 현황 분석(Python)	36
Ⅳ 부산시 인구수 현황 격차 분석(Python)	44
Ⅴ 서울시 청년인구 격차별 순위 분석(R)	52
Ⅵ 서울시 인구수 현황 격차 분석(ArcGIS)	62

참고자료 | SGIS 소지역 통계 이용매뉴얼



SGIS 소지역 통계 이용매뉴얼에는 SGIS에서 제공하는 소지역 통계와 경계의 개요와 파일구조, 제공범위와 신청방법 등이 상세하게 안내되어 있습니다. 분석사례집의 시각화 도구별 분석 과정을 이해하는데 도움이 되니 함께 참고해주시기 바랍니다.



모바일 바로 보기

〈자료 위치〉

SGIS 포털 ➔ 「자료제공」 메뉴 ➔ 신청자료 다운로드 ➔ 코드표 및 이용설명서



PART

I

전국 행정동별
교육 서비스업 분석
(QGIS)





2 통계 레이어 추가하기(.txt)

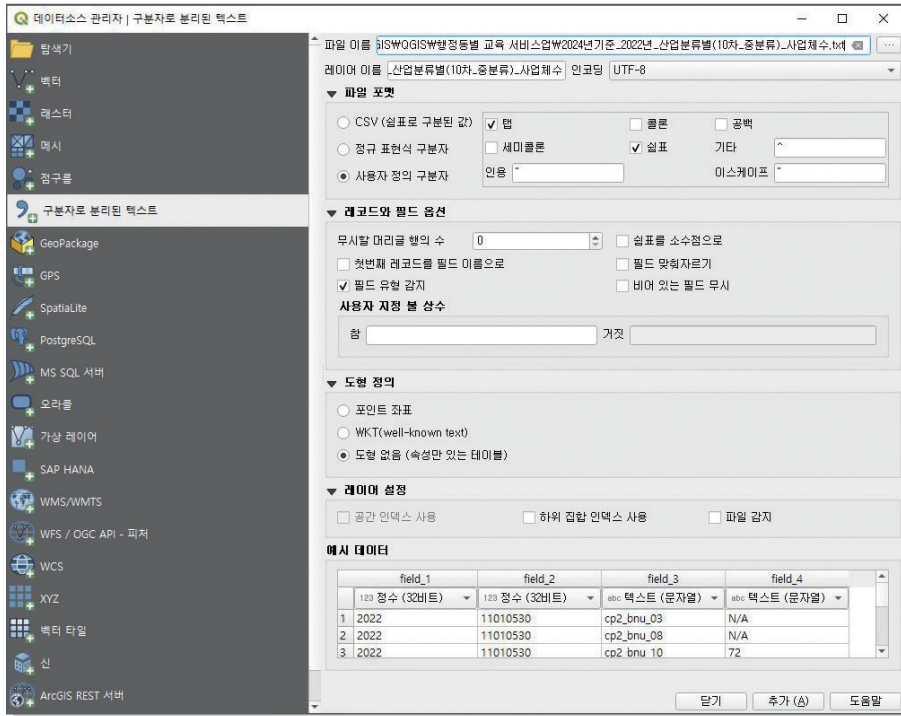
레이어 → 레이어 추가 → 구분자로 분리된 텍스트 레이어 추가 에서 2022년_산업분류별(10차_중분류)

_사업체수 통계를 추가합니다.

※ 파일 포맷 → 사용자 정의 구분자 → 기타 항목에서 구분자를 '^'로 설정

※ CSV 파일을 다운 받은 경우, 파일 포맷 → CSV(실패로 구분된 값) 선택

※ 예시 데이터를 미리 보면서 'windows-949' 또는 'UTF-8' 등 한글이 깨지지 않는 인코딩 선택



통계 레이어(2022년_산업분류별(10차_중분류)_사업체수)를 마우스 오른쪽 클릭해서 속성 테이블 열기 로 사업체 통계를 확인합니다.

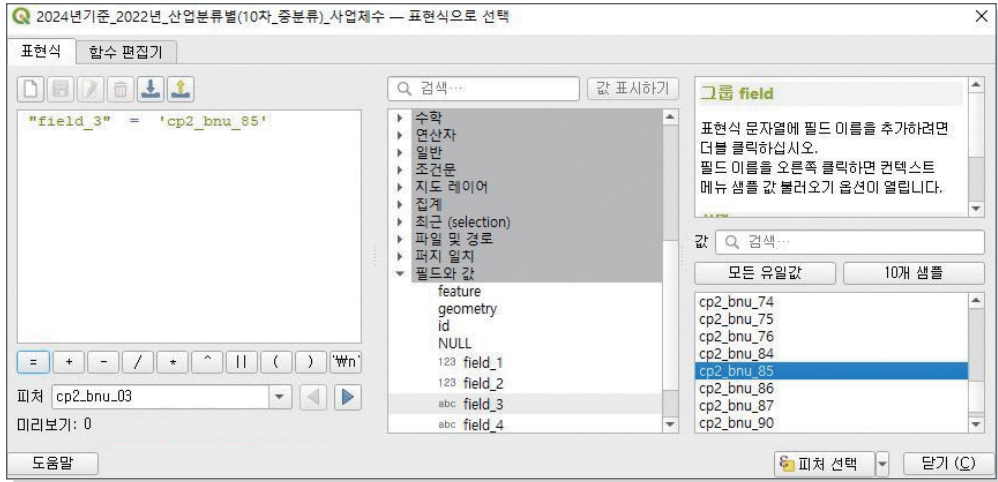
2024년기준_2022년_산업분류별(10차_중분류)_사업체수 — 총 피쳐 개수: 184002, 필터링된 피쳐 개수: 184002, 선택한 피쳐 개수: 0

	field_1	field_2	field_3	field_4
1	2022	11010530	cp2_bnu_03	N/A
2	2022	11010530	cp2_bnu_08	N/A
3	2022	11010530	cp2_bnu_10	72
4	2022	11010530	cp2_bnu_11	N/A
5	2022	11010530	cp2_bnu_13	7
6	2022	11010530	cp2_bnu_14	17
7	2022	11010530	cp2_bnu_15	N/A
8	2022	11010530	cp2_bnu_17	N/A
9	2022	11010530	cp2_bnu_18	21
10	2022	11010530	cp2_bnu_20	8

모든 피쳐 표시

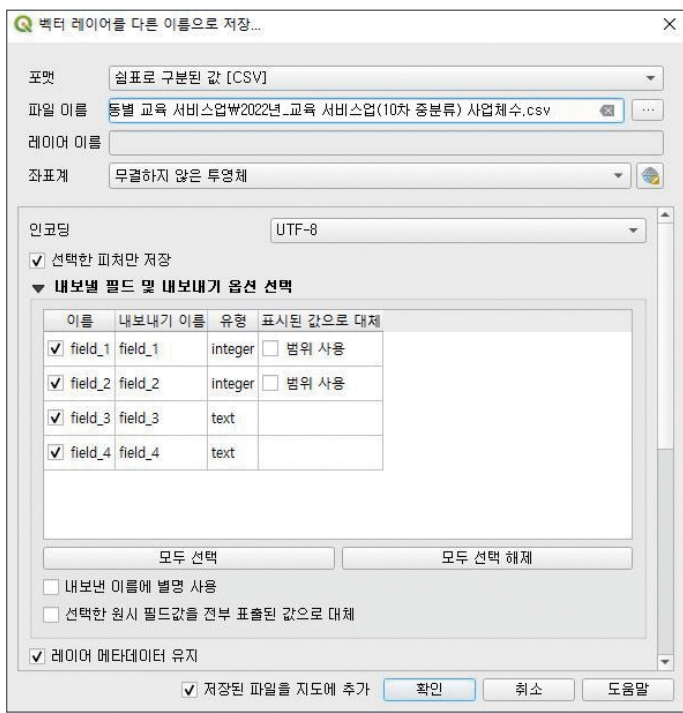
3 통계 레이어에서 속성값(교육 서비스업)에 해당하는 통계항목 선택하기

- ▶ 통계 확인 후, **표현식을 이용해서 피쳐 선택** () 화면에서 세 번째 항목('field_3')의 값을 'cp2_bnu_85'(교육 서비스업)로 선택합니다.



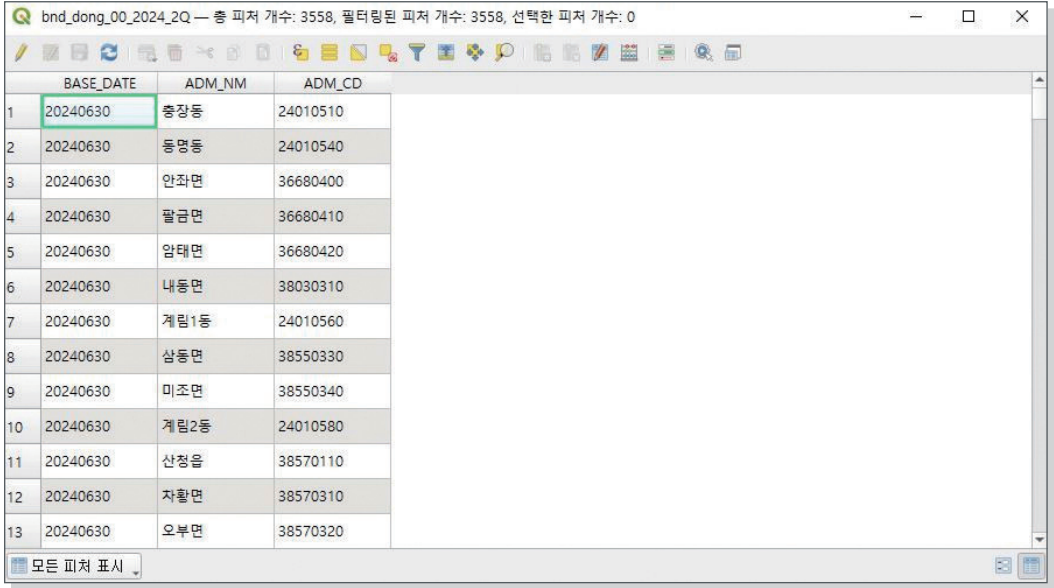
- ※ **모든 유일값** 버튼을 클릭하면 통계속성 코드의 목록을 확인할 수 있고, 음영으로 표시된 부분의 항목을 선택하면 직접 입력하지 않고도 표현식을 작성할 수 있습니다.

- ▶ 2022년 산업분류별(10차 중분류) 사업체수 레이어를 마우스 오른쪽 클릭해서 **내보내기** ➔ **선택한 피처를 다른 이름으로 저장** 합니다.(2022년_교육 서비스업(10차 중분류) 사업체수.csv)



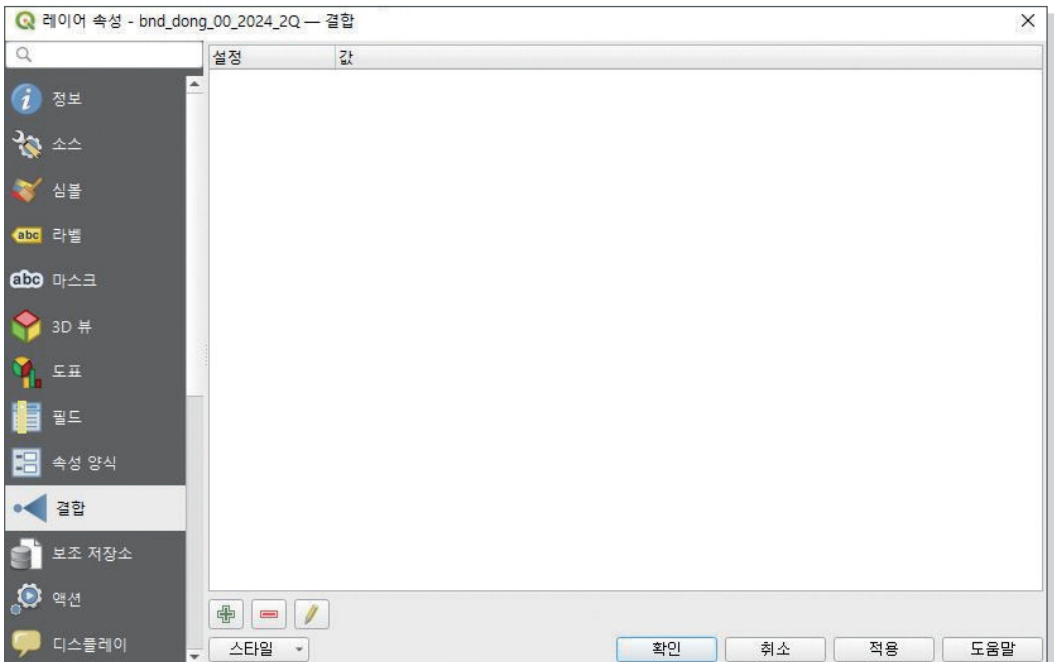
4 전국 행정동 경계와 교육 서비스업 통계 결합하기

- ▶ **bnd_dong_00** 레이어를 마우스 오른쪽 클릭해서 **속성 테이블 열기** 를 선택합니다. 행정동 경계 레이어에는 BASE_DATE(기준일자), ADM_NM(행정동 이름), ADM_CD(행정동 코드)가 있습니다.



	BASE_DATE	ADM_NM	ADM_CD
1	20240630	충장동	24010510
2	20240630	동명동	24010540
3	20240630	안좌면	36680400
4	20240630	팔금면	36680410
5	20240630	암태면	36680420
6	20240630	내동면	38030310
7	20240630	계림1동	24010560
8	20240630	삼동면	38550330
9	20240630	미조면	38550340
10	20240630	계림2동	24010580
11	20240630	산청읍	38570110
12	20240630	차황면	38570310
13	20240630	오부면	38570320

- ▶ **bnd_dong_00** 레이어를 마우스 오른쪽 클릭 후 **속성** 을 선택해서 레이어 속성 화면을 열고 왼쪽 메뉴에서 **결합** , 화면 하단의  버튼을 순서대로 선택합니다.



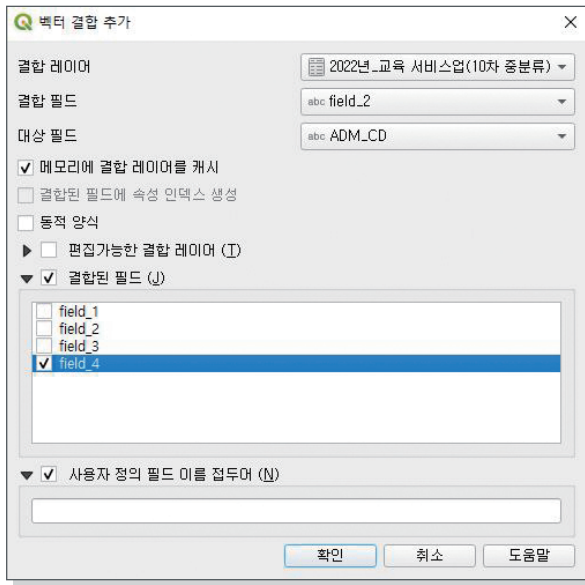
- ▶ 결합 레이어, 결합 필드, 대상 필드, 결합된 필드 등을 적절히 선택합니다.

결합 레이어 : 2022년_교육 서비스업(10차 중분류) 사업체수

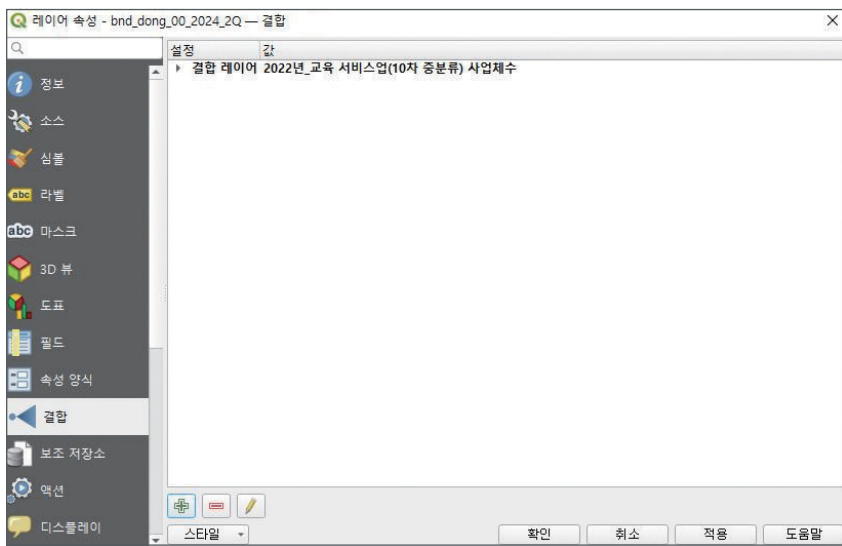
결합 필드 : field_2

대상 필드 : ADM_CD

※ **결합된 필드** 에서 통계값 필드인 'field_4'는 필수 선택이며, **접두어** 는 생략 가능



- ▶ 레이어 속성 화면으로 돌아와서 적용, 확인을 선택합니다.



- ▶ **bnd_dong_00** 레이어를 마우스 오른쪽 클릭해서 **속성 테이블 열기** 를 다시 선택합니다. 통계값 필드(field_4)가 결합되어 있습니다.

bnd_dong_00_2024_2Q — 총 피쳐 개수: 3558, 필터링된 피쳐 개수: 3558, 선택한 피쳐 개수: 0

	BASE_DATE	ADM_NM	ADM_CD	field_4
1	20240630	충장동	24010510	187
2	20240630	동명동	24010540	55
3	20240630	안좌면	36680400	8
4	20240630	팔금면	36680410	N/A
5	20240630	암태면	36680420	5
6	20240630	내동면	38030310	16
7	20240630	계림1동	24010560	47
8	20240630	삼동면	38550330	10
9	20240630	미조면	38550340	8

모든 피쳐 표시

주 의

결합이 안될 경우, **2022년_교육 서비스업(10차 중분류) 사업체수** 레이어의 **속성** → **소스** 에서 인코딩을 'UTF-8'로 적용하고 4단계부터 다시 실시합니다.

레이어 속성 - 2022년_교육 서비스업(10차 중분류) 사업체수 — 소스

정보

소스

심볼

라벨

마스킹

3D 뷰

도표

필드

속성 양식

결합

보조 저장소

▼ 설정

레이어 이름: 2022년_교육 서비스업(10차 중분류) 사업체수

데이터소스 인코딩: UTF-8

▼ 제공자 피쳐 필터

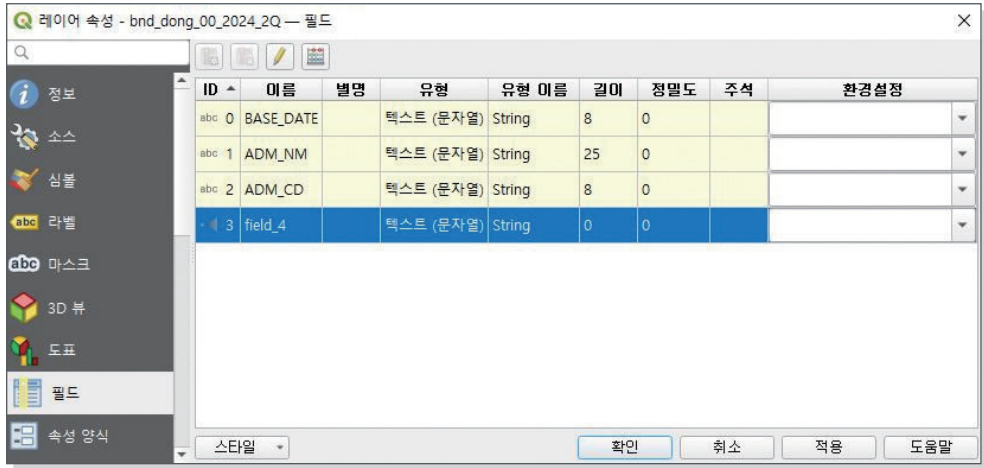
쿼리 작성기

스타일

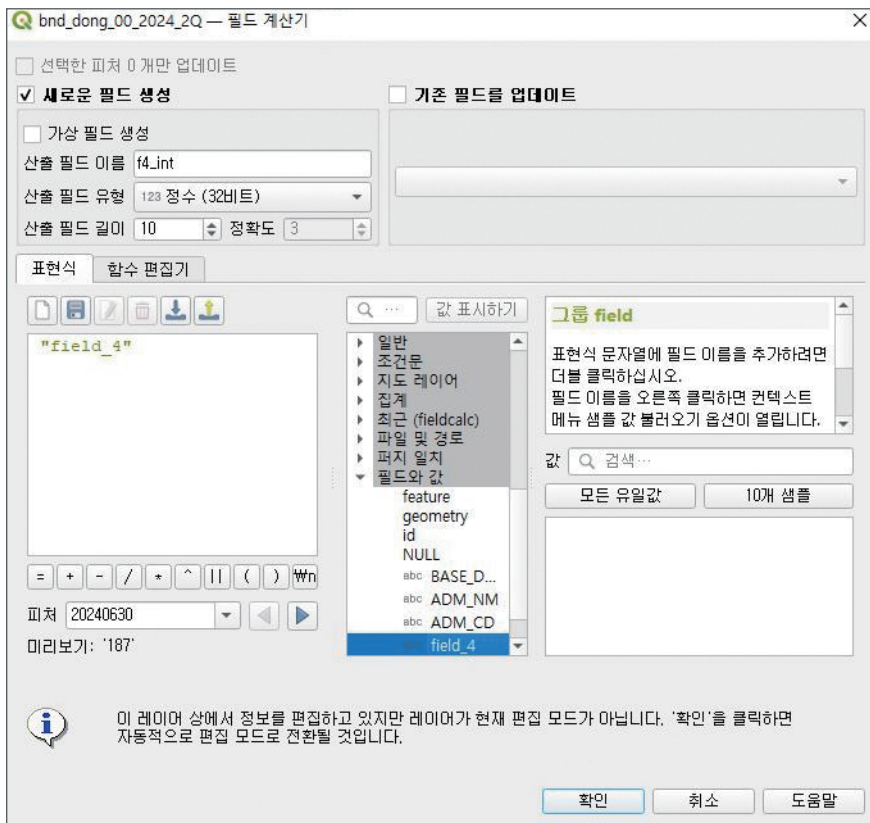
확인 취소 적용 도움말

5 단계구분도 표시하기

- ▶ 2022년_교육 서비스업(10차 중분류) 사업체수 통계값 컬럼이 문자형이기 때문에 정수형 컬럼을 추가하기 위하여 **속성** → **필드** 에서 필드계산기()를 선택합니다.



