API-231 / GIS-PubPol
Meeting 11 (Lab Exercise + Problem Set 6)

Yuri M. Zhukov
Visiting Associate Professor of Public Policy
Harvard Kennedy School

March 7, 2024

Goal: integrate spatially-misaligned data

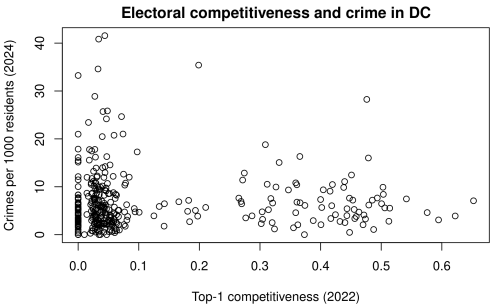


Figure 1: We'll make this

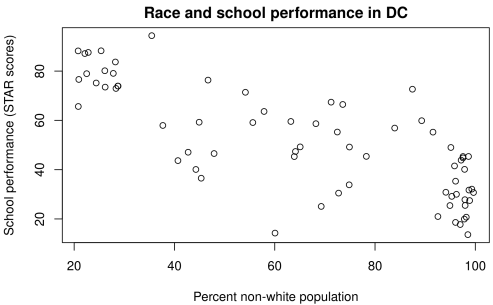


Figure 2: You'll make this

Overview of lab exercise and problem set

1. Lab exercise
    a) Use areal interpolation to estimate per capita crime in DC electoral districts
    b) Create a scatterplot of electoral competitiveness and per capita crime in DC
2. Problem set
    a) Use areal interpolation to estimate racial makeup of DC school zones
    b) Create a scatterplot of racial disparities in school performance in DC