- ▶ 변경할 컬럼(field_4)을 **표현식** 에 작성하고, 추가할 정수형 컬럼명을 'f4_int' 등으로 저장합니다.



- ▶ 레이어 → 편집 모드 켜고 끄기 ()를 선택 해제하여 컬럼 추가 내용을 레이어에 저장합니다.
bnd_dong_00 레이어의 속성 → 심볼 에서 단계구분도 표시를 위한 설정값을 적절히 선택합니다.

심볼 종류 : 단계 구분

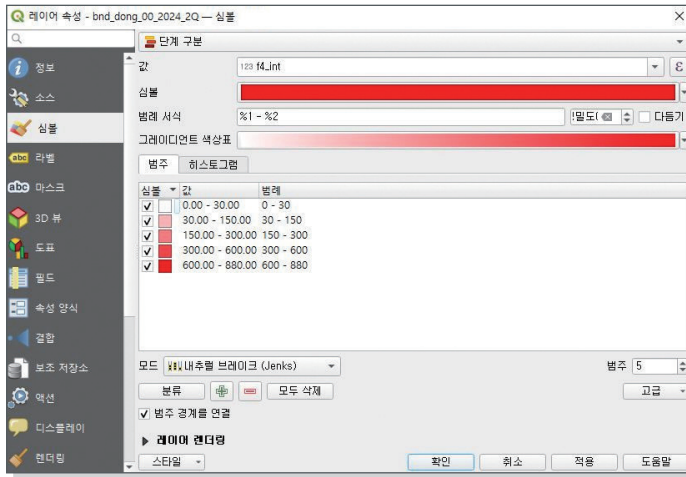
값 : f4_int

심볼 → 채우기 → 단순 채우기

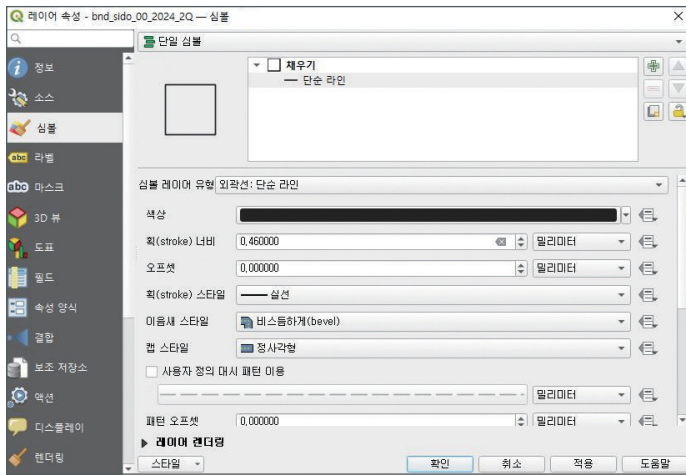
모드 : 내추럴 브레이크(Jenks)

범주 : 5

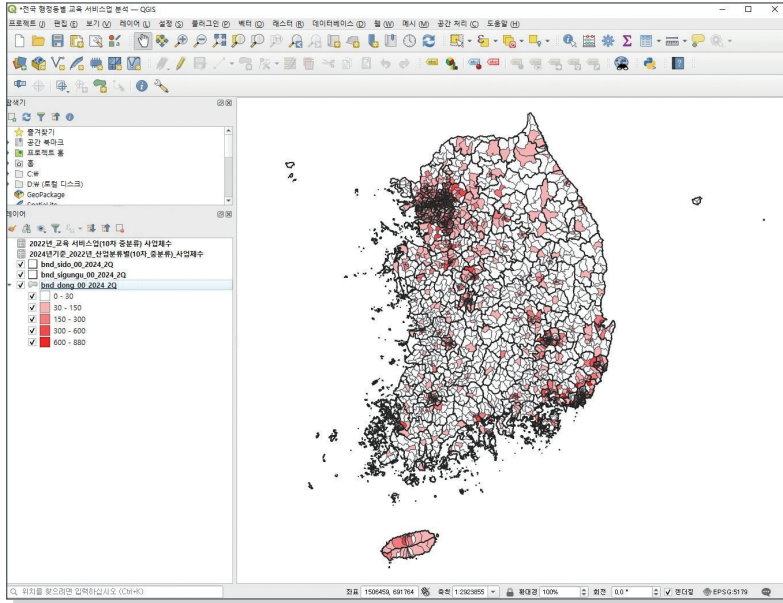
※ 범주 단계별 값의 범위는 심볼의 값을 더블클릭해서 각각 변경할 수 있습니다.



- ▶ 시각화 결과가 잘 보이도록 bnd_sido_00 , bnd_sigungu_00 을 단순 라인(선)으로 지정합니다.
경계 레이어의 속성 → 심볼 → 레이어 유형 → 외곽선: 단순라인

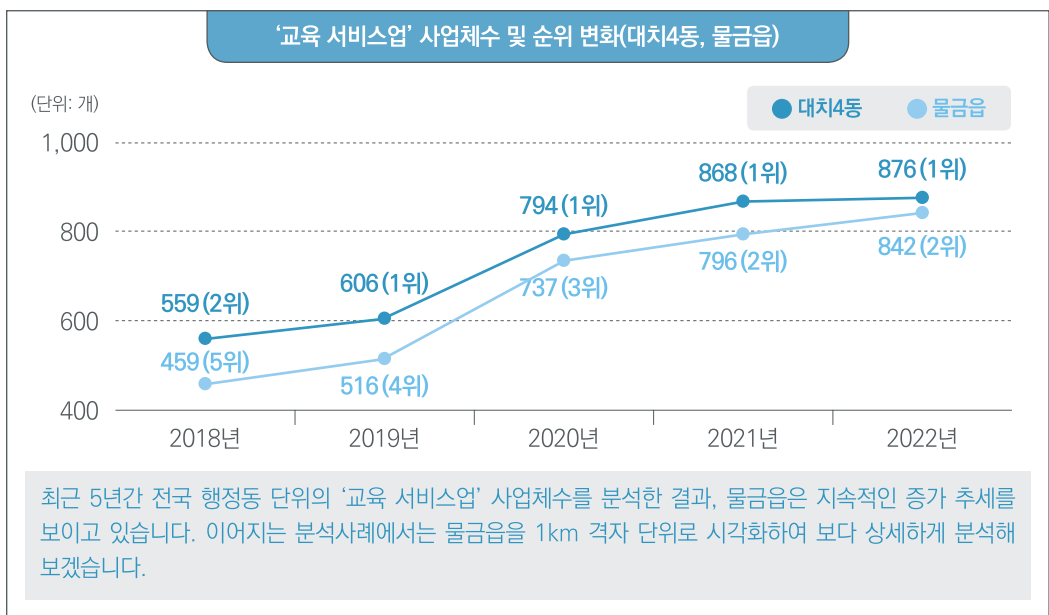


- ▶ 전국 행정동별 교육 서비스업 업종에 대한 사업체수를 시각화한 결과입니다.



| 분석 결과(요약) | 2022년 전국 행정동에서 ‘교육 서비스업’ 사업체가 가장 많은 행정동 Top2

순위	행정동	지역	교육 서비스업 사업체수
1위	대치4동	서울특별시 강남구	876개
2위	물금읍	경상남도 양산시	842개





PART

II

물금읍 교육 서비스업
격자 분석
(QGIS)



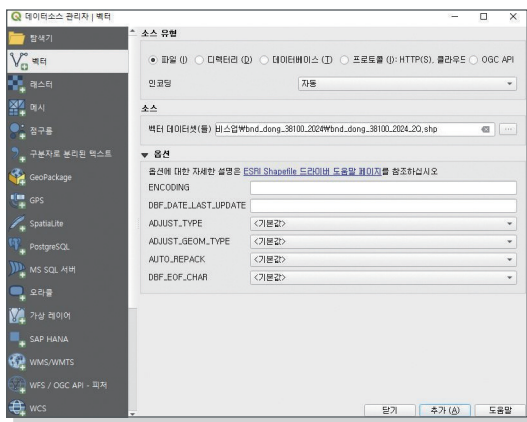


II 물금읍 교육 서비스업 격자 분석

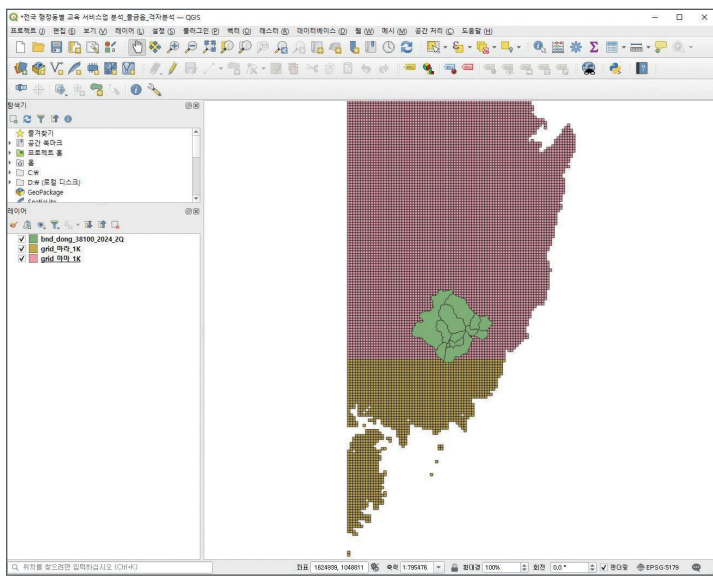
앞서 분석한 결과를 바탕으로 **bnd_dong_38100** 경계, **grid_마라_1K** 경계, **grid_마마_1K** 경계, **2022년_사업체중분류_마라_1K** 통계, **2022년_사업체중분류_마마_1K** 통계 를 활용하여 물금읍의 1km 격자 단위를 추가로 분석하는 과정을 단계별로 설명합니다.

1 경계 레이어 추가하기(.shp)

레이어 → 레이어 추가 → 벡터 레이어 추가 에서 양산시 행정동 **bnd_dong_38100** 경계, **grid_마라_1K** 경계, **grid_마마_1K** 경계 를 각각 추가합니다.



bnd_dong_38100 경계, **grid_마라_1K** 경계, **grid_마마_1K** 경계 를 추가한 화면입니다.



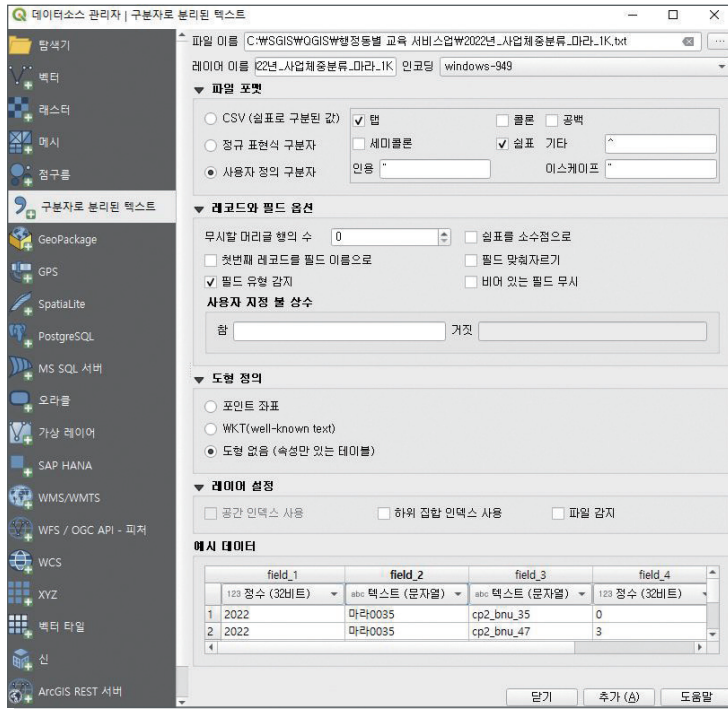
2 통계 레이어 추가하기(.txt)

레이어 → 레이어 추가 → 구분자로 분리된 텍스트 레이어 추가 에서 2022년_사업체중분류_마라_1K , 2022년_사업체중분류_마마_1K 통계를 각각 추가합니다.

※ 파일 포맷 → 사용자 정의 구분자 → 기타 항목에서 구분자를 '^'로 설정

※ CSV 파일을 다운 받은 경우, 파일 포맷 → CSV(쉼표로 구분된 값) 선택

※ 예시 데이터를 미리 보면서 'windows-949' 또는 'UTF-8' 등 한글이 깨지지 않는 인코딩 선택

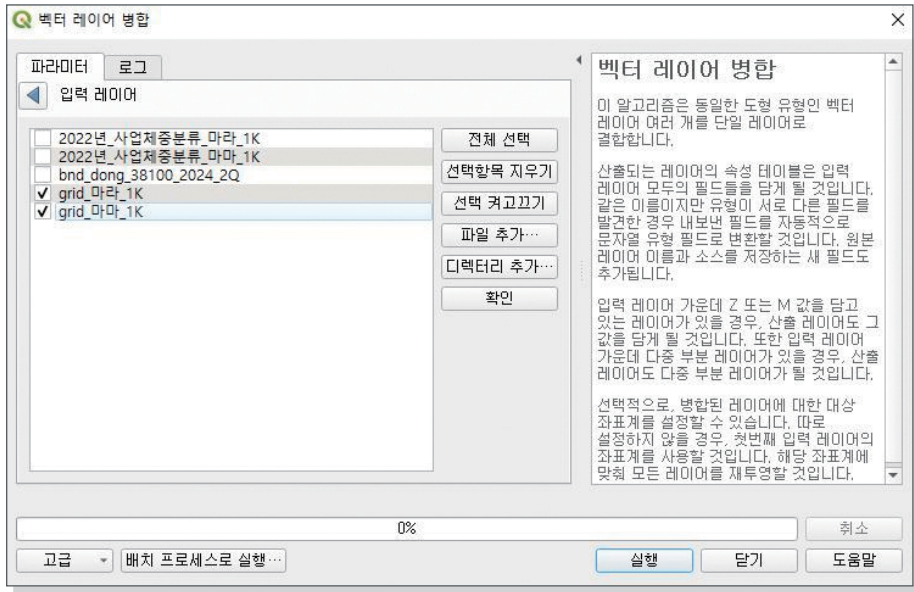


통계 레이어(2022년_사업체중분류_마라_1K , 2022년_사업체중분류_마마_1K)를 마우스 오른쪽 클릭 하여 속성 테이블 열기 로 사업체 통계를 확인합니다.

	field_1	field_2	field_3	field_4
1	2022	마라0035	cp2_bnu_35	0
2	2022	마라0035	cp2_bnu_47	3
3	2022	마라0035	cp2_bnu_55	8
4	2022	마라0035	cp2_bnu_56	3
5	2022	마라0035	cp2_bnu_75	3
6	2022	마라0035	cp2_bnu_94	3
7	2022	마라0036	cp2_bnu_94	0
8	2022	마라0037	cp2_bnu_47	5
9	2022	마라0037	cp2_bnu_55	8
10	2022	마라0037	cp2_bnu_56	5

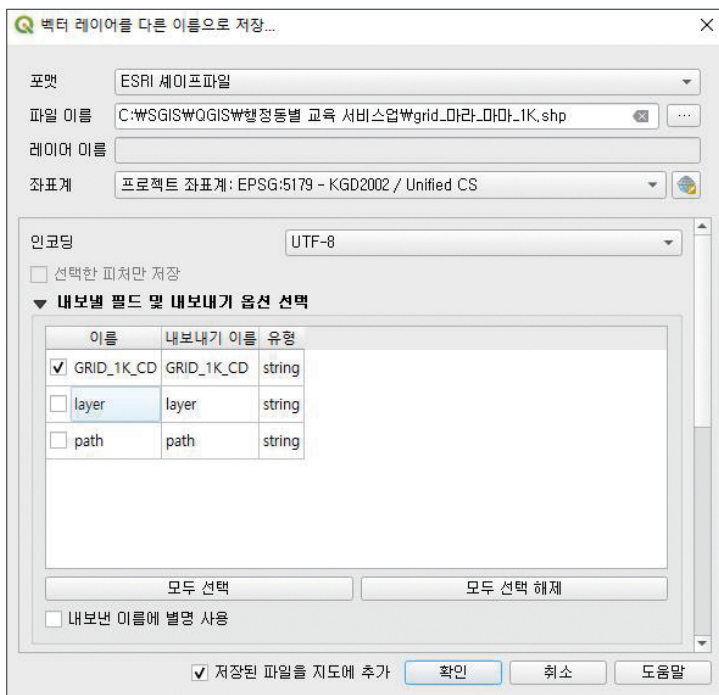
3 격자 경계 병합 및 행정동 경계 선택하여 저장하기

▶ 벡터 → 데이터 관리 도구 → 벡터 레이어 병합 에서 grid_마라_1K 와 grid_마마_1K 를 병합합니다.

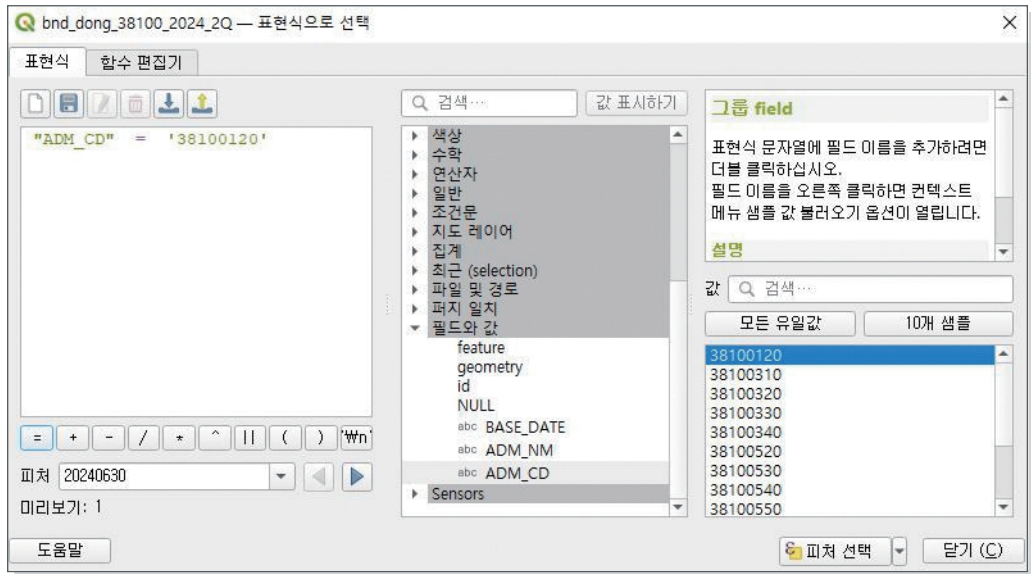


▶ 병합한 산출물 레이어를 마우스 오른쪽 클릭해서 내보내기 → 피처를 다른 이름으로 저장 합니다.
(grid_마라_마마_1K.shp)

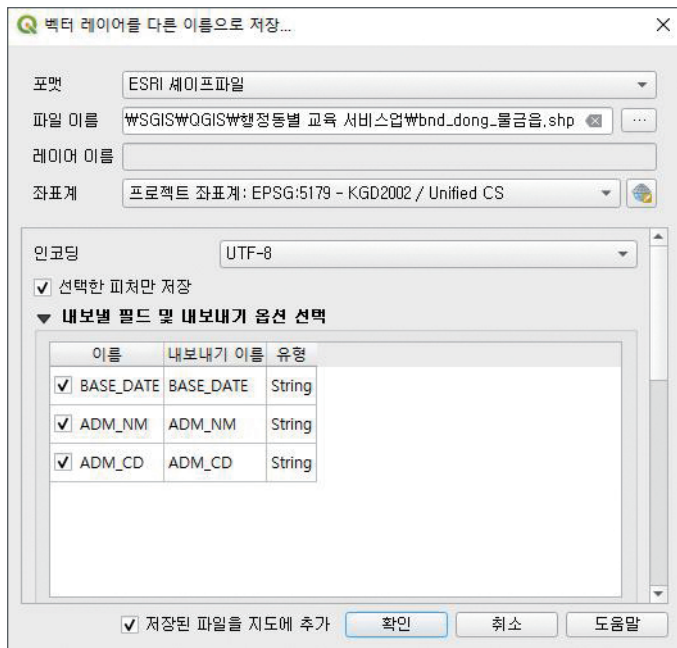
※ 병합한 레이어에서 필요한 정보(GRID_1K_CD)만 저장합니다.



- ▶ **bnd_dong_38100** 행정동 경계에서 **표현식을 이용해서 피쳐 선택** () 화면을 통해 물금읍(38100120)에 해당하는 경계만 선택합니다.



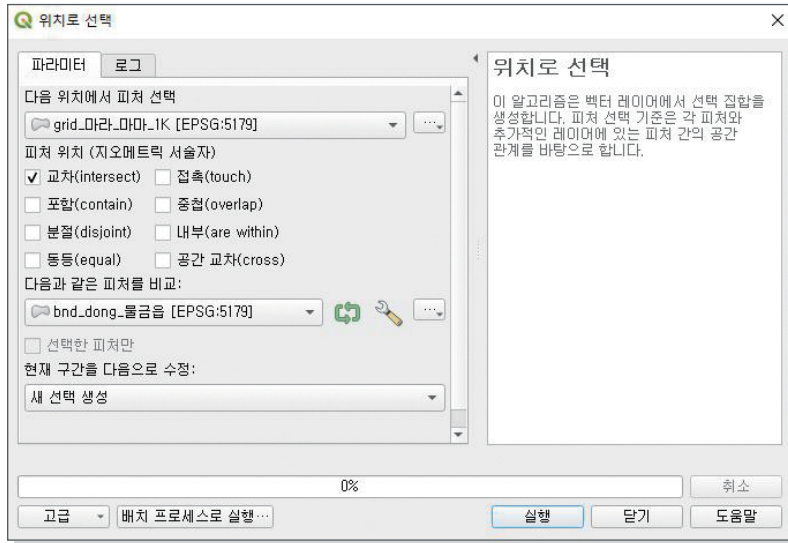
- ▶ **bnd_dong_38100** 레이어를 마우스 오른쪽 클릭해서 **내보내기** ➔ **선택한 피쳐를 다른 이름으로 저장** 합니다.(**bnd_dong_물금읍.shp**)



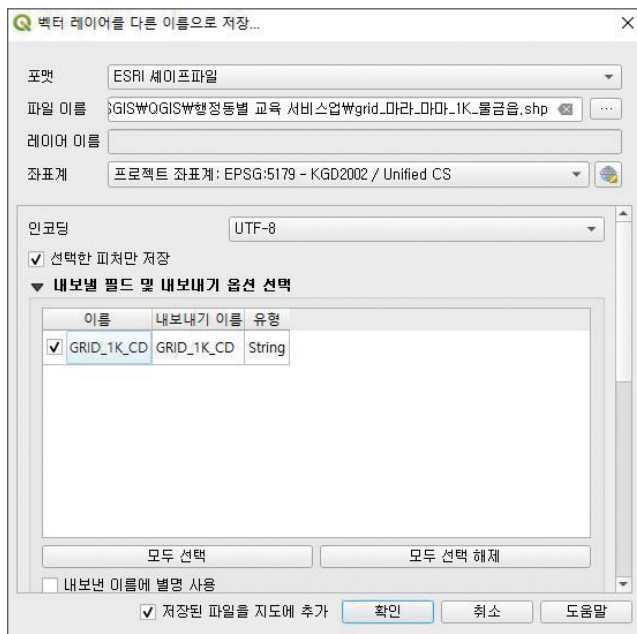
4 격자 경계에서 물금읍과 겹치는 부분 선택하기

▶ **grid_마라_마마_1K** 레이어를 선택하고 **벡터** → **조사 도구** → **위치로 선택** 화면에서 교차(intersect), 물금읍 경계를 각각 선택하여 실행합니다.

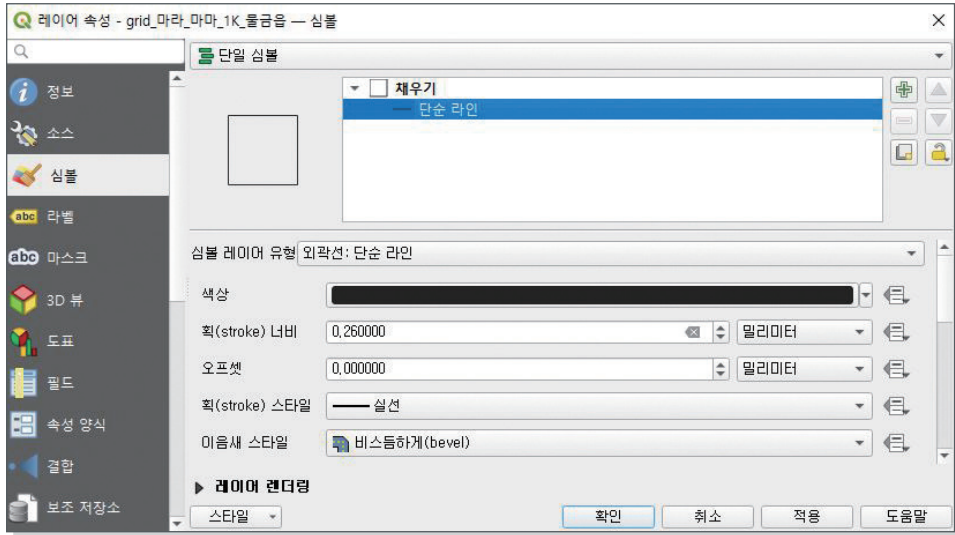
※ **grid_마라_마마_1K** 레이어에서 선택된 격자는 노란색으로 표시됩니다.



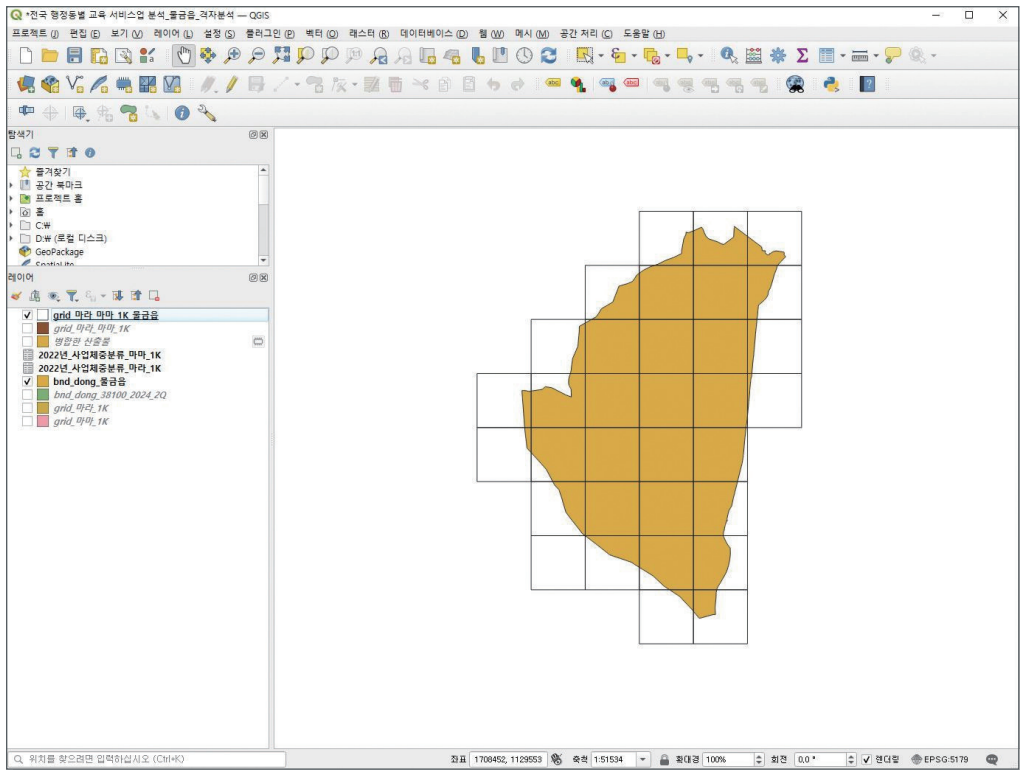
▶ **grid_마라_마마_1K** 레이어를 마우스 오른쪽 클릭해서 **내보내기** → **선택한 피처를 다른 이름으로 저장** 합니다. (**grid_마라_마마_1K_물금읍.shp**)



- ▶ **grid_마라_마마_1K_물금읍** 레이어를 마우스 오른쪽 클릭해서 **속성** ➔ **심볼** 에서 **채우기** 의 **심볼 레이어 유형** 을 **외곽선 : 단순 라인** 으로 설정합니다.

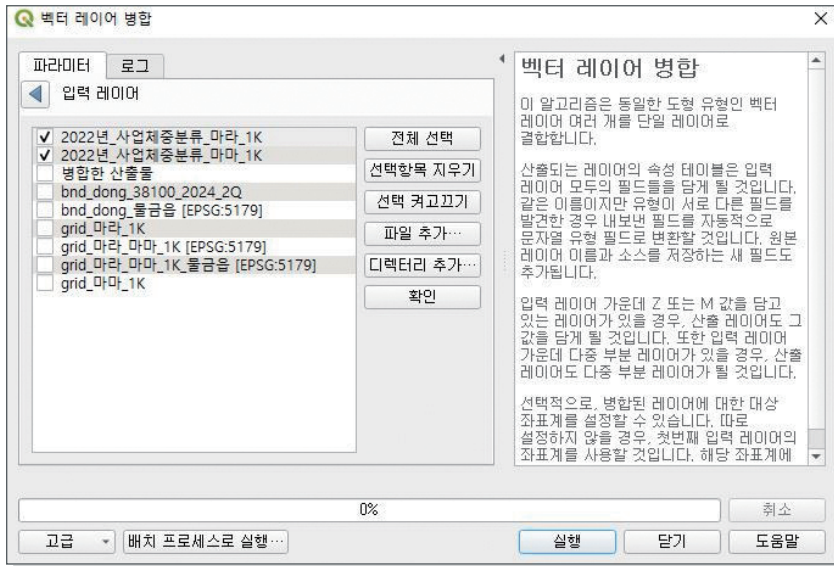


- ▶ 물금읍과 겹치는 1km 격자가 선택된 것을 확인할 수 있습니다.



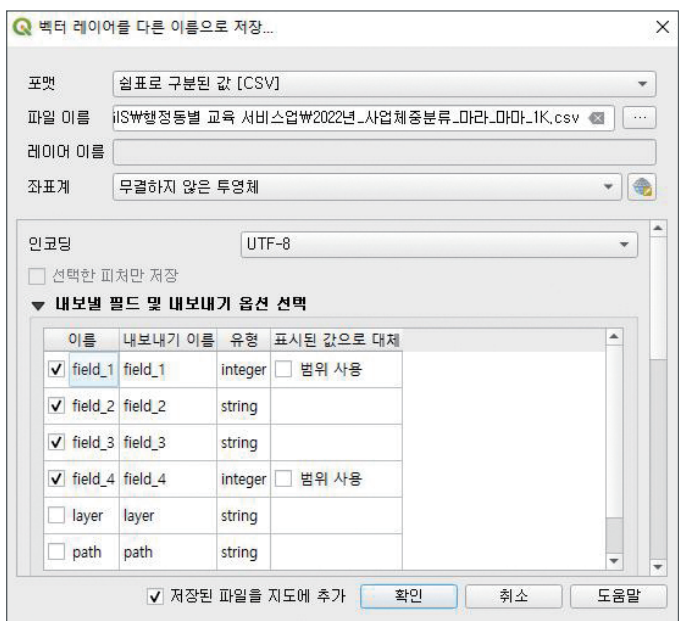
5 통계 레이어 병합 및 해당 통계항목 선택하기

- ▶ 벡터 → 데이터 관리 도구 → 벡터 레이어 병합 에서 2022년_사업체중분류_마라_1K 와 2022년_사업체중분류_마마_1K 를 병합합니다.

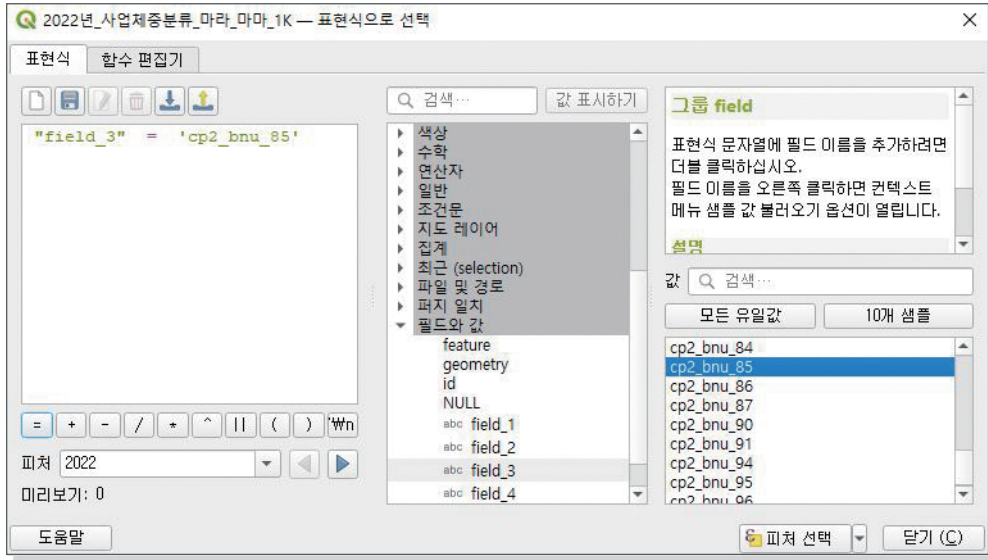


- ▶ 병합한 산출물 레이어를 마우스 오른쪽 클릭해서 **내보내기** → **피처를 다른 이름으로 저장** 합니다.
(2022년_사업체중분류_마라_마마_1K.csv)

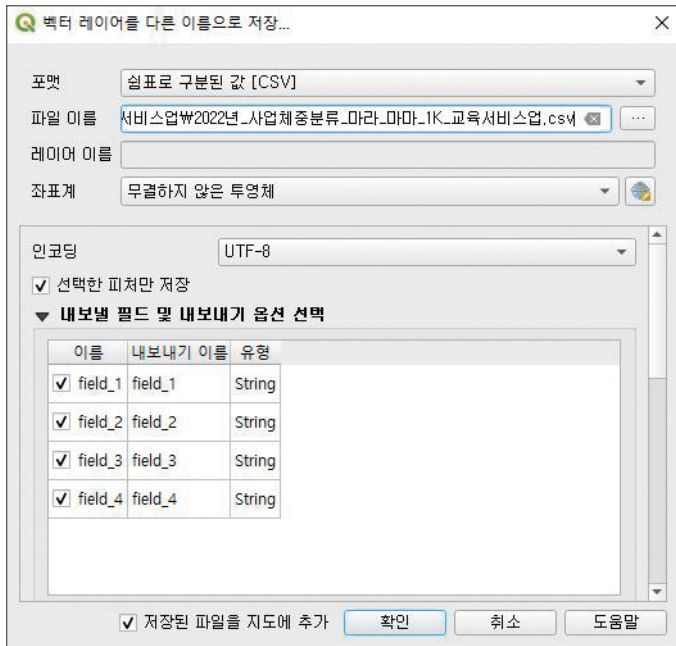
※ 병합한 레이어에서 필요한 정보(기준년도(field_1), 격자코드(field_2), 통계항목(field_3), 통계값(field_4))만 저장합니다.



- ▶ 2022년_사업체중분류_마라_마마_1K 레이어를 선택하고 편집 ➔ 선택 ➔ 표현식으로 피쳐 선택
(화면에서 세번째 항목(field_3)의 값을 'cp2_bnu_85'(교육 서비스업)로 선택합니다.

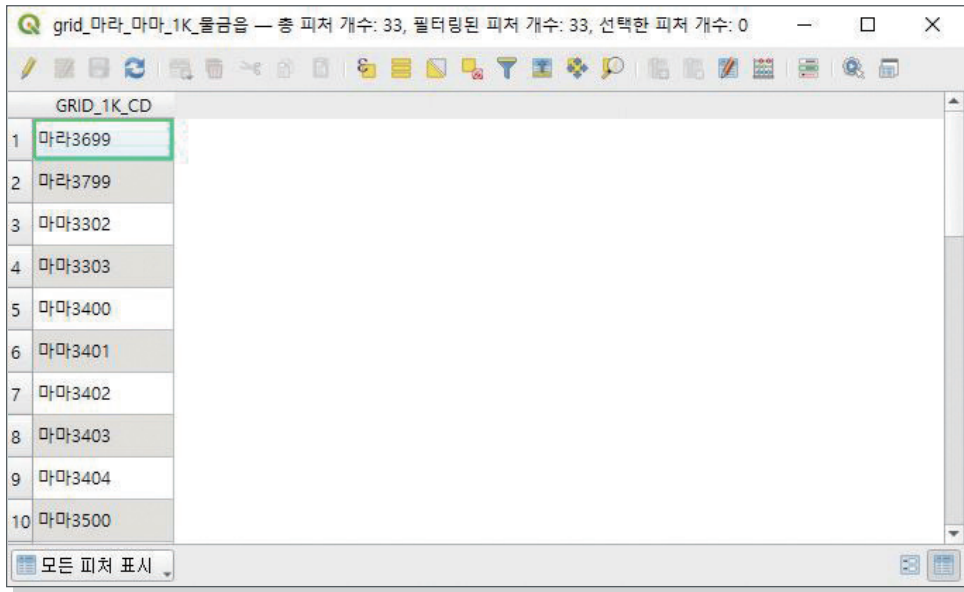


- ▶ 2022년_사업체중분류_마라_마마_1K 레이어를 마우스 오른쪽 클릭해서 내보내기 ➔ 선택한 피쳐를
다른 이름으로 저장 합니다.(2022년_사업체중분류_마라_마마_1K_교육서비스업.csv)

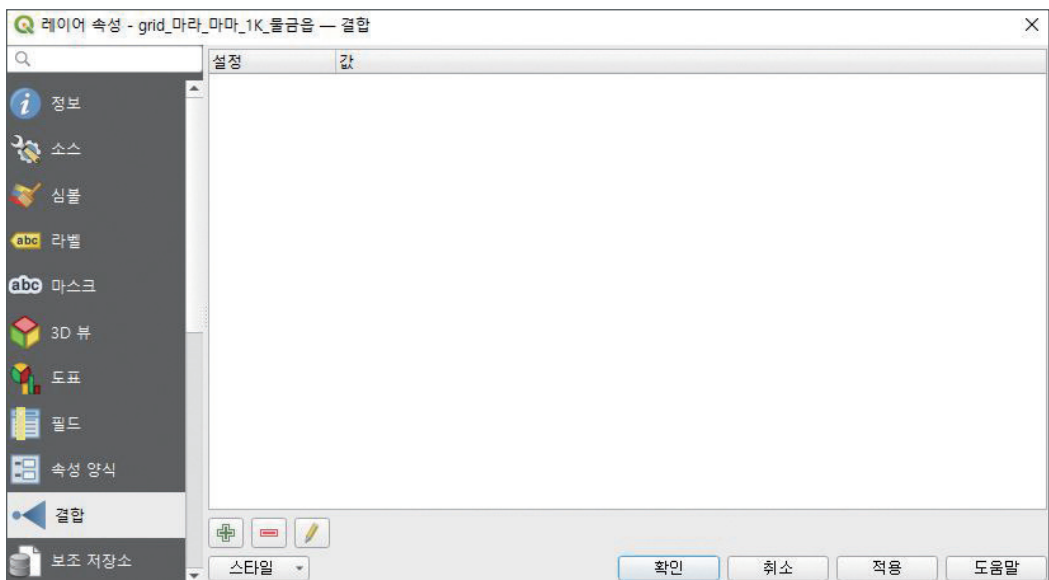


6 물금읍 격자와 교육 서비스업 통계 결합하기

- ▶ **grid_마라_마마_1K_물금읍** 레이어를 마우스 오른쪽 클릭해서 **속성 테이블 열기** 를 선택합니다. 격자경계 레이어에는 격자 코드(GRID_1K_CD)만 있습니다.