Use case: different data are collected for different spatial units

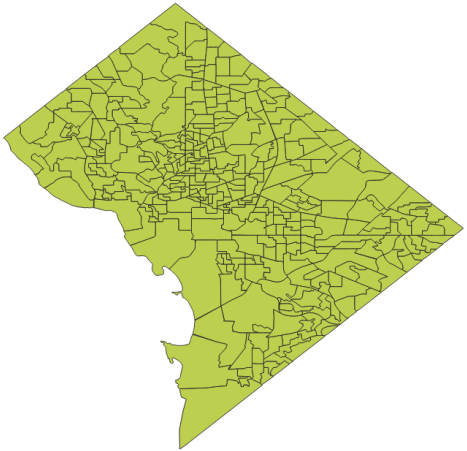


Figure 3: Elections @ single-member districts

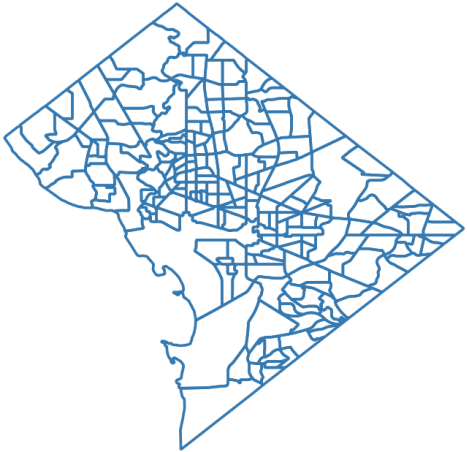


Figure 4: Population counts @ census tracts

Solution: change the support of the data $\rightarrow$ single member districts

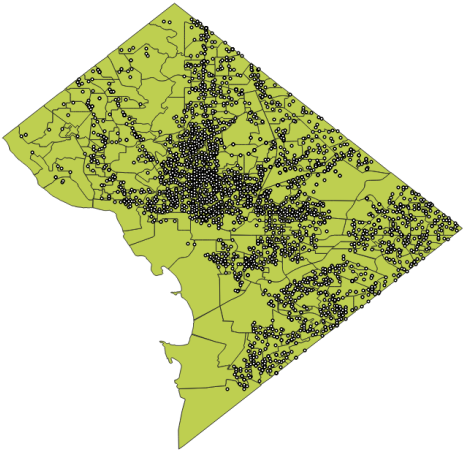


Figure 5: Count crimes in SMDs

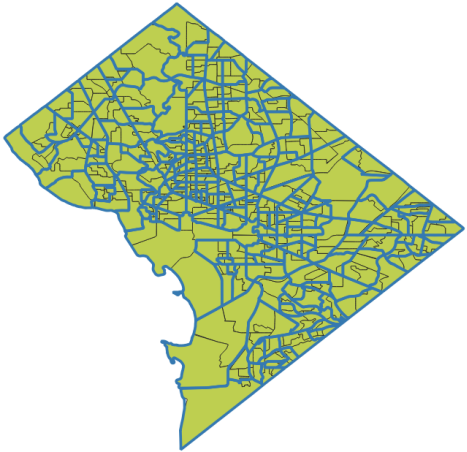


Figure 6: Interpolate population to SMDs

Analyze relationship between electoral competitiveness and per capita crime in SMDs

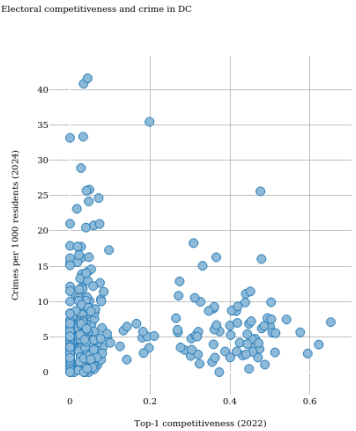


Figure 7: Scatterplot

Scatterplots: plots displaying data as points, with one variable on the horizontal axis and another on the vertical axis