- ▶ **grid_마라_마마_1K_물금읍** 레이어를 마우스 오른쪽 클릭 후 **속성** 을 선택해서 레이어 속성 화면을 열고 왼쪽 메뉴에서 **결합** , 화면 하단의 **+** 버튼을 순서대로 선택합니다.



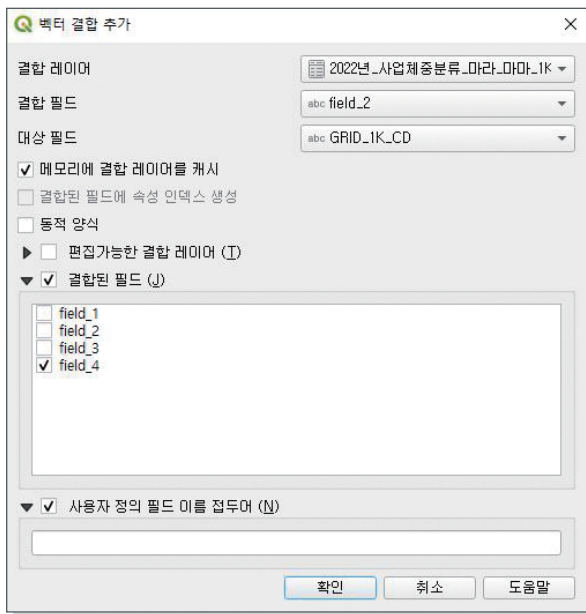
- ▶ 결합 레이어, 결합 필드, 대상 필드, 결합된 필드 등을 적절히 선택합니다.

결합 레이어 : 2022년_사업체중분류_마라_마마_1K_교육서비스업

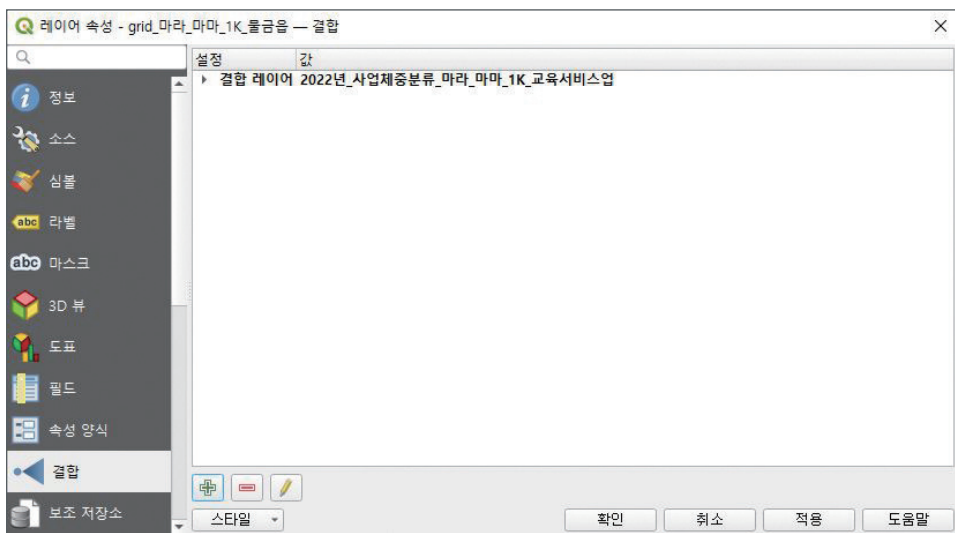
결합 필드 : field_2

대상 필드 : GRID_1K_CD

※ **결합된 필드** 에서 통계값 필드인 'field_4'는 필수 선택이며, **접두어** 는 생략 가능



- ▶ 레이어 속성 화면으로 돌아와서 적용, 확인을 선택합니다.

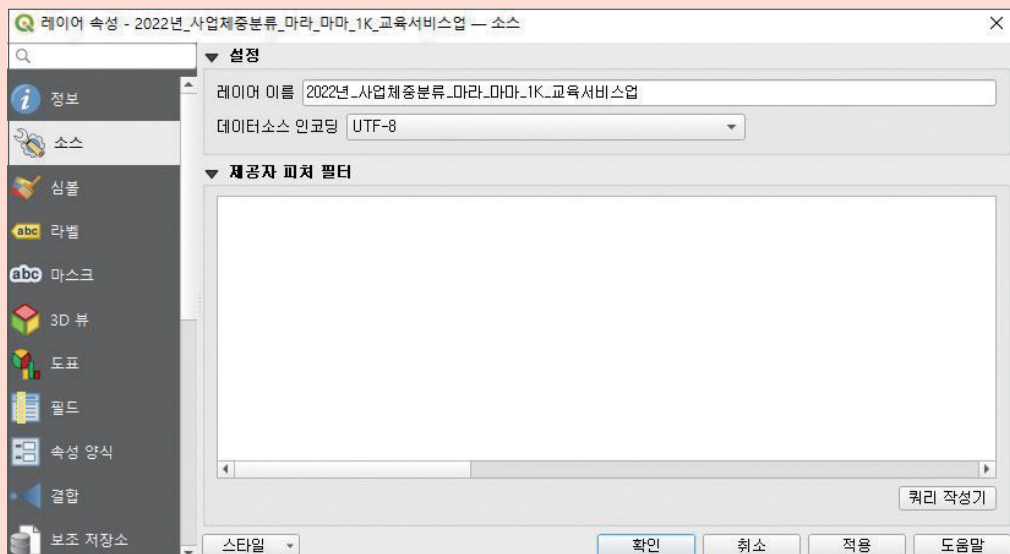


- ▶ **grid_마라_마마_1K_물금읍** 레이어를 마우스 오른쪽 클릭해서 **속성 테이블 열기** 를 다시 선택합니다.
통계값 필드가 결합되어 있습니다.

	GRID_1K_CD	field_4
1	마라3699	NULL
2	마라3799	NULL
3	마마3302	NULL
4	마마3303	NULL
5	마마3400	NULL
6	마마3401	NULL
7	마마3402	5
8	마마3403	NULL
9	마마3404	NULL

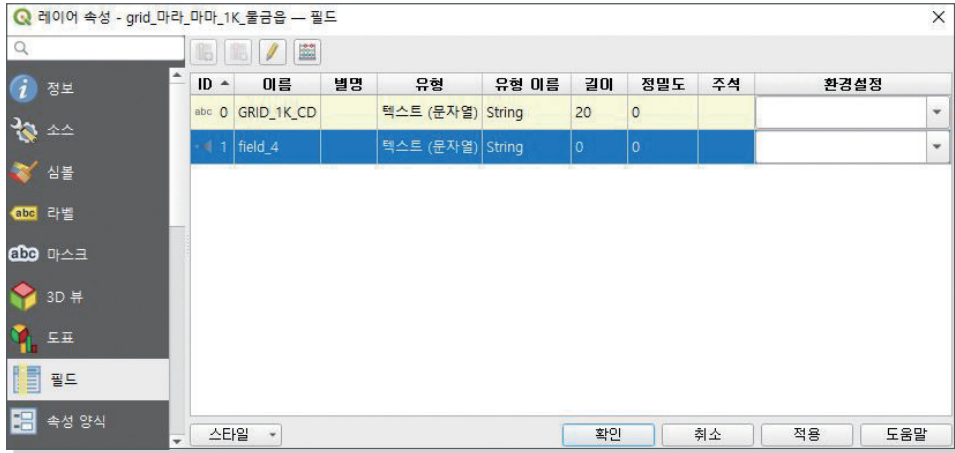
주의

결합이 안 될 경우, **grid_마라_마마_1K_물금읍** 레이어의 **속성** → **소스** 에서 인코딩을 'UTF-8'로 적용하고 6단계부터 다시 실시합니다.

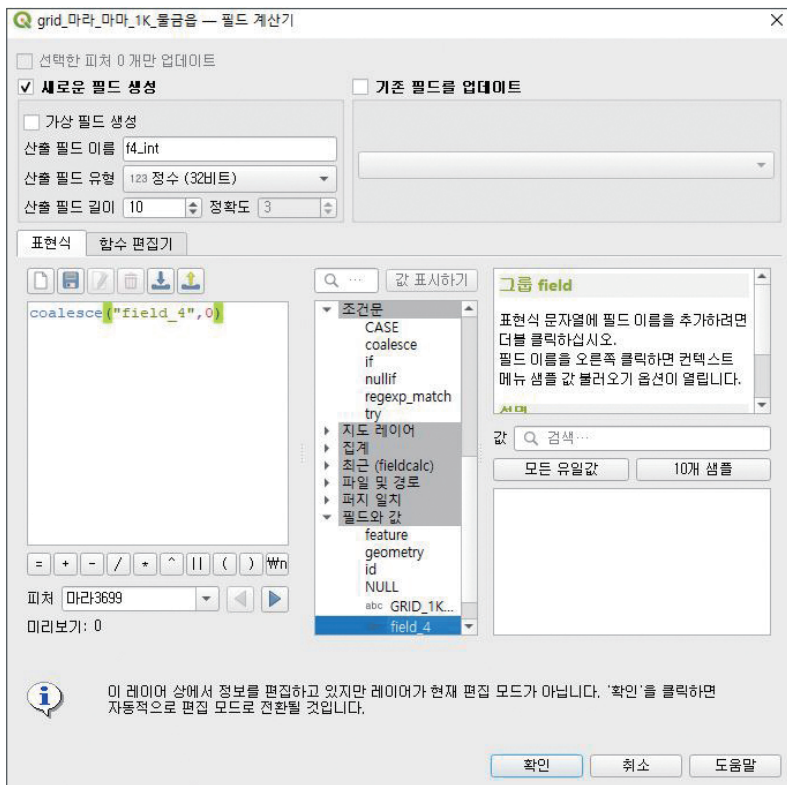


7 단계구분도 표시하기

- ▶ **grid_마라_마마_1K_물금읍** 통계값 컬럼(field_4)이 문자형이기 때문에 정수형 컬럼을 추가합니다.
속성 → 필드 에서 필드계산기()를 선택합니다.



- ▶ 변경할 컬럼(field_4)을 표현식에 작성하고, 추가할 정수형 컬럼명을 'f4_int' 등으로 저장합니다.
※ 「SGIS 자료제공」의 격자 통계 특성 상, 인구 또는 사업체가 없는 격자에 대해서는 통계 데이터를 생성하지 않으므로 결측값을 '0'으로 채울 수 있습니다.



- ▶ 레이어 → 편집 모드 켜고 끄기 ()를 선택 해제하여 컬럼 추가 내용을 레이어에 저장합니다.

grid_마라_마마_1K_물금읍 레이어의 속성 → 심볼 에서 단계구분도 표시를 위한 설정값을 적절히 선택합니다.

심볼 종류 : 단계 구분

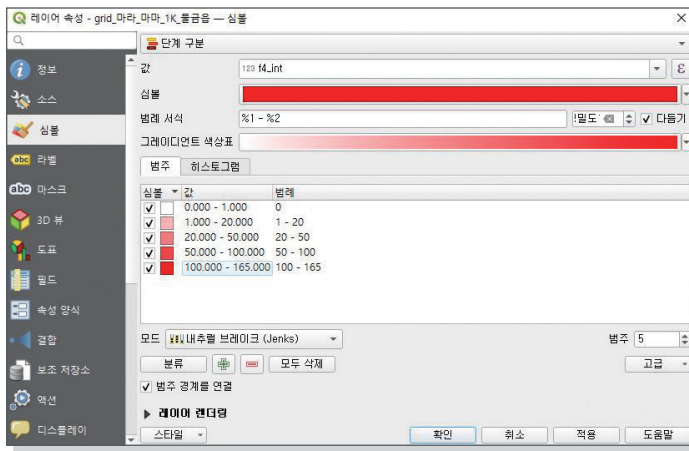
값 : f4_int

심볼 → 채우기 → 단순 채우기

모드 : 내추럴 브레이크(Jenks)

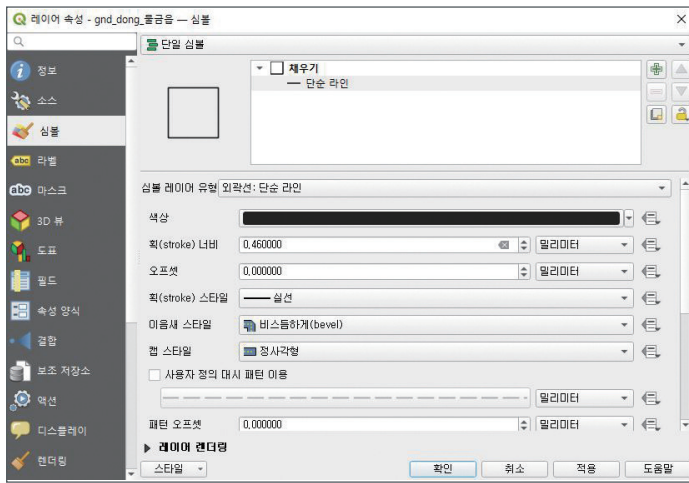
범주 : 5

※ 범주 단계별 값의 범위는 심볼의 값을 더블클릭해서 각각 변경할 수 있습니다.

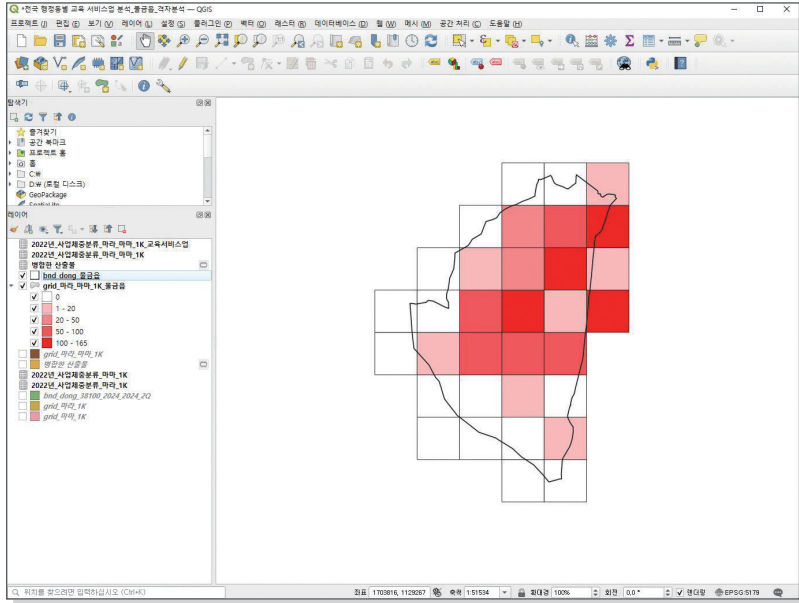


- ▶ 시각화 결과가 잘 보이도록 bnd_dong_물금읍 을 단순 라인(선)으로 지정합니다.

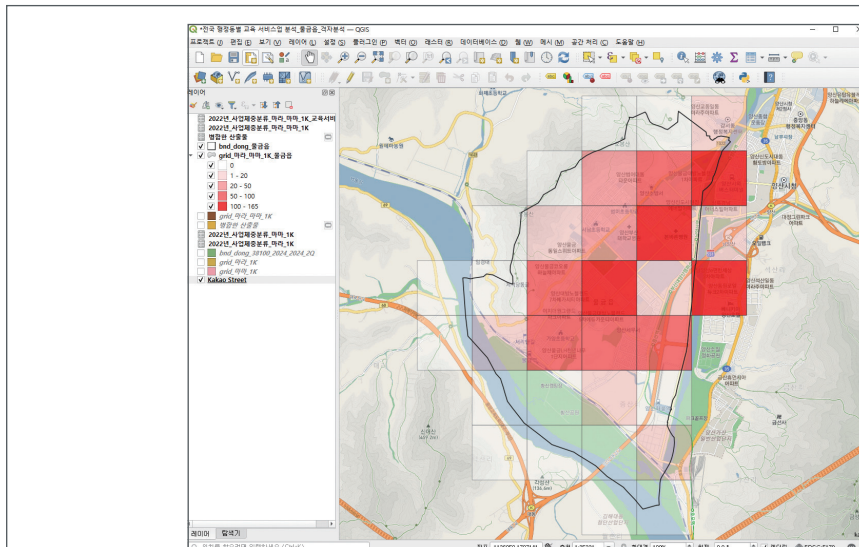
경계 레이어의 속성 → 심볼 → 레이어 유형 → 외곽선 : 단순라인



- ▶ 물금읍과 교차하는 1km 격자를 기준으로 교육 서비스업 업종에 대한 사업체수를 시각화한 결과입니다.



| 분석 결과(요약) | 물금읍 중심부에 ‘교육 서비스업’ 사업체가 집중적으로 분포하는 경향



경남 양산시에 위치한 물금읍은 부산 인접 지역으로 교육 서비스업 수요가 많은 지역이며, 신도시 개발로 인한 인구 증가에 따라 학원 및 교육 기관이 증가하고 있습니다. 물금읍 중심부에 아파트 단지가 집중되어 있으며, 대학의 지역 캠퍼스와 다수의 초·중·고등학교가 분포하는 것을 지도에서 확인할 수 있습니다.



PART

III

서울시 행정동
인구수 현황 분석
(Python)





III | 서울시 행정동 인구수 현황 분석

SGIS 자료제공 메뉴에서 내려받은 서울시 행정동 경계(`bnd_dong_11_2024_2Q.shp`), 2023년 서울시 인구 통계(`11_2023년_인구총괄(총인구).txt`)를 파이썬(Jupyter Notebook)으로 분석하는 과정을 설명합니다.

1 분석 라이브러리 로드 및 사용 준비하기

- ▶ geopandas, pandas, matplotlib 라이브러리를 임포트합니다. 각각 공간 데이터 분석, 표 형식 데이터 처리, 시각화에 사용됩니다.

```
[1]: import geopandas as gpd # 지리정보 파일 읽고 처리
import pandas as pd # 표 형식 데이터 처리
import matplotlib.pyplot as plt # 그래프, 지도 등 시각화
```

```
plt.rcParams['font.family'] = 'Malgun Gothic' # 윈도우에서 한글 폰트 설정
```

그래프, 지도 등 시각화에 사용되는 matplotlib는 사용하기 전 한글 폰트를 명시적으로 설정해야 합니다. 한글이 사용되는 경우, 폰트 파일 로딩 지연 문제 등으로 소스코드 첫부분에서 실행해줍니다.

주의

지리정보를 처리하는데 사용되는 geopandas 라이브러리는 별도로 설치해주어야 합니다. 윈도우 앱 목록에서 **Anaconda** → **Anaconda Prompt** 를 선택하고 아래의 명령어로 geopandas를 설치합니다.

```
Anaconda Prompt
(base) C:\> pip install geopandas
```

2 서울시 경계 및 통계 데이터 불러오기

- ▶ gpd.read_file()로 경계 데이터를 불러오고, pd.read_csv()로 인구통계 데이터를 불러옵니다. 두 파일을 각각 `bord_sido`와 `stat`에 저장합니다. (※ 경로는 사용자 PC에 맞게 수정)

```
[2]: prj_dir = 'C:/SGIS/Python/서울시 행정동 인구 분석'
bord_sido = gpd.read_file(prj_dir + '/' + 'bnd_dong_11_2024_2Q.shp')
stat = pd.read_csv(prj_dir + '/' + '11_2023년_인구총괄(총인구).txt', sep='^',
names=['BASE_YEAR', 'ADM_CD', 'STAT_CD', 'POP'])
```

※ 데이터를 로드하기 전, 다운받은 통계 파일을 열어보면 구분자는 '^'(Caret)이고, 첫 행에 컬럼명이 없는 것을 확인할 수 있습니다. 이 경우 파일을 불러오면서 컬럼명을 함께 지정할 수 있습니다.