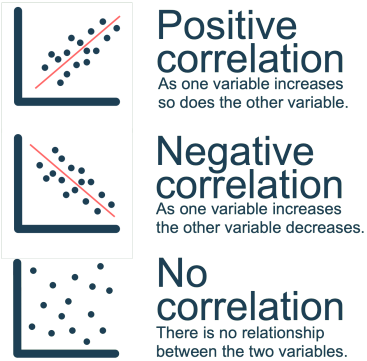


Figure 8: Scatterplots are useful to assess direction/strength of correlations

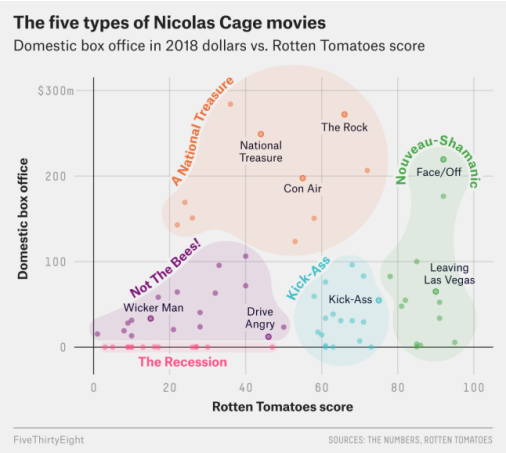


Figure 9: . . . and to identify outliers

The problem set will investigate a similar use case

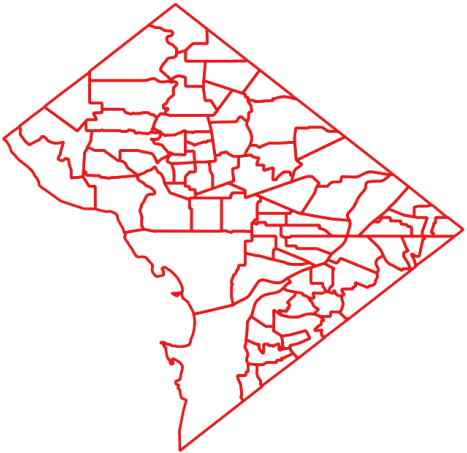


Figure 10: Scores @ school attendance zones

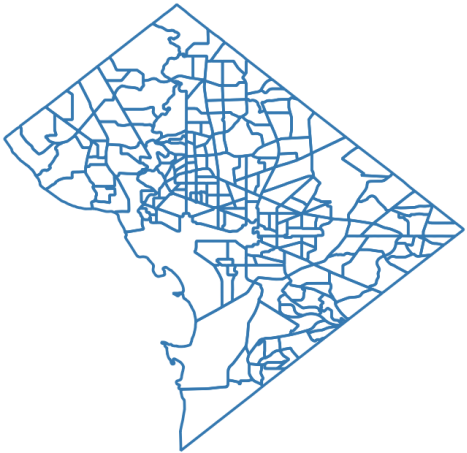


Figure 11: Demographics @ census tracts

Solution: change the support of the data → school attendance zones

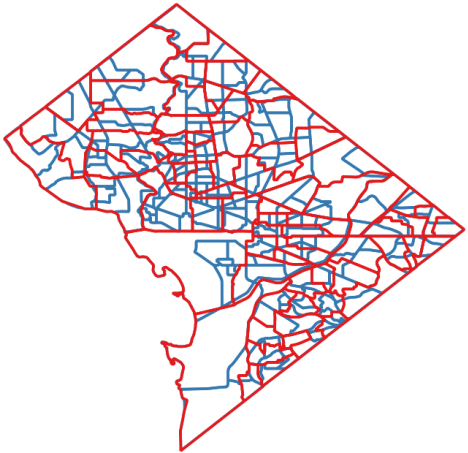


Figure 12: Interpolate demographics to SAZs

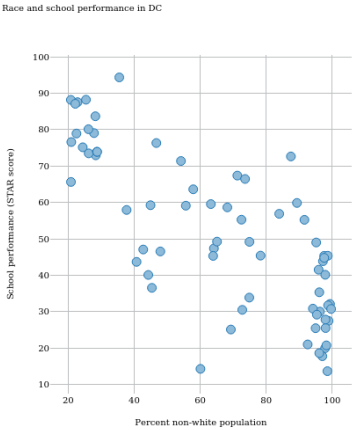


Figure 13: Scatterplot

You can make these plots in QGIS or in R. Instructions for both are below.

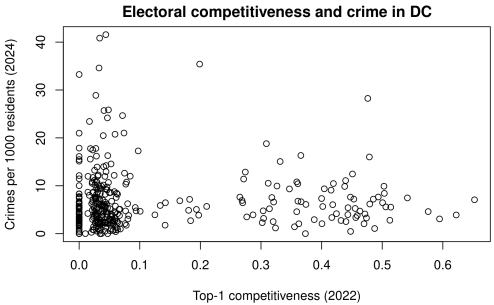


Figure 14: Scatterplot 1 in R

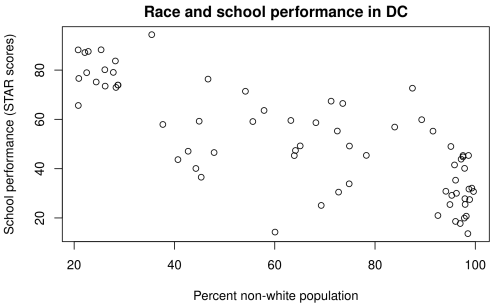


Figure 15: Scatterplot 2 in R

We have four (and a half) sources of data:

Category	Type	Format	Data source
Single member districts	Vector (polygon)	`.geojson`	`DC OpenData`
ACS DC Census Tract	Vector (polygon)	`.geojson`	`DC OpenData`
Crime Incidents in 2024	Vector (point)	`.geojson`	`DC OpenData`
School Attendance Zones, Elementary	Vector (polygon)	`.geojson`	`DC OpenData`
+ School STAR Scores	Table (non-spatial)	`.csv`	`DC OpenData`

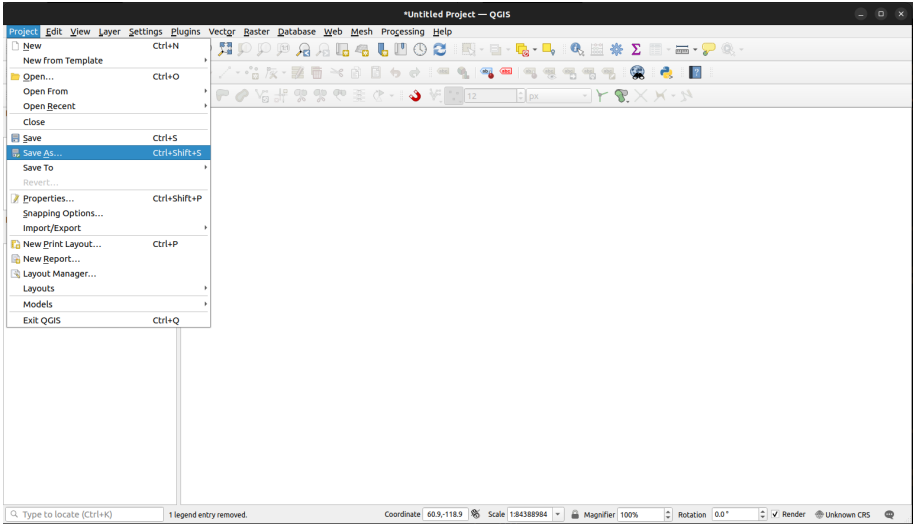
These are all in the `PS07.zip` file posted on Canvas.

Let's open QGIS...

QGIS

Always save your progress!

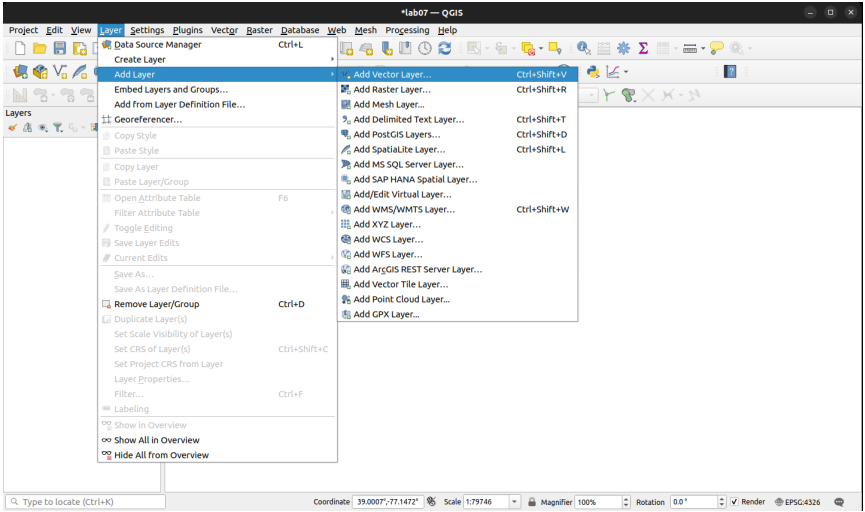
Go to `Project` → `Save As...`



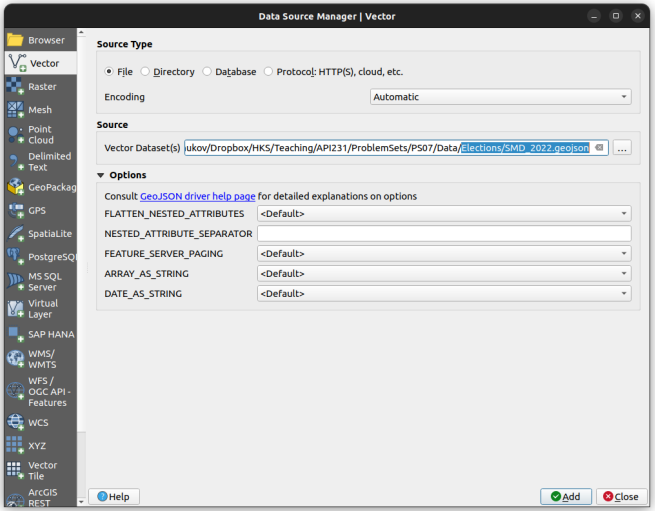
Interpolation

Load the Single Member Districts file:

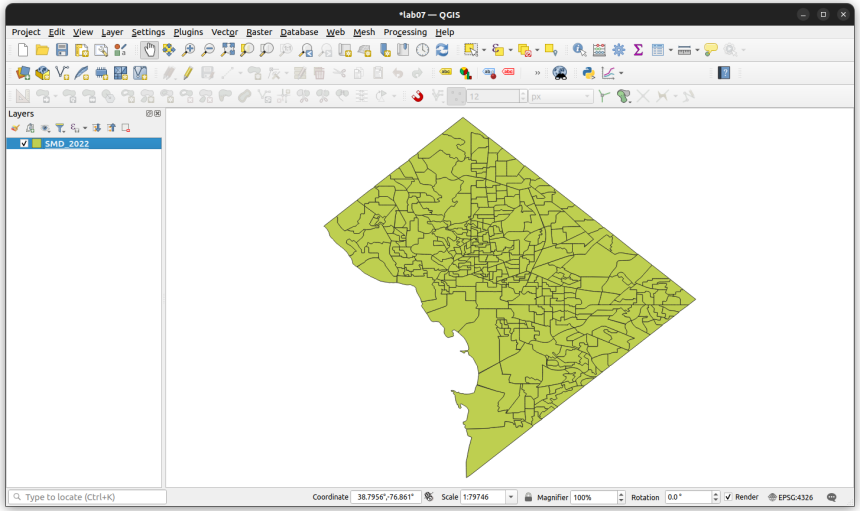
`Layer` → `Add Layer` → `Add Vector Layer...`



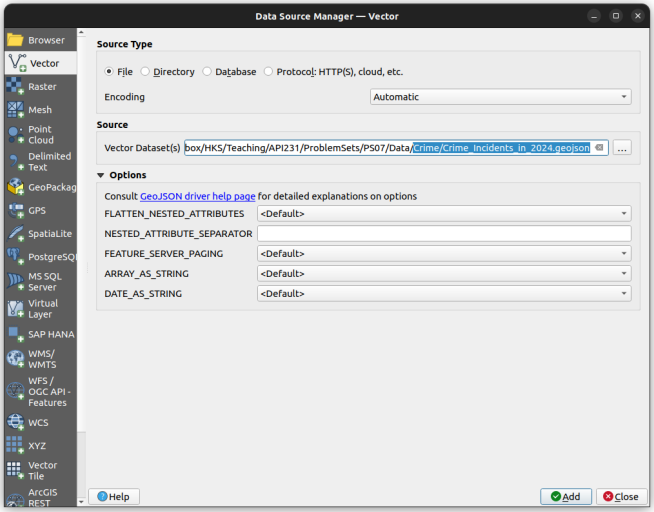
Navigate to the `SMD_2022.geojson` file in the `Data/Elections` directory:



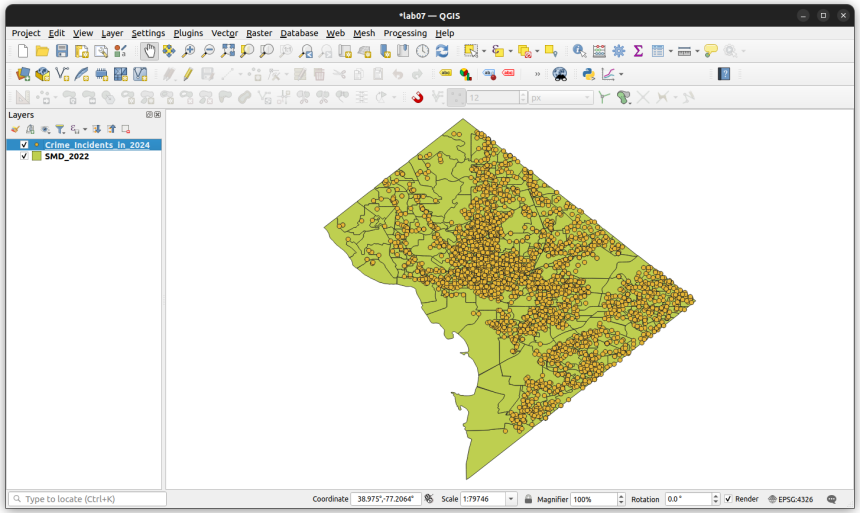
The SMD_2022 layer should appear in you project window. These polygons are the lowest-level electoral units in DC (DC has 8 Wards, 46 Advisory Neighborhood Commissions, 345 SMDs)