3 데이터 미리보기

- ▶ head() 함수로 두 데이터의 첫 5개 행을 출력하여 구조를 확인합니다.

```
[3]: print(bord_sido.head()) # 기준날짜(BASE_DATE), 행정구역 이름(ADM_NM),  
# 행정구역 코드(ADM_CD), 지리정보(geometry)로 구성
```

	BASE_DATE	ADM_NM	ADM_CD	\
0	20240630	잠실4동	11240750	
1	20240630	잠실7동	11240780	
2	20240630	잠실2동	11240790	
3	20240630	잠실6동	11240770	
4	20240630	방이2동	11240580	

	geometry
0	POLYGON ((966144.947 1947247.39, 966140.973 19...
1	POLYGON ((962306.123 1944876.429, 962298.368 1...
2	POLYGON ((963373.434 1945897.333, 963372.972 1...
3	POLYGON ((964764.987 1947265.422, 964790.647 1...
4	POLYGON ((965792.751 1946532.439, 965878.771 1...

```
[4]: print(stat.head()) # 기준년도(BASE_YEAR), 행정구역 코드(ADM_CD),  
# 통계 코드(STAT_CD), 인구수(POP)
```

	BASE_YEAR	ADM_CD	STAT_CD	POP
0	2023	11010530	to_in_001	8377
1	2023	11010540	to_in_001	2115
2	2023	11010550	to_in_001	9136
3	2023	11010560	to_in_001	16475
4	2023	11010570	to_in_001	7557

4 통계 항목 필터링

- ▶ STAT_CD 컬럼에 있는 고유값을 확인하고 분석에 필요한 'to_in_001(총인구수)'만 필터링하여 stat_total에 저장합니다.

```
[5]: # STAT_CD 열에 있는 고유 값 확인  
unique_stat_cd = stat['STAT_CD'].unique()  
print(unique_stat_cd)
```

```
['to_in_001' 'to_in_007' 'to_in_008']
```

```
[6]: # 총인구 데이터만 필터링 ('to_in_001')  
stat_total = stat[stat['STAT_CD'] == 'to_in_001']  
# 필터링 후 STAT_CD 고유값 재확인  
unique_stat_cd = stat_total['STAT_CD'].unique()  
print(unique_stat_cd)
```

```
['to_in_001']
```

5 분석에 필요한 컬럼만 선택하기

- ▶ stat_total에서 ADM_CD와 POP 컬럼만 남겨서 나머지 불필요한 열을 제거합니다.(추후 경계 파일과 병합 과정에서 필요한 컬럼만 저장)

```
[7]: stat_total = stat_total[['ADM_CD', 'POP']]
```

6 경계와 통계 데이터 타입 확인 및 변환하기

- ▶ 각각의 파일에 키값으로 사용가능한 행정구역 코드(ADM_CD) 데이터 타입 확인 후 문자열로 일치 시킵니다.(병합 시 발생하는 오류 방지)

```
[8]: print(bord_sido.dtypes) # 경계(bord_sido) 각 열의 데이터 타입 확인
```

```
BASE_DATE      object
ADM_NM         object
ADM_CD         object
geometry       geometry
dtype: object
```

```
[9]: print(stat_total.dtypes) # 통계(stat_total) 각 열의 데이터 타입 확인
```

```
ADM_CD      int64
POP         int64
dtype: object
```

```
[10]: stat_total['ADM_CD'] = stat_total['ADM_CD'].astype(str) # 문자열로 변환
```

7 경계와 통계 병합하기

- ▶ bord_sido와 stat_total 두 개의 데이터프레임을 'ADM_CD'를 기준으로 left join(왼쪽 조인) 하여 병합합니다.

```
[11]: bord_sido = bord_sido.merge(stat_total, on='ADM_CD', how='left')
print(bord_sido.isna().sum()) # 결측값이 있는지 확인
print(bord_sido.head()) # 병합 데이터 미리보기
```

```
BASE_DATE    0
ADM_NM       0
ADM_CD       0
geometry     0
POP          0
dtype: int64
```

	BASE_DATE	ADM_NM	ADM_CD	\	geometry	POP
0	20240630	잠실4동	11240750			19916
1	20240630	잠실7동	11240780			8581
2	20240630	잠실2동	11240790			32990
3	20240630	잠실6동	11240770			15381
4	20240630	방이2동	11240580			25888

8 총인구수를 기준으로 구간 나누기

- ▶ 총인구수('POP')를 5개의 구간으로 나누고 각 구간에 대한 라벨을 지정합니다.

```
[12]: print(bord_sido['POP'].min(), bord_sido['POP'].max()) # 최소값과 최대값 확인하기
```

26 52789

```
[13]: bins = [25, 10000, 20000, 30000, 40000, 53000] # 적절한 구간 지정
labels = ['25~10,000', '10,000~20,000', '20,000~30,000', '3,0000~40,000', '40,000~53,000'] # 구간의 라벨
bord_sido['POP_BINS'] = pd.cut(bord_sido['POP'], bins=bins, labels=labels, right=False)
print(bord_sido.head()) # 인구 구간('POP_BINS') 컬럼 추가 확인
```

	BASE_DATE	ADM_NM	ADM_CD	\	geometry	POP	POP_BINS
0	20240630	잠실4동	11240750			19916	10,000~20,000
1	20240630	잠실7동	11240780			8581	25~10,000
2	20240630	잠실2동	11240790			32990	3,0000~40,000
3	20240630	잠실6동	11240770			15381	10,000~20,000
4	20240630	방이2동	11240580			25888	20,000~30,000

9 지도 시각화 하기

```
[14]: # 시각화 설정(그림과 축 설정)
fig, ax = plt.subplots(1, 1, figsize=(12, 12))

# 지도 시각화('POP_BINS' 컬럼을 기준으로 OrRd 색상 맵 사용)
bord_sido.plot(column='POP_BINS', ax=ax, legend=True, cmap='OrRd')

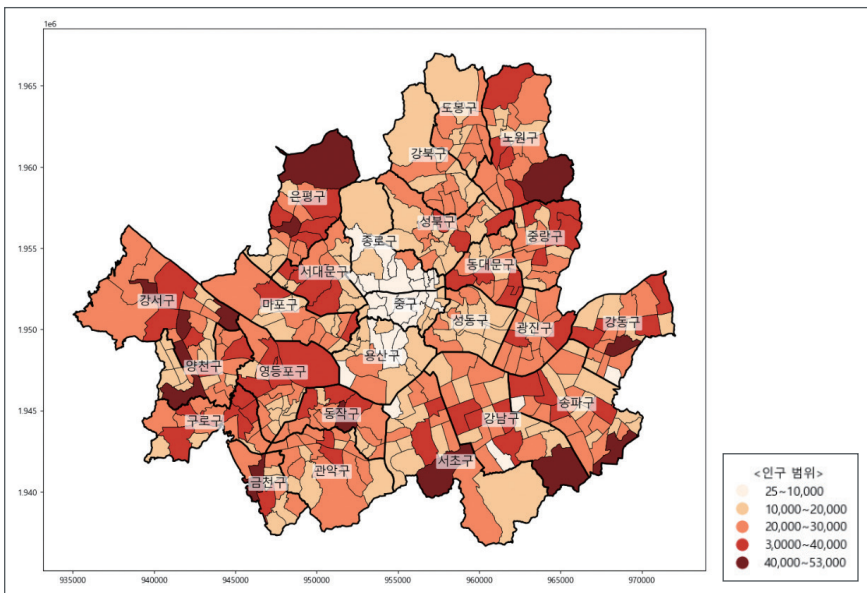
# 서울시 행정동 경계 추가 (경계선만 보이도록 설정, 배경은 투명)
bord_sido.plot(ax=ax, color='none', edgecolor='black', linewidth=0.5)

# (추가) 서울시 시군구 경계를 추가하여 명확하게 분석하기
bord_sgg = gpd.read_file(prj_dir + '/' + 'bnd_sigungu_11_2024_2Q.shp')
bord_sgg.plot(ax=ax, color='none', edgecolor='black', linewidth=2)
bord_sgg['centroid'] = bord_sgg.geometry.centroid # 중심점 계산
for idx, row in bord_sgg.iterrows():
    ax.text(row['centroid'].x, row['centroid'].y, row['SIGUNGU_NM'], fontsize=15, ha='center',
            color='black', bbox=dict(facecolor='white', alpha=0.7, edgecolor='none',
            boxstyle='round,pad=0.1'))

# 범례 위치 및 제목 설정
ax.get_legend().set_bbox_to_anchor((0.5, -0.05)) # 범례 위치 조정
ax.get_legend().set_title("<인구 범위>") # 범례 제목 설정
ax.set_title('<서울시 행정동 인구 분석>', fontsize=24) # 지도 제목 설정

# 레이아웃 조정 및 출력
plt.tight_layout() # 그래프 레이아웃이 겹치지 않도록 자동 조정
plt.savefig(prj_dir + '/' + '서울시 행정동 인구 분석 지도.png')
plt.show() # 시각화 결과 화면에 출력
```

| 서울시 행정동 인구 분석 |





PART

IV

부산시 인구수 현황
격자 분석
(Python)





IV | 부산시 인구수 현황 격자 분석

SGIS 자료제공 메뉴에서 내려받은 부산시 경계(`bnd_sido_21_2024_2Q.shp`), 격자 경계(`grid_마라_1K.shp` , `grid_마마_1K.shp`), 2023년 격자 통계(`2023년_인구_마라_1K.txt` , `2023년_인구_마마_1K.txt`)를 파이썬(Jupyter Notebook)으로 분석하는 과정을 단계별로 설명합니다.

1 분석 라이브러리 로드 및 사용 준비하기

- ▶ geopandas, pandas, matplotlib 라이브러리를 임포트합니다. 각각 공간 데이터 분석, 표 형식 데이터 처리, 시각화에 사용됩니다.

```
[1]: import geopandas as gpd # 지리정보 파일 읽고 처리
import pandas as pd # 표 형식 데이터 처리
import matplotlib.pyplot as plt # 그래프, 지도 등 시각화
```

▶ `plt.rcParams['font.family'] = 'Malgun Gothic'` # 윈도우에서 한글 폰트 설정

그래프, 지도 등 시각화에 사용되는 matplotlib는 사용하기 전 한글 폰트를 명시적으로 설정해야 합니다. 한글이 사용되는 경우, 폰트 파일 로딩 지연 문제 등으로 소스코드 첫부분에서 실행해줍니다.

2 부산시 경계, 격자 경계 및 통계 데이터 불러오기

- ▶ `gpd.read_file()`로 경계 데이터를 불러오고, `pd.read_csv()`로 인구 통계 데이터를 불러옵니다.
`bord_sido( bnd_sido_21_2024_2Q.shp ), grid_mara( grid_마라_1K.shp ), grid_mama( grid_마마_1K.shp ), stat_mara( 2023년_인구_마라_1K.txt ), stat_mama( 2023년_인구_마마_1K.txt )`로 각각 저장합니다. (※ 경로는 사용자 PC에 맞게 수정)

```
[2]: prj_dir = 'C:/SGIS/Python/부산시 격자 인구 분석'
bord_sido = gpd.read_file(prj_dir + '/' + 'bnd_sido_21_2024_2Q.shp')
grid_mara = gpd.read_file(prj_dir + '/' + 'grid_마라_1K.shp')
grid_mama = gpd.read_file(prj_dir + '/' + 'grid_마마_1K.shp')
stat_mara = pd.read_csv(prj_dir + '/' + '2023년_인구_마라_1K.txt', sep='^',
                        names=['BASE_YEAR', 'GRID_1K_CD', 'STAT_CD', 'POP'])
stat_mama = pd.read_csv(prj_dir + '/' + '2023년_인구_마마_1K.txt', sep='^',
                        names=['BASE_YEAR', 'GRID_1K_CD', 'STAT_CD', 'POP'])
```

※ 데이터를 로드하기 전, 다운받은 통계 파일을 열어보면 구분자는 '^'(Caret)이고, 첫 행에 컬럼명이 없는 것을 확인할 수 있습니다. 이런 경우 파일을 불러오면서 컬럼명을 함께 지정할 수 있습니다.

3 격자 경계 및 통계 데이터 병합하기

- ▶ `grid_마라` 경계와 통계 및 `grid_마마` 경계와 통계 데이터를 각각 병합합니다.

```
[3]: grid_merged = pd.concat([grid_mara, grid_mama], ignore_index=True)
      stat_merged = pd.concat([stat_mara, stat_mama], ignore_index=True)
```

4 데이터 구조 확인하기

- ▶ 경계와 통계 각각의 파일이 잘 병합되었는지, 데이터 타입은 무엇인지 등 구조를 확인합니다.

```
[4]: print(grid_merged.info()) # 격자 경계 코드(GRID_1K_CD), 지리정보(geometry)로 구성
      print(stat_merged.info()) # 기준년도(BASE_YEAR), 격자 경계 코드(GRID_1K_CD)
                                   # 통계 코드(STAT_CD), 인구수(POP)
```

```
<class 'geopandas.geodataframe.GeoDataFrame'>
RangeIndex: 9549 entries, 0 to 9548
Data columns (total 2 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   GRID_1K_CD  9549 non-null   object
1   geometry    9549 non-null   geometry
dtypes: geometry(1), object(1)
memory usage: 149.3+ KB
None
```

➤ 데이터프레임은 총 9,549개의 행으로 구성되며, 2개의 열이 있습니다. 각각의 열에는 비어 있는 값이 없으며, 데이터 타입은 object(문자열)입니다.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 261928 entries, 0 to 261927
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   BASE_YEAR    261928 non-null  int64
1   GRID_1K_CD   261928 non-null  object
2   STAT_CD      261928 non-null  object
3   POP          261928 non-null  int64
dtypes: int64(2), object(2)
memory usage: 8.0+ MB
None
```

➤ 데이터프레임은 총 261,928개의 행으로 구성되며, 4개의 열이 있습니다. 각각의 열에는 비어 있는 값이 없으며, 데이터 타입은 int64(정수형), object(문자열)입니다.

※ GRID_1K_CD의 데이터 타입(object)이 동일하므로 추후 데이터 타입 일치 과정 없이 바로 결합이 가능합니다.

※ 추가로 `print(grid_mara.info())`와 `print(grid_mama.info())`, `print(stat_mara.info())`와 `print(stat_mama.info())`를 통해 병합 전의 각각 데이터 수, 데이터 타입 등을 확인할 수 있습니다.

5 통계 항목 필터링

- ▶ STAT_CD 컬럼에 있는 고유값을 확인하고 분석에 필요한 'to_in_001(총인구수)'만 필터링하여 stat_total에 저장합니다.

```
[5]: # STAT_CD 열에 있는 고유 값 확인
unique_stat_cd = stat_merged['STAT_CD'].unique()
print(unique_stat_cd)
```

```
['in_age_005' 'in_age_006' 'in_age_007' 'in_age_009' 'in_age_010'
'in_age_011' 'in_age_012' 'in_age_013' 'in_age_014' 'in_age_015'
'in_age_016' 'in_age_017' 'in_age_035' 'in_age_036' 'in_age_039'
'in_age_041' 'in_age_042' 'in_age_043' 'in_age_044' 'in_age_045'
'in_age_046' 'in_age_047' 'in_age_066' 'in_age_067' 'in_age_070'
'in_age_071' 'in_age_072' 'in_age_073' 'in_age_074' 'in_age_075'
'in_age_076' 'in_age_077' 'to_in_001' 'to_in_007' 'to_in_008'
'in_age_001' 'in_age_008' 'in_age_031' 'in_age_037' 'in_age_038'
'in_age_061' 'in_age_003' 'in_age_004' 'in_age_018' 'in_age_019'
'in_age_020' 'in_age_040' 'in_age_048' 'in_age_063' 'in_age_064'
'in_age_069' 'in_age_078' 'in_age_079' 'in_age_080' 'in_age_033'
'in_age_034' 'in_age_068' 'in_age_002' 'in_age_062' 'in_age_065'
'in_age_032' 'in_age_049' 'in_age_021' 'in_age_050' 'in_age_081'
'in_age_051']
```

```
[6]: # 총인구 데이터만 필터링 ('to_in_001')
stat_total = stat_merged[stat_merged['STAT_CD'] == 'to_in_001']
# 필터링 후 STAT_CD 고유값 재확인
unique_stat_cd = stat_total['STAT_CD'].unique()
print(unique_stat_cd)
```

```
['to_in_001']
```

6 분석에 필요한 컬럼만 선택하기

- ▶ stat_total에서 GRID_1K_CD와 POP 컬럼만 남겨서 나머지 불필요한 열을 제거합니다.(추후 경계 파일과 병합 과정에서 필요한 컬럼만 저장)

```
[7]: stat_total = stat_total[['GRID_1K_CD', 'POP']]
```

7 부산시 경계와 격자 경계가 교차하는 격자만 선택하기

```
[8]: grid_intersects = grid_merged[grid_merged.geometry.intersects(bord_sido.geometry.union_all())]
```

- ▶ grid_mara와 grid_mama는 부산과 경상도 일대를 포함하는 1km 격자 경계이며, 부산시 인구 분석을 위해서 부산시 경계와 교차하는 격자 경계만 추출합니다.

8 경계와 통계 병합하기

- ▶ grid_intersects와 stat_total 두 개의 데이터프레임을 'GRID_1K_CD'를 기준으로 left join(왼쪽 조인)하여 병합합니다.

```
[9]: grid_intersects = grid_intersects.merge(stat_total, on='GRID_1K_CD', how='left')
      # 결합 후 통계값이 없는 구간에 NaN이 들어가 있을 수 있음(결측값 확인)
      print(grid_intersects.isna().sum())
```

```
GRID_1K_CD    0
geometry      0
POP          244
dtype: int64
```

```
[10]: # 결측값을 0으로 채우기
      grid_intersects['POP'] = grid_intersects['POP'].fillna(0)
```

※ 「SGIS 자료제공」의 격자 통계 특성 상, 인구 또는 사업체가 없는 격자에 대해서는 통계 데이터를 생성하지 않으므로 결측값을 '0'으로 채울 수 있습니다.

9 총인구수를 기준으로 구간 나누기

- ▶ 총인구수('POP')를 5개의 구간으로 나누고 각 구간에 대한 라벨을 지정합니다.

```
[11]: # 인구 구간 직접 지정하여 나누기
      print(grid_intersects['POP'].min(), grid_intersects['POP'].max()) # 최소값과 최대값 확인
```

```
0.0 31611.0
```

```
[12]: bins = [0, 1000, 5000, 10000, 20000, 32000] # 적절한 구간 지정
      labels = ['0~1000', '1000~5000', '5000~10,000', '10,000~20,000', '20,000~32,000'] # 구간의 라벨
      # 각 인구 수에 대해 구간 할당
      grid_intersects['POP_BINS'] = pd.cut(grid_intersects['POP'], bins=bins, labels=labels, right=False)
      print(grid_intersects.head()) # 인구 구간('POP_BINS') 컬럼 추가 확인
```

	GRID_1K_CD	geometry	POP	POP_BINS
0	마라1569	POLYGON ((1115000 1669000, 1115000 1670000, 11...	0.0	0~1000
1	마라1668	POLYGON ((1116000 1668000, 1116000 1669000, 11...	0.0	0~1000
2	마라1669	POLYGON ((1116000 1669000, 1116000 1670000, 11...	0.0	0~1000
3	마라1773	POLYGON ((1117000 1673000, 1117000 1674000, 11...	0.0	0~1000
4	마라1774	POLYGON ((1117000 1674000, 1117000 1675000, 11...	0.0	0~1000

10 지도 시각화 하기

[13]:

```
# 시각화 설정(그림과 축 설정)
fig, ax = plt.subplots(1, 1, figsize=(12, 12))

# 격자 시각화(격자 경계 선은 연하게 설정)
grid_intersects.plot(column='POP_BINS', ax=ax, legend=True,
                     cmap='OrRd', edgecolor='black', linewidth=0.5, alpha=0.7)

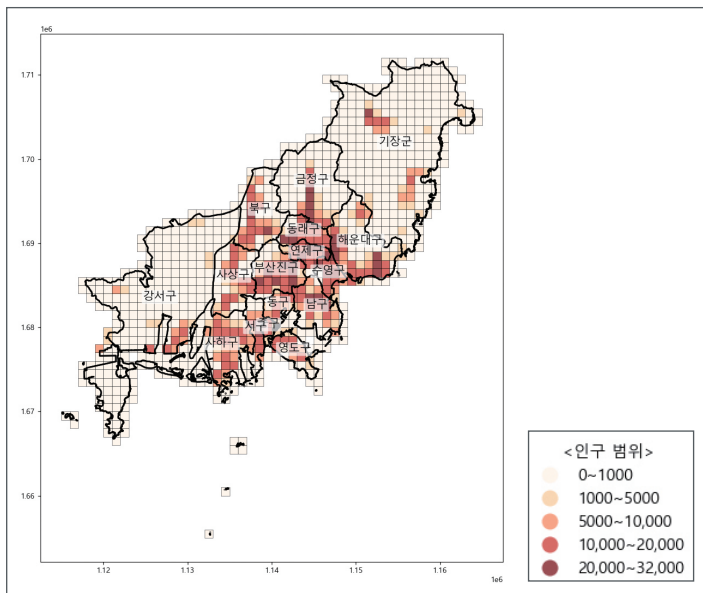
# 부산시 경계 추가 (경계선만 보이도록 설정, 배경은 투명)
bord_sido.plot(ax=ax, color='none', edgecolor='black', linewidth=2)

# (추가) 부산시 시군구 경계를 추가하여 명확하게 분석하기
bord_sgg = gpd.read_file(prj_dir + '/' + 'bnd_sigungu_21_2024_2Q.shp')
bord_sgg.plot(ax=ax, color='none', edgecolor='black', linewidth=2)
bord_sgg['centroid'] = bord_sgg.geometry.centroid # 중심점 계산
for idx, row in bord_sgg.iterrows():
    ax.text(row['centroid'].x, row['centroid'].y, row['SIGUNGU_NM'], fontsize=15, ha='center',
           color='black', bbox=dict(facecolor='white', alpha=0.7, edgecolor='none',
                                   boxstyle='round,pad=0.1'))

# 범례 위치 및 제목 설정
ax.get_legend().set_bbox_to_anchor((0.5, -0.05)) # 범례 위치 조정
ax.get_legend().set_title("<인구 범위>") # 범례 제목 설정
ax.set_title('<부산시 1km 격자 인구 분석>', fontsize=16) # 지도 제목 설정

# 레이아웃 조정 및 출력
plt.tight_layout() # 그래프 레이아웃이 겹치지 않도록 자동 조정
plt.savefig(prj_dir + '/' + '부산시 격자 인구 분석 지도.png')
plt.show() # 시각화 결과 화면에 출력
```

| 부산시 1km격자 인구 분석 |





PART

V

서울시 청년인구
격자별 순위 분석
(R)

R



V | 서울시 청년인구 격자별 순위 분석

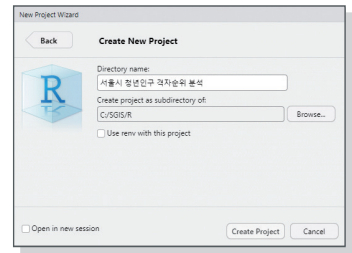
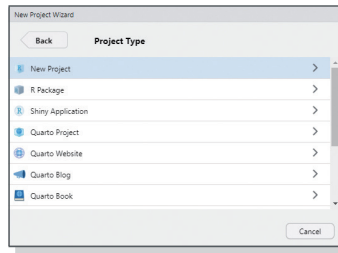
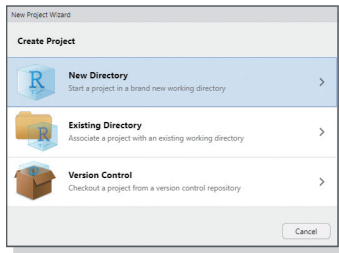
통계데이터센터(SDC)에서 서비스하는 청년인구 격자통계와 R Studio를 이용해서 서울시 청년인구 격자순위를 분석하는 과정을 단계별로 설명합니다.

1 프로젝트 설정 및 분석 준비하기

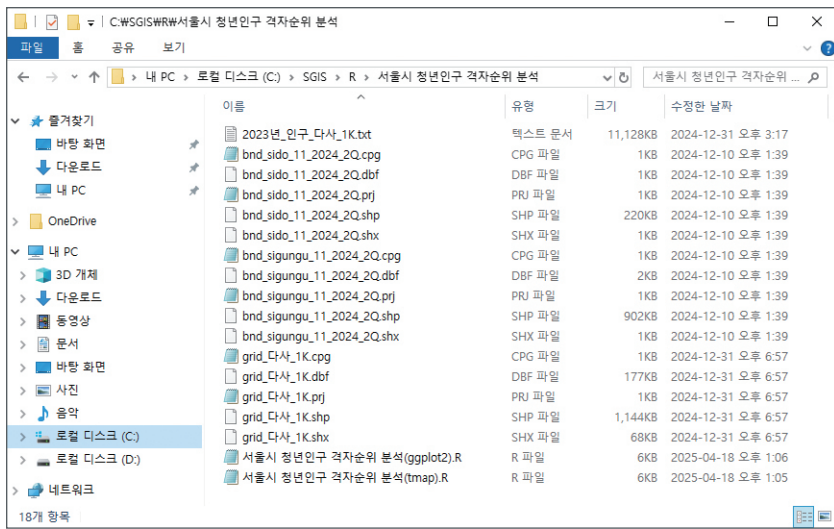
- ▶ R Studio*를 실행하고 **File**, **New Project...** 메뉴를 클릭하고 **New Project Wizard** 창에서 첫 번째 **New Directory** 를 선택합니다. **Project Type** 창에서는 **New Project** 를 선택합니다.

* R 4.4.3-win, RStudio 2024.12.1, RTools 4.4 버전을 기준으로 작성했습니다.

- ▶ **Browse...** 를 눌러서 프로젝트를 저장할 폴더를 선택하고 **Directory name** 에 **서울시 청년인구 격자순위 분석** 등 프로젝트명을 입력하고 **Create Project** 를 클릭하면 프로젝트가 생성됩니다.



- ▶ 여기서는 **C:\WSGIS\RW\서울시 청년인구 격자순위 분석** 폴더에 분석에 필요한 경계와 통계, 분석 소스 코드 파일을 저장한다고 가정합니다.



- ▶ **File** , **Open File...** 메뉴에서 **서울시 청년인구 격자순위 분석.R** 파일을 선택해서 엽니다. **File** , **New File** , **R Script** 메뉴에서 새 파일을 열고 아래의 코드를 순서대로 입력해도 됩니다.

```

1
2 # 1. 분석 패키지 설치 및 사용준비 (tmap 버전)
3
4 # install.packages("dplyr")      # 한번 설치하고 주석처리(#) 하기
5 # install.packages("sf")        # 한번 설치하고 주석처리(#) 하기
6 # install.packages("tmap")      # 한번 설치하고 주석처리(#) 하기
7
8 library(dplyr)                  # 데이터 처리: mutate() 등
9 library(sf)                    # 공간 데이터 처리: st_read() 등
10 library(tmap)                  # tmap 시각화: tm_shape() 등
11
12 # 프로젝트 폴더 지정
13 prj_dir <- "C:/SGIS/R/서울시 청년인구 격자순위 분석"
14

```

- ▶ `install.packages()` 함수로 분석에 필요한 `sf`, `tmap`, `dplyr` 패키지와 의존성이 있는 패키지를 설치하고 `library()` 함수로 사용할 준비를 합니다. 필요한 패키지를 설치한 후에는 4~6째 줄은 다시 실행되지 않도록 주석 처리 해줍니다.
- ▶ 프로젝트 디렉토리(**C:/SGIS/R/서울시 청년인구 격자순위 분석**)를 변수에 저장해서 통계와 경계 파일을 불러올 때 사용할 준비를 합니다. R 소스코드에서 폴더를 구분할 때는 'W' 대신 '/'를 사용합니다.

| R Studio 창 구조 |

The screenshot shows the R Studio interface with the following components:

- R 코드 창 (R Script Editor):** Contains the R script code for loading data and creating maps.
- Environment 창 (Environment Pane):** Lists the objects in the environment, including `join`, `join_rank`, `join_topn`, `map_all`, `map_intersects`, `map_topn`, `stat`, and `stat_young`.
- Plots/Help 창 (Plots/Help Pane):** Displays a map titled "2023년 서울시 청년인구 격자 순위" (2023 Seoul Youth Population Grid Ranking). The map shows the city of Seoul divided into grid cells, colored according to the ranking. A legend on the left indicates the ranking ranges: 1-5,000 (lightest), 5,000-10,000, 10,000-15,000, 15,000-20,000, and 20,000-24,000 (darkest).
- Console 창 (Console Pane):** Shows the output of the R script, including messages about loading data and creating maps.

2 서울시 경계와 겹치는 격자 경계 만들기

```

15
16 # 2. 서울시 경계와 겹치는 격자 경계 만들기
17
18 # 서울시 시도 경계
19 file_path <- paste(prj_dir, "bnd_sido_11_2024_2Q.shp", sep="/")
20 bord_sido <- st_read(file_path) # 경계 읽어들이기, geometry 컬럼에 공간정보 포함
21
22 # 서울시 시군구 경계
23 file_path <- paste(prj_dir, "bnd_sigungu_11_2024_2Q.shp", sep="/")
24 bord_sigungu <- st_read(file_path)
25
26 # 서울시를 포함하는 1km 격자 경계('다사')
27 file_path <- paste(prj_dir, "grid_다사_1K.shp", sep="/")
28 bord_grid <- st_read(file_path)
29

```

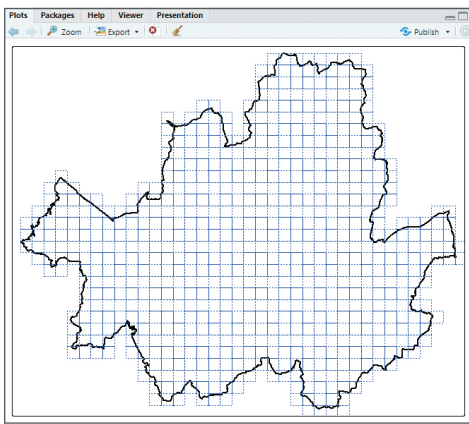
- ▶ `st_read()` 함수로 서울시 시도 경계(`bnd_sido_11_2024_2Q.shp`), 서울시 시군구 경계(`bnd_sigungu_11_2024_2Q.shp`), 1km 다사 격자 경계(`grid_다사_1K.shp`)를 읽어옵니다. `st_read()` 함수는 geometry 등 공간정보 컬럼이 있는 데이터를 처리할 수 있습니다.

```

29
30 # 서울시 경계와 겹치는 격자 경계만 저장
31 bord_intersects <-
32   st_join(bord_grid, bord_sido, join=st_intersects, left=FALSE)
33
34 # 서울시와 겹치는 격자 경계를 지도에서 확인
35 map_intersects <-
36   tm_shape(bord_sido) + tm_borders(lwd=2.0, col="black") +
37   tm_shape(bord_intersects) + tm_borders(lty="dotted", lwd=0.8, col="blue")
38 # 선종류(lty), 너비(lwd), 색상 지정
39
40 map_intersects # 지도를 'Plots' 탭에서 확인
41

```

- ▶ `st_join()` 함수로 서울시 시도 경계와 겹치는 격자 경계를 선택합니다. `join` 파라미터는 공간연산 종류를 지정하는데 `st_intersects` 외에 다양한 공간연산을 선택할 수 있습니다. `left` 파라미터를 `FALSE`로 지정해서 inner join으로 처리했습니다.
- ▶ `tm_shape()` 함수와 `tm_borders()`, `tm_polygons()` 함수 등을 조합해서 지도를 그립니다. 서울시 시도 경계와 격자 경계를 중첩하면 R Studio 화면 오른쪽 **Plots** 탭에서 아래와 같은 지도를 확인할 수 있습니다.



3 청년인구 통계 선별하기

```

42
43 # 3. 청년인구 통계 선별하기
44
45 file_path <- paste(prj_dir, "2023년_인구_다사_1k.txt", sep="/")
46 stat <- read.table(file=file_path, fileEncoding="UTF-8", sep="\t")
47 # 한글 포함한 파일 인코딩을 'UTF-8'로 지정
48 colnames(stat) <- c("BASE_YEAR", "GRID_CODE", "STAT_CODE", "STAT_VAL")
49 # 통계파일의 컬럼명 지정
50
51 head(stat) # 통계파일의 컬럼명과 데이터 확인
52
53 sort(unique(stat[, "STAT_CODE"]))
54 # 통계코드 종류 확인(5세연령 인구, 연령그룹 인구, 총인구 등)
55
56 stat_young <- stat[stat$STAT_CODE=="in_grp_005", ]
57 # 청년인구 통계를 별도로 저장
58
59 sort(unique(stat_young[, "STAT_CODE"]))
60 # 청년인구('in_grp_005' 컬럼) 저장 확인
61

```

- ▶ read.table() 함수로 인구통계 파일을 읽어서 colnames() 함수로 컬럼명을 의미있게 지정합니다. 청년인구 통계코드 'in_grp_005'에 해당하는 통계값만 별도로 저장합니다. head() 함수로 전과 후의 통계코드 종류를 출력해서 자료처리 결과를 확인할 수 있습니다.

※ 청년인구(15~34세)는 통계데이터센터(SDC)에서만 이용할 수 있습니다. SGIS 포털 자료를 내려받는 경우 in_age_005(20~24세) 등 5세 연령 인구를 이용해서 분석해보세요.

※ 통계속성 코드는 자료제공 메뉴에서 내려받을 수 있는 statistics_code.xlsx 참조

4 경계와 통계 조인하고 전체 순위 계산하기

```

62
63 # 4. 경계와 통계 조인하고 순위 계산하기
64
65 # 경계와 통계 데이터셋과 key 컬럼 지정
66 join <- merge(x=bord_intersects, y=stat_young,
67 by.x="GRID_1K_CD", by.y="GRID_CODE", all.x=TRUE)
68 # all.x=TRUE: 통계값(y)이 매칭되지 않아도 경계(x)는 남기기
69
70 join # 조인 결과 확인
71
72 # 불필요한 컬럼 정리, 기준연도(BASE_YEAR) NULL값 채우기
73 join <- join %>% select(-BASE_DATE, -SIDO_CD, -SIDO_NM, -STAT_CODE)
74 join$BASE_YEAR <- "2023"
75
76 # RANK(순위) 컬럼 추가
77 join_rank <- join %>% mutate(RANK = min_rank(desc(STAT_VAL)))
78
79 # RANK 컬럼 추가 확인
80 join_rank
81

```

- ▶ merge() 함수로 서울시와 겹치는 격자 경계와 청년인구 통계를 조인합니다. key에 해당하는 컬럼명이 다를 경우 각각 지정합니다. 조인 방법은 all.x 등 파라미터로 조정할 수 있습니다. mutate() 함수는 새로운 컬럼을 추가하고 select() 함수는 컬럼을 선택해서 저장하는 방법을 제공합니다. 컬럼명 앞의 '-'는 해당 컬럼을 제외한다는 의미입니다.
- ▶ 통계값이 매칭되지 않은 경우 조인된 컬럼은 'NA'로 저장되어 있습니다. 필요하다면, 기준연도 (BASE_YEAR) 컬럼을 '2023' 등으로 일괄 저장할 수 있습니다.
- ▶ min_rank() 함수로 청년인구 순위를 계산합니다. desc() 함수를 이용하면 내림차순으로 정렬되어 청년 인구가 가장 많은 격자가 1순위가 됩니다.

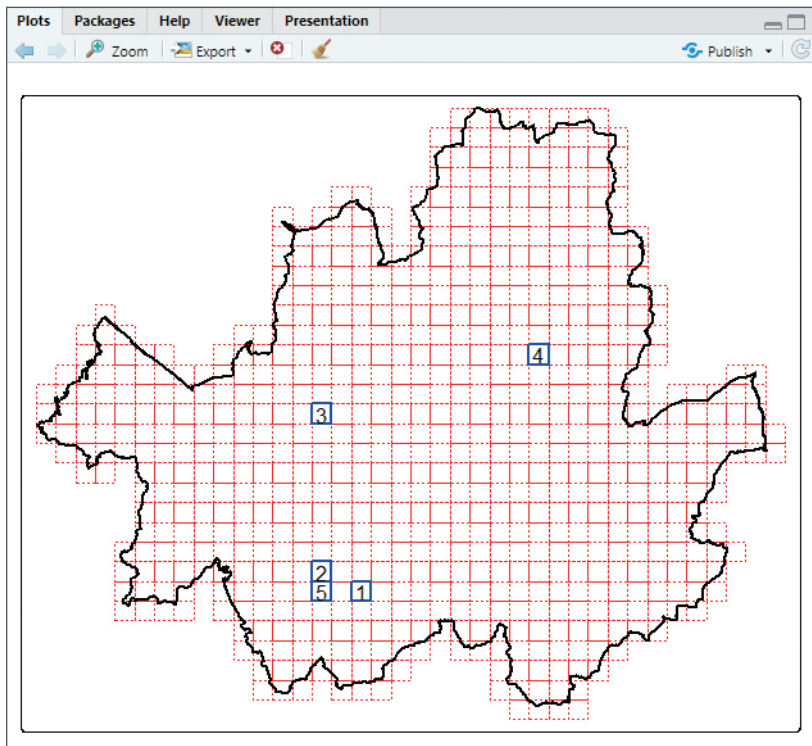
5 상위 순위 계산하기

```

82 # 5. 상위 순위 계산하기
83
84 # Top N 순위 계산
85 topN <- 5 # 상위 순위 N
86
87 # 상위 순위 저장
88 join_topN <-
89   join_rank[join_rank$RANK <= topN & ! is.na(join_rank$RANK), ]
90
91 # Top N 순위 포함한 지도
92 map_topN <-
93   # 1. 격자 경계(전체 순위)
94   tm_shape(join_rank) + tm_borders(lty="dotted", lwd=0.8, col="red") +
95   # 2. 서울시 시도 경계
96   tm_shape(bord_sido) + tm_borders(lty="solid", lwd=2.0, col="black") +
97   # 3. 격자 경계(상위 순위)
98   tm_shape(join_topN) + tm_borders(lty="solid", lwd=2.0, col="blue") +
99   tm_text(text="RANK", col="black")
100
101
102
103 map_topN # 지도 중간 결과 확인
104

```

- ▶ 상위 순위 격자를 별도로 저장합니다. 통계값이 없는 경우 순위 컬럼에 'NA'가 저장되어 있기 때문에 'NA'가 아닌 자료만 선별하도록 is.na() 함수로 조건을 지정합니다.
- ▶ 격자 경계와 서울시 경계, 상위 순위 격자를 지도로 확인해 봅니다. 2개 이상의 경계는 '+'로 연결해서 중첩할 수 있습니다. **Plots** 탭에서 아래와 같은 지도를 확인할 수 있습니다.



6 청년인구 단계구분도와 격자순위 시각화하기

```

105 # 6. 청년인구 단계구분도와 격자순위 시각화
106
107 map_all <-
108   # 1. 격자 경계(전체 순위)
109   tm_shape(join_rank) +
110     tm_polygons(fill="STAT_VAL", lty="dotted",
111               fill.legend=tm_legend(title="격자별 청년인구(명)", position=c("left", "top")),
112               fill.scale=tm_scale_intervals(breaks=c(1, 5000, 10000, 15000, 20000, 24000),
113               labels=c("1~5,000", "5,000~10,000", "10,000~15,000", "15,000~20,000", "20,000~24,000"),
114               values=c("#FFFFB2", "#FECC5C", "#FD8D3C", "#F03B20", "#BD0026"), # 색상 코드 열거
115               # values="brewer.yl_or_rd", # 또는 색상 파레트 지정
116               value.na="white", label.na="")) +
117   # 2. 서울시 시도 경계
118   tm_shape(bord_sido) + tm_borders(lwd=2.0, col="black") +
119   # 3. 서울시 시군구 경계
120   tm_shape(bord_sigungu) + tm_borders(lwd=1.0, col="black") +
121   tm_text(text="SIGUNGU_NM", col="black", size=1.0, col_alpha=0.7) +
122   # 4. 격자 경계(상위 순위)
123   tm_shape(join_topN) + tm_borders(lwd=2.0, col="blue") +
124   tm_text(text="RANK", col="white", size=0.6) +
125   # 5. 제목과 레이아웃
126   tm_title(text="2023년 서울시 청년인구 격자 순위") +
127   tm_layout(inner.margins=0.1)
128
129
130
131 map_all # 전체 맵을 'Plots' 탭에서 확인
132