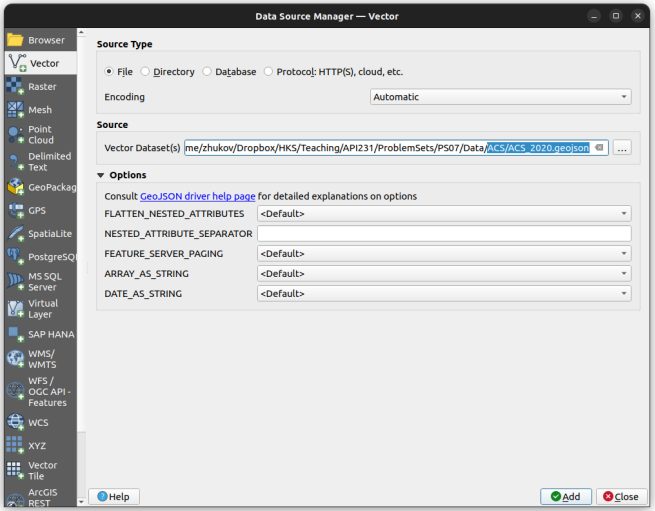
Load the crime incidents vector data:
Navigate to the `Crime_Incidents_in_2024.geojson` file in `Data/Crime`



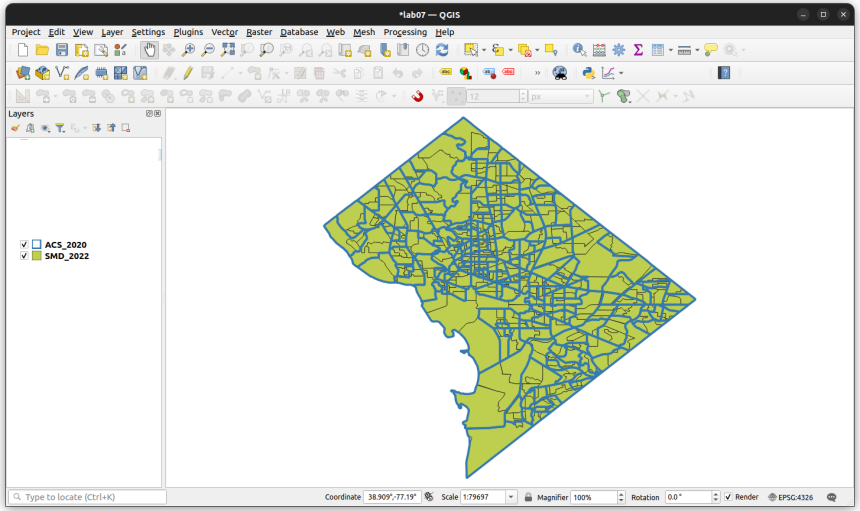
The `Crime_Incidents_in_2024` layer should appear in the project window. These are points, representing the locations of reported crimes



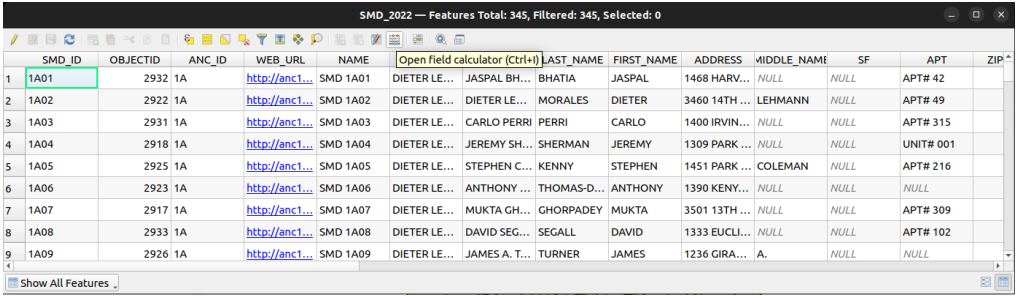
The third set of data we need is the `ACS_2020.geojson` file in `Data/ACS` folder



This layer contains American Community Survey data on local demographics (including population counts, which we need to estimate per capita crime). These data are at the level of Census Tracts, the borders of which do not align with SMDs