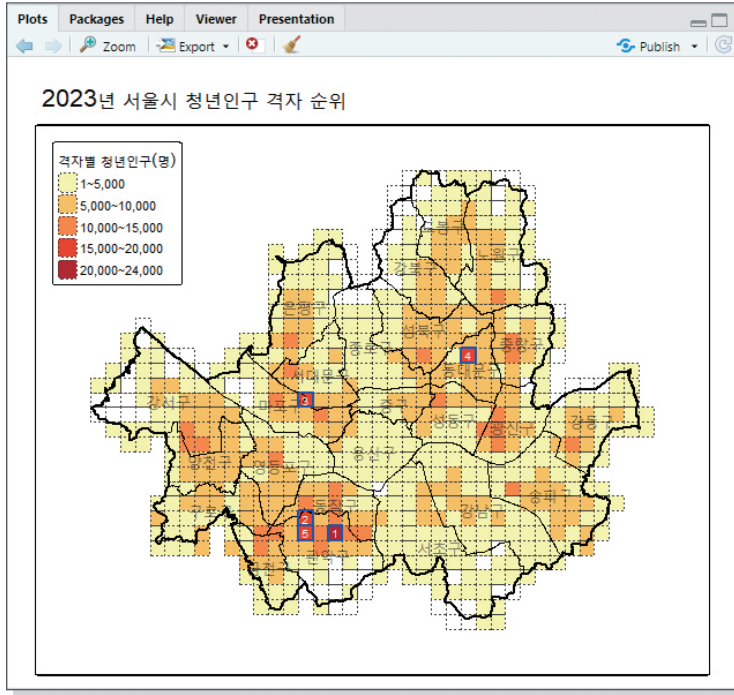
```

- ▶ `tm_polygons()` 함수와 `tm_scale_intervals()`, `tm_legend()` 함수를 함께 사용하여 단계구분도와 범례를 지도에 나타냅니다. 단계구분 색상은 색상 코드를 열거하거나 미리 정의된 색상 파레트를 지정하는 방법으로 정할 수 있습니다.

5단계 색상 예시 및 코드					색상 파레트
#FFFFB2	#FECC5C	#FD8D3C	#F03B20	#BD0026	brewer.yl_or_rd
#EFF3FF	#BDD7E7	#6BAED6	#3182BD	#08519C	brewer.blues
#EDF8FB	#B2E2E2	#66C2A4	#2CA25F	#006D2C	brewer.bu_gn

- ▶ 상위 순위의 격자를 강조해서 표시하고 시군구 경계와 명칭을 함께 표시해서 해당 격자가 속하는 시군구를 확인할 수 있도록 했습니다.
- ▶ `tm_title()` 함수와 `tm_layout()` 함수를 사용해서 지도 제목과 여백 비율 등을 지정합니다.

▶ **Plots** 탭에서 아래와 같은 최종 지도를 확인할 수 있습니다.



💡 도움말 참고하기

- Help 창에서 함수를 검색하면 파라미터의 종류와 세부사항, 코드 작성 예시, 참고자료 등을 확인할 수 있습니다.
- `tm_shape()`, `ggplot()` 등의 함수 도움말 페이지를 참고해서 단계구분도를 원하는 형태로 바꿔보세요.

Shape (spatial object) specification

Description
Specify a shape, which is a spatial object from one of these spatial object class packages: [sf](#), [stars](#), or [terra](#).

Usage

```
tm_shape(
  shp,
  bbox = NULL,
  crs = NULL,
  is.main = NA,
  name = NULL,
  unit = NULL,
  filter = NULL,
  ...
)
```

Arguments

shp	Spatial object
bbox	Bounding box of the map (only used if shp is the main shape (see is.main))

The search string was "ggplot()"

비니에트:

- [ggplot2-extending-ggplot2](#) Extending ggplot2 [HTML](#) [source](#) [R code](#)
- [ggplot2-ggplot2-in-packages](#) Using ggplot2 in packages [HTML](#) [source](#) [R code](#)
- [ggplot2-ggplot2-scales](#) Aesthetic specifications [HTML](#) [source](#) [R code](#)
- [ggplot2-ggplot2](#) Introduction to ggplot2 [HTML](#) [source](#) [R code](#)

코드 설명:

- [plotly-crosstalk-highlight-ggplotly](#) Linking ggplot2 plots [Run demo](#)
- [plotly-sf-ggplot2](#) Mapping sf objects via ggplot2 and geom_sf() [Run demo](#)
- [sf-ggplot](#) shows use of geom_sf [Run demo](#)
- [units-ggplot2](#) demonstrates automatic units in default axis labels in ggplot2 [Run demo](#)

도움말 페이지:

- [colorspace-scale colour binned diverging](#) HCL-Based Binned Diverging Color Scales for ggplot2

7 지도 그림과 경계파일 저장하기

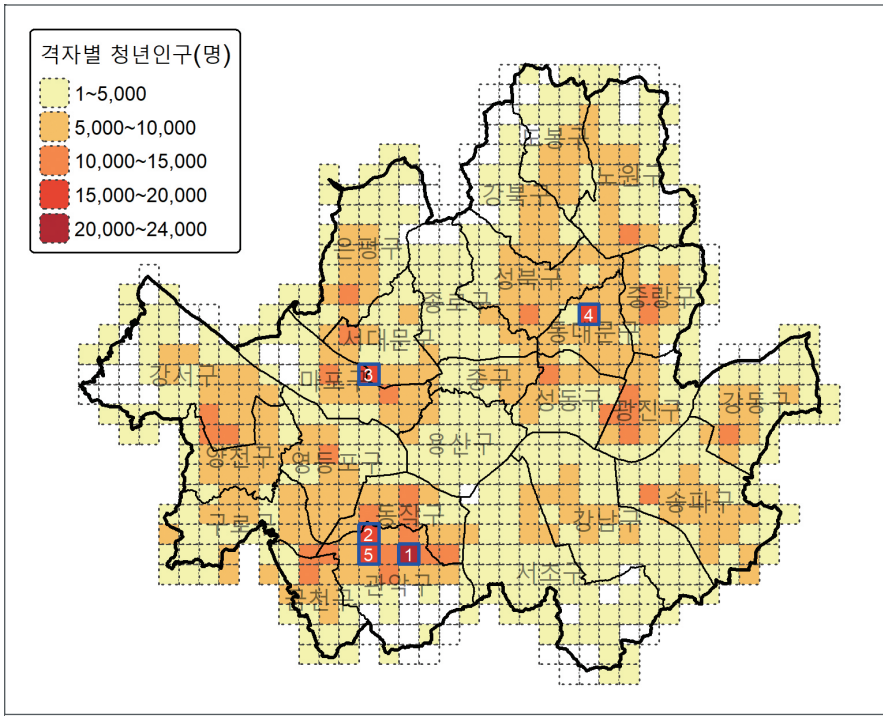
```

133
134 # 7. 지도 그림과 경계파일 저장하기
135
136 # 지도를 그림 파일로 저장
137 file_path <- paste(prj_dir, "2023년 서울시 청년인구 격자순위 맵.png", sep="/")
138 tmap_save(tm=map_all, filename=file_path, width=1800, height=1500, dpi=300)
139 # 너비, 높이(픽셀 단위), 해상도 지정
140
141 # 경계를 SHP 파일로 저장(전체 격자, 상위순위 격자)
142 file_path <- paste(prj_dir, "2023년 서울시 청년인구 격자순위.shp", sep="/")
143 st_write(join_rank, dsn=file_path, append=FALSE, layer_options="ENCODING=UTF-8")
144
145 file_path <- paste(prj_dir, "2023년 서울시 청년인구 격자순위 TopN.shp", sep="/")
146 st_write(join_topN, dsn=file_path, append=FALSE, layer_options="ENCODING=UTF-8")
147

```

- ▶ 분석 과정의 지도와 경계를 파일로 저장할 수 있습니다. `tmap_save()` 함수를 이용해서 지도를 지정된 크기와 해상도의 그림파일로 저장합니다.

| 2023년 서울시 청년인구 격자 순위 |



- ▶ `st_write()` 함수로 통계속성이 결합된 격자경계를 .shp 파일 등으로 저장하면 다른 지도시각화 도구에서도 활용할 수 있습니다.



SGIS 소지역 통계
분석사례집

PART • VI

서울시 인구수 현황
격자 분석
(ArcGIS)



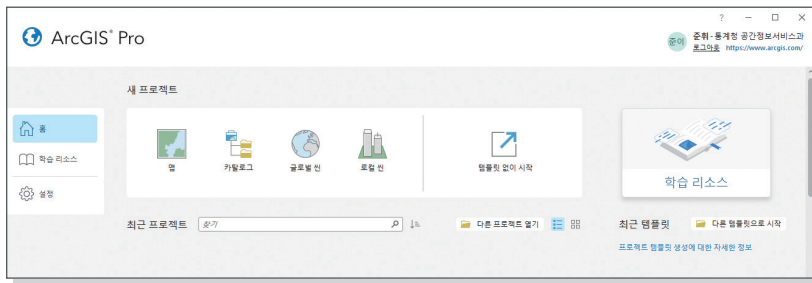


VI | 서울시 인구수 현황 격자 분석

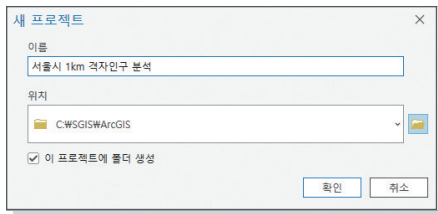
공공데이터포털에서 내려받은 서울시 경계(**bnd_sido_11**), 격자 경계(**grid_다사_1K**), 격자 인구 (**2023년_인구_다사_1K**)를 ArcGIS Pro(3.2 버전)으로 분석하는 과정을 단계별로 설명합니다.

1 프로젝트 설정하기

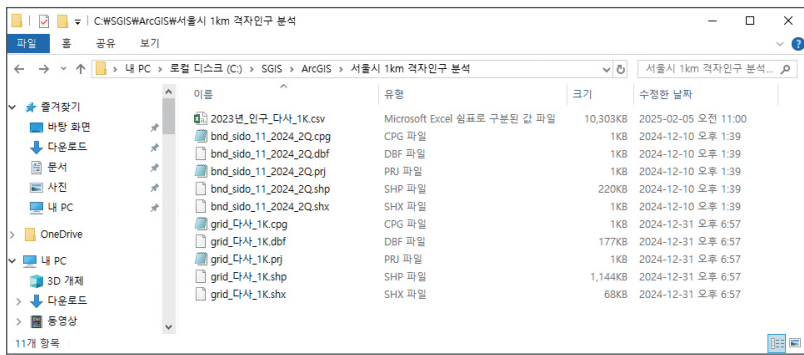
- ▶ ArcGIS Pro를 실행하고 첫 화면에서 새 프로젝트 종류 중 **맵** ()을 선택합니다.



- ▶ **새 프로젝트 생성** 화면에서 **찾아보기** 아이콘()을 눌러서 프로젝트 폴더를 선택합니다. 생성할 프로젝트 이름을 입력하고 **확인** 을 선택하면 프로젝트가 생성됩니다.

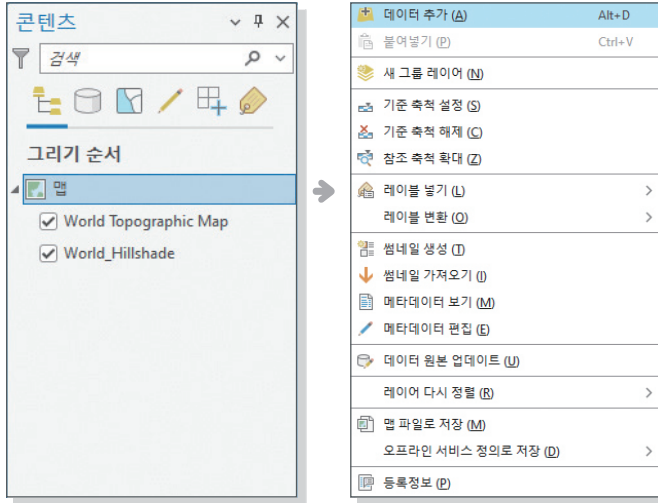


- ▶ 여기서는 **C:\WSGISWArcGIS\서울시 1km 격자인구 분석** 폴더에 분석에 필요한 경계와 통계를 저장한 것으로 가정합니다.

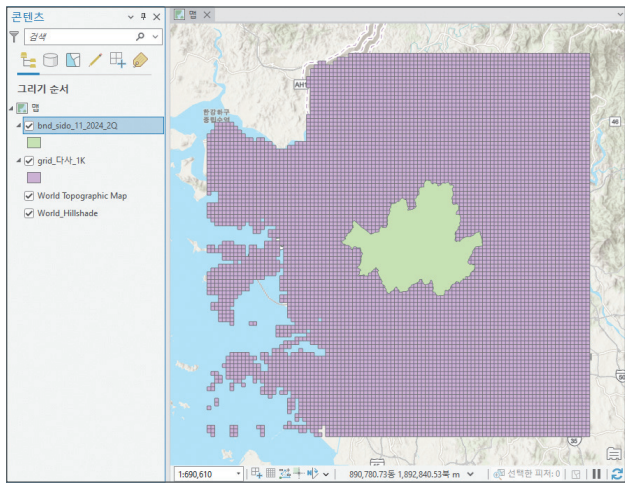
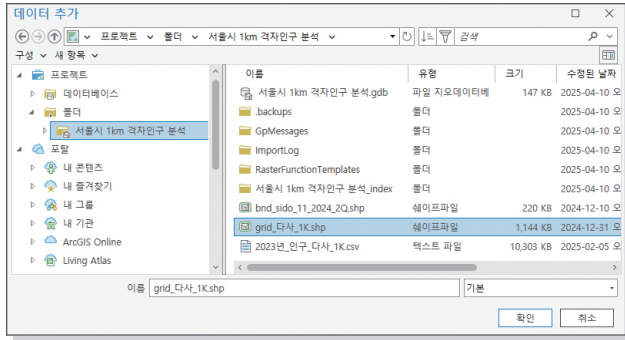


2 경계를 맵에 추가하기

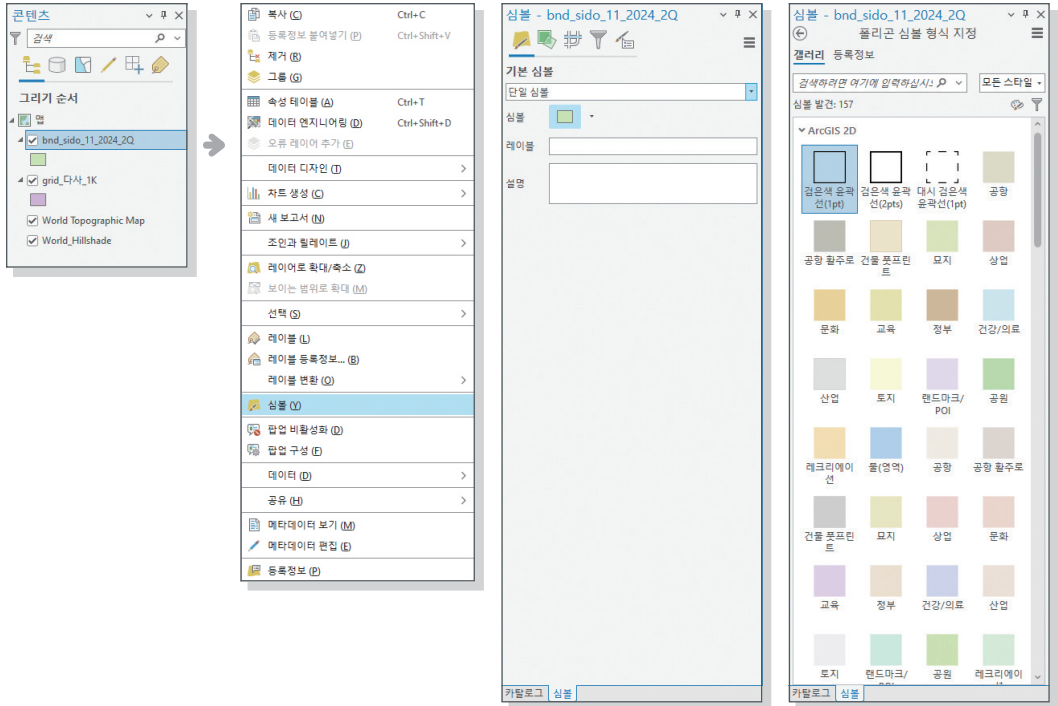
- ▶ 화면 왼쪽 **콘텐츠** 창 맵 위에서 마우스 오른쪽 버튼을 클릭해서 **데이터 추가** 를 선택합니다.



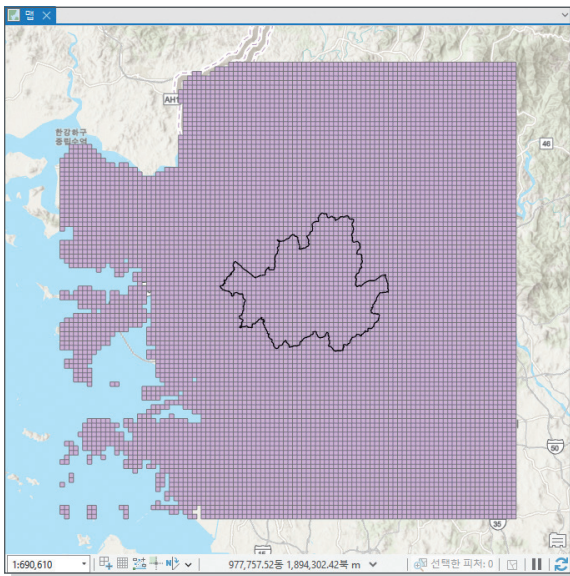
- ▶ **데이터 추가** 화면에서 서울시 영역을 포함하는 격자 경계 **grid_다사_1K.shp** 와 서울시 경계 **bnd_sido_11_2024_2Q.shp** 파일 순으로 각각 맵에 추가합니다.



- ▶ 서울시 경계(**bnd_sido_11_2024_2Q**) 위에서 마우스 오른쪽 버튼을 클릭해서 **심볼** () 메뉴를 선택합니다. 화면 오른쪽에 심볼 화면이 나타나면 현재 심볼()을 선택하고 심볼 갤러리에서 **검은색 윤곽선(1pt)** 을 선택합니다.

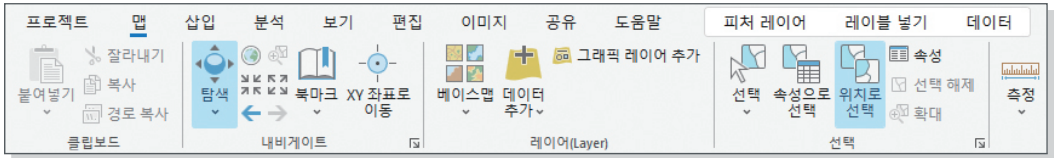


- ▶ **맵** 화면에서 서울시 경계가 검은색 윤곽선으로 표시되는 것을 확인할 수 있습니다.

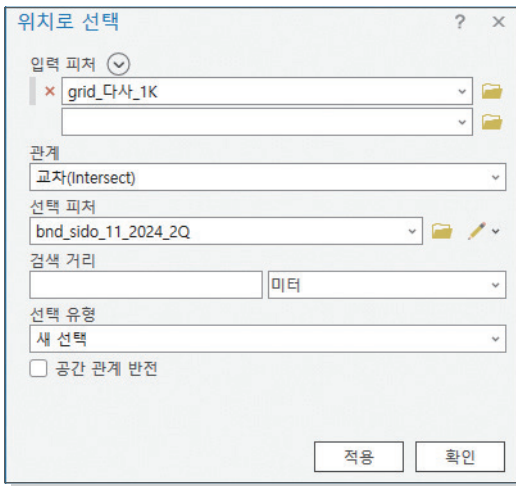


3 서울시 경계와 겹치는 격자경계 선택하기

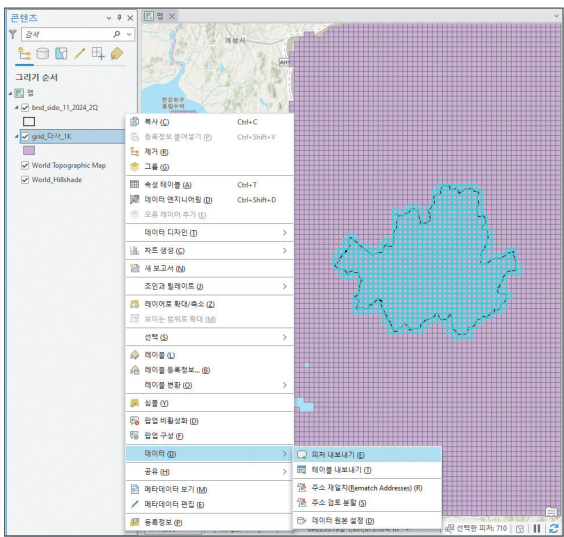
- ▶ 서울시 경계와 겹치는 격자 경계를 선택하기 위해 상단 두번째 **맵** 메뉴에서 선택 그룹의 **위치로 선택** () 메뉴를 선택합니다.



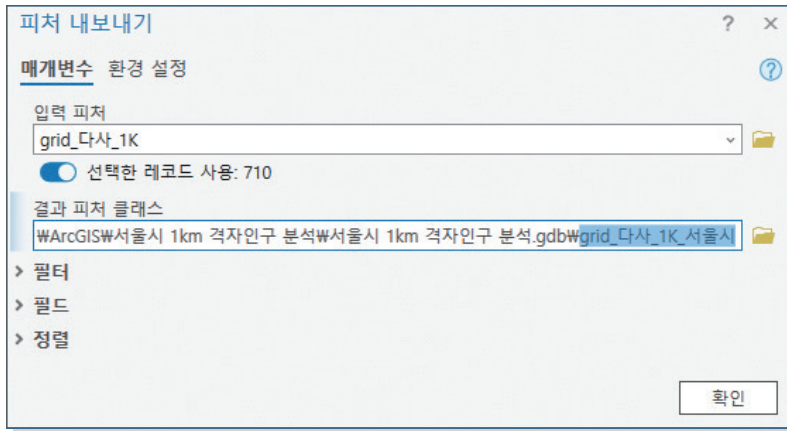
- ▶ **위치로 선택** 화면에서 **입력 피쳐** 로 격자 경계(**grid_다사_1K**), **선택 피쳐** 로 서울시 경계 (**bnd_sido_11_2024_2Q**)를 각각 입력하고 확인을 선택합니다.



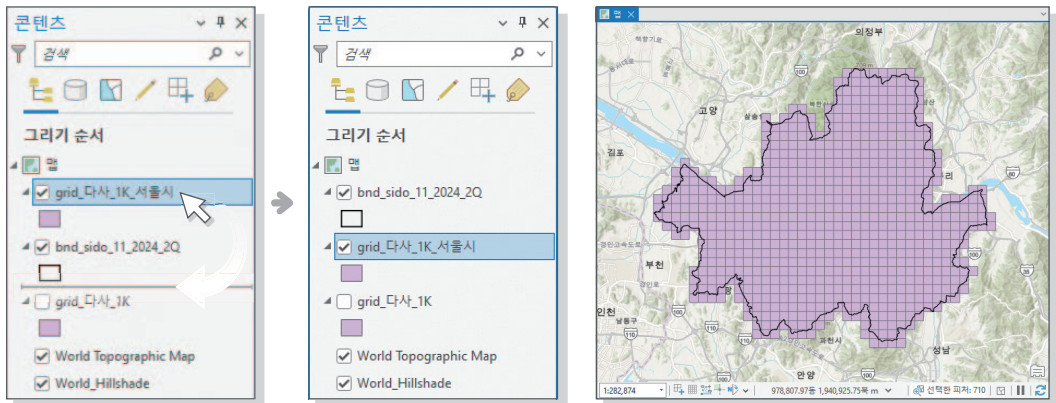
- ▶ 서울시 경계와 겹치는 격자 경계가 맵 화면에 하늘색으로 표시됩니다. 격자 경계(**grid_다사_1K**) 위에서 마우스 오른쪽 버튼을 클릭하고 **데이터** , **피쳐 내보내기** 를 순서대로 선택합니다.



- ▶ 피쳐 내보내기 창에서 **결과 피쳐 클래스** 를 **grid_다사_1K_서울시** 등으로 수정 입력하고 확인을 선택합니다. 선택한 레코드만 저장된다는 내용이 표시되어 있습니다.

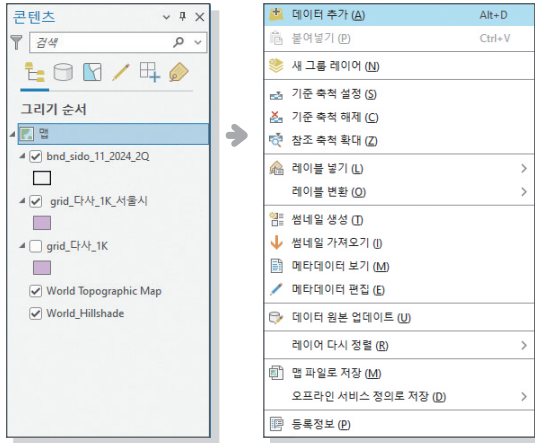


- ▶ **콘텐츠** 창에서 **grid_다사_1K** 앞 체크박스를 해제하고 서울시 격자경계(**grid_다사_1K_서울시**)를 서울시 경계(**bnd_sido_11_2024_2Q**) 아래쪽으로 끌어서 옮깁니다. **맵** 창에는 서울시 경계, 서울시와 겹치는 격자경계가 함께 표시됩니다.

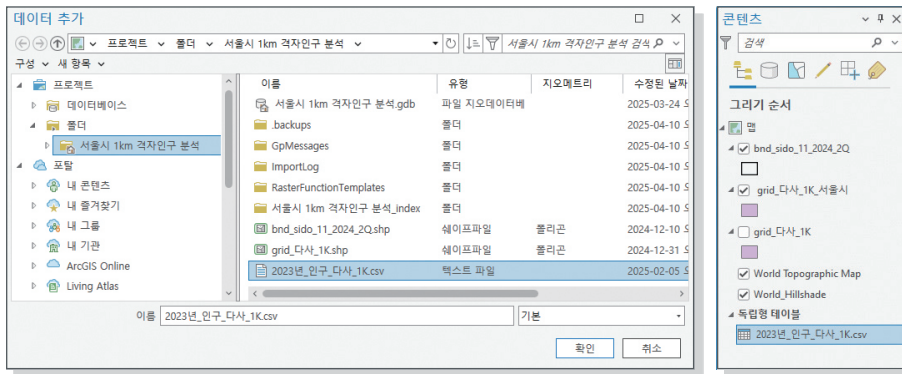


4 격자 인구통계 불러오기 및 총인구 선택하기

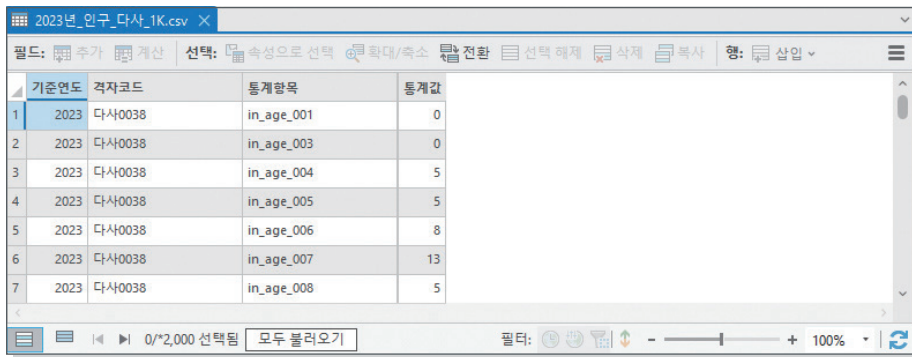
- ▶ CSV 형식의 격자 인구통계를 불러오기 위해 **콘텐츠** 창 **맵** 위에서 마우스 오른쪽 버튼을 클릭해서 **데이터 추가** 를 선택합니다.



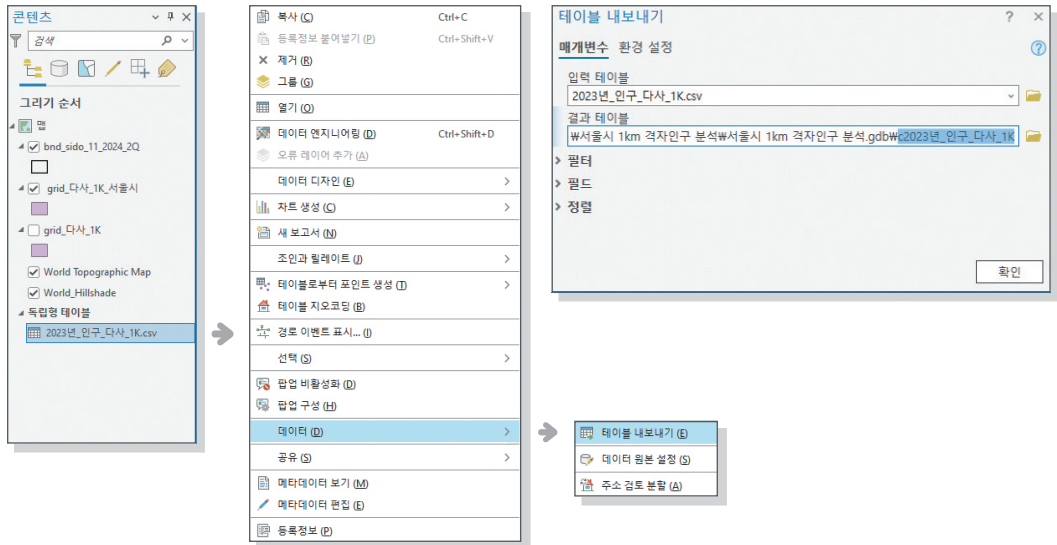
- ▶ **데이터 추가** 화면에서 **2023년_인구_다사_1K.csv** 파일을 선택하여 맵에 추가합니다. **콘텐츠** 창에 독립된 테이블 로 분류되어 파일이 추가된 것을 확인할 수 있습니다.



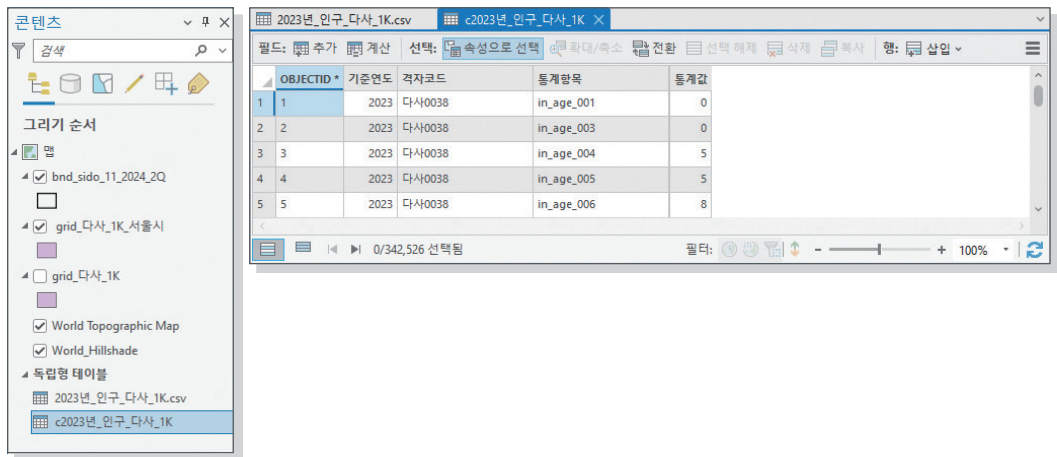
- ▶ **콘텐츠** 창에서 **2023년_인구_다사_1K.csv** 테이블을 마우스 오른쪽 버튼 클릭해서 **열기** (열기 아이콘)를 선택하거나 **Ctrl+T** 를 누르면 테이블 내용을 확인할 수 있는 창이 나타납니다.



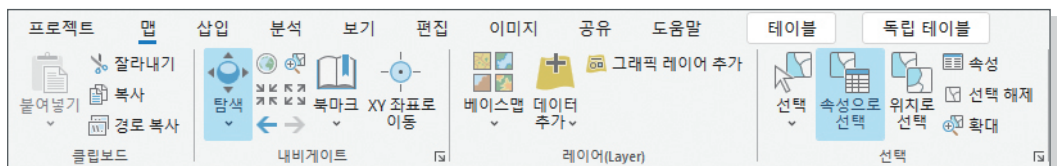
- ▶ ‘총인구’ 통계를 선별하기 위해 편집할 수 있는 테이블 복사본을 생성하겠습니다. **테이블 내보내기** 에서 **결과 테이블** 을 **c2023년_인구_다사_1K** 등으로 수정 입력하고 확인을 선택합니다.



- ▶ 생성한 테이블이 **독립형 테이블** 목록에 추가되었습니다. **열기** (열기 아이콘)로 테이블 내용을 확인하면 **속성으로 선택** (속성으로 선택 아이콘) 버튼이 활성화된 것을 확인할 수 있습니다.



- ▶ **독립형 테이블** 이 선택된 상태에서 **맵** 메뉴에서 선택 그룹의 **속성으로 선택** (속성으로 선택 아이콘)을 선택할 수도 있습니다.



- ▶ **속성으로 선택** 창에서 **통계항목** 을 '총인구'를 의미하는 **to_in_001** 로 설정하고 확인을 선택합니다.

속성으로 선택

입력 행: c2023년_인구_다사_1K

선택 유형: 새 선택

식:
 [통계항목] 이(가) [to_in_001] [다음과 같음(is equal to)]

+ 절 추가

☐ Where 절 반전

[적용] [확인]

- ▶ **독립된 테이블** 목록에 있는 **c2023년_인구_다사_1K** 테이블을 마우스 오른쪽 버튼으로 클릭해서 **데이터** , **테이블 내보내기** 를 차례로 선택합니다. **테이블 내보내기** 창이 나타나면 **결과 테이블** 항목을 **c2023년_인구_다사_1K_총인구** 등으로 수정 입력하고 확인을 선택합니다.

테이블 내보내기

매개변수 환경 설정

입력 테이블: c2023년_인구_다사_1K

☒ 선택한 레코드 사용: 7,024

결과 테이블: c2023년_인구_다사_1K_총인구

> 필터
 > 필드
 > 정렬

[확인]

- ▶ **독립형 테이블** 목록에서 추가된 **c2023년_인구_다사_1K_총인구** 테이블을 열어보면 **통계항목** 이 모두 **to_in_001** 인 것을 확인할 수 있습니다. **격자코드** 는 격자경계와 조인할 때 활용됩니다.

OBJECTID	기준연도	격자코드	통계항목	통계값
1	2023	다사0038	to_in_001	118
2	2023	다사0039	to_in_001	293
3	2023	다사0040	to_in_001	246
4	2023	다사0041	to_in_001	93
5	2023	다사0048	to_in_001	218

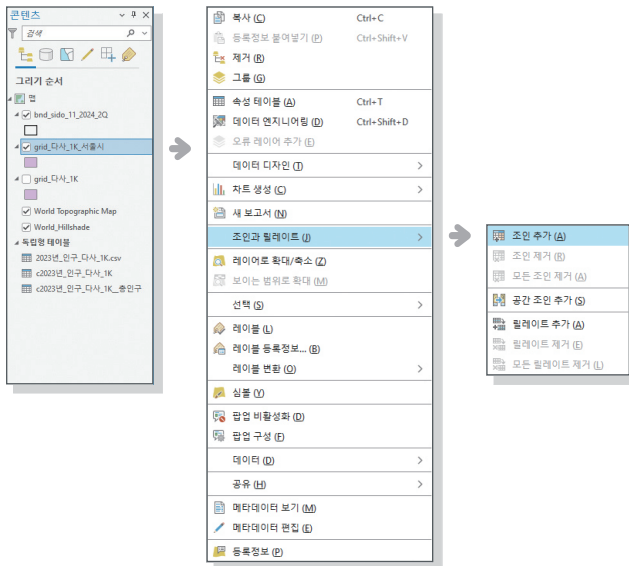
필터: 0/7,024 선택됨

5 경계와 통계 조인 및 시각화하기

- ▶ **콘텐츠** 창에서 **grid_다사_1K_서울시** 를 선택하고 **Ctrl+T** 를 눌러서 속성을 확인하겠습니다. GRID_1K_CD 속성에 격자코드가 저장되어 있습니다.

OBJECTID	Shape	GRID_1K_CD	Shape_Length	Shape_Area
1	폴리곤	다사3550	4000	1000000
2	폴리곤	다사3551	4000	1000000
3	폴리곤	다사3552	4000	1000000
4	폴리곤	다사3649	4000	1000000
5	폴리곤	다사3650	4000	1000000

- ▶ **grid_다사_1K_서울시** 를 마우스 오른쪽 버튼으로 클릭하고 **조인과 릴레이트** , **조인 추가** 메뉴를 차례로 선택합니다.



- ▶ **조인 추가** 창에서 **입력 필드** 에 **GRID_1K_CD** , **조인 테이블** 에 **c2023년_인구_다사_1K_총인구** , **필드 조인** 에 **격자코드** 를 각각 선택합니다.

조인 추가(Add Join)

입력 테이블: grid_다사_1K_서울시

입력 필드: GRID_1K_CD

조인 테이블: c2023년_인구_다사_1K_총인구

필드 조인: 격자코드

☒ 입력 레코드 모두 유지
☐ 색인 조인 필드

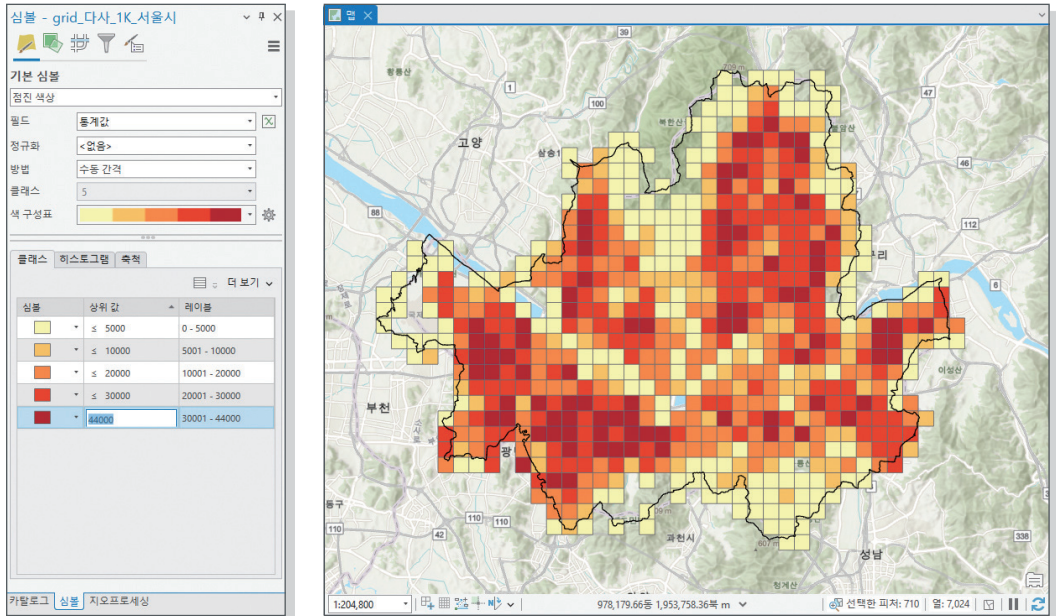
조인 유효성 검사

확인

- ▶ **grid_다사_1K_서울시** 속성을 다시 확인하겠습니다. **통계값** 을 포함하여 조인된 속성이 추가된 것을 확인할 수 있습니다.

OBJECTID	Shape	GRID_1K_CD	Shape_Length	Shape_Area	OBJECTID	기준연도	격자코드	통계항목	통계값
1	폴리곤	다사3550	4000	1000000	1529	2023	다사3550	to_in_001	0
2	폴리곤	다사3551	4000	1000000	1530	2023	다사3551	to_in_001	13
3	폴리곤	다사3552	4000	1000000	1531	2023	다사3552	to_in_001	0
4	폴리곤	다사3649	4000	1000000	1608	2023	다사3649	to_in_001	309
5	폴리곤	다사3650	4000	1000000	<없음>	<없음>	<없음>	<없음>	<없음>
6	폴리곤	다사3651	4000	1000000	1609	2023	다사3651	to_in_001	0
7	폴리곤	다사3652	4000	1000000	1610	2023	다사3652	to_in_001	5

- ▶ **grid_다사_1K_서울시** 를 마우스 오른쪽 버튼으로 클릭해서 **심볼** (📁) 메뉴를 선택합니다. 심볼 종류에 **점진 색상** , 필드에 **통계값** , 방법은 **네츄럴 브레이크(Jenks)** 또는 **수동 간격** 을 선택합니다.
- ▶ 색 구성표는 콤보 박스에서 **모두 표시** 를 체크하면 다양한 색상을 선택할 수 있고 클래스 탭에서 상위 값의 숫자 부분을 더블 클릭하면 원하는 간격으로 조정할 수 있습니다.





SGIS⁺plus
통계지리정보서비스

| 문의사항 | 042-481-2246, 2517(자료제공)
<https://sgis.kostat.go.kr/>