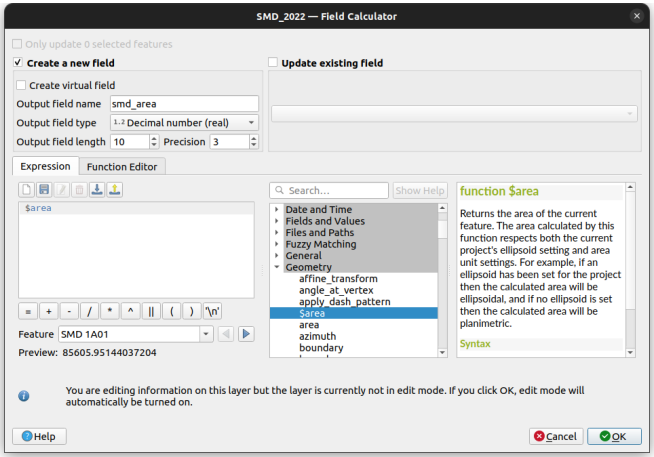


Our first order of business is to *construct the area weights* we will be using for interpolation, starting with a_j (area of destination polygons). Open the Attribute Table for `SMD_2022` (destination layer) and open the Field Calculator

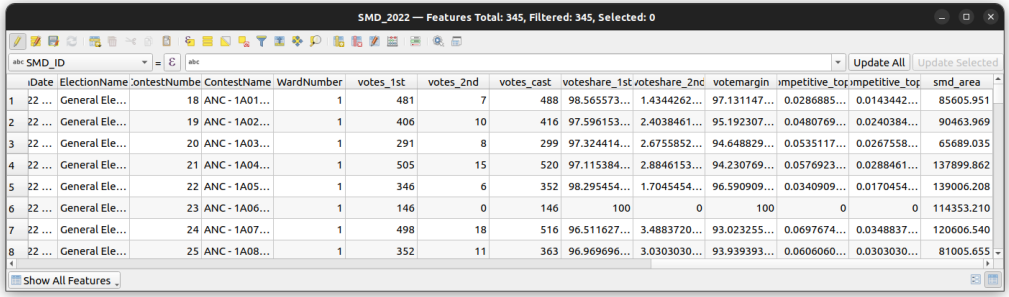


	SMD_ID	OBJECTID	ANC_ID	WEB_URL	NAME		LAST_NAME	FIRST_NAME	ADDRESS	MIDDLE_NAME	SF	APT	ZIP
1	1A01	2932	1A	http://anc1...	SMD 1A01	DIETER LE...	JASPAL BH...	BHATIA	JASPAL	1468 HARV...	NULL	NULL	APT# 42	
2	1A02	2922	1A	http://anc1...	SMD 1A02	DIETER LE...	DIETER LE...	MORALES	DIETER	3460 14TH ...	LEHMANN	NULL	APT# 49	
3	1A03	2931	1A	http://anc1...	SMD 1A03	DIETER LE...	CARLO PERRI	PERRI	CARLO	1400 IRVIN...	NULL	NULL	APT# 315	
4	1A04	2918	1A	http://anc1...	SMD 1A04	DIETER LE...	JEREMY SH...	SHERMAN	JEREMY	1309 PARK ...	NULL	NULL	UNIT# 001	
5	1A05	2925	1A	http://anc1...	SMD 1A05	DIETER LE...	STEPHEN C...	KENNY	STEPHEN	1451 PARK ...	COLEMAN	NULL	APT# 216	
6	1A06	2923	1A	http://anc1...	SMD 1A06	DIETER LE...	ANTHONY ...	THOMAS-D...	ANTHONY	1390 KENY...	NULL	NULL	NULL	
7	1A07	2917	1A	http://anc1...	SMD 1A07	DIETER LE...	MUKTA GH...	GHORPADEY	MUKTA	3501 13TH ...	NULL	NULL	APT# 309	
8	1A08	2933	1A	http://anc1...	SMD 1A08	DIETER LE...	DAVID SEG...	SEGALL	DAVID	1333 EUCLI...	NULL	NULL	APT# 102	
9	1A09	2926	1A	http://anc1...	SMD 1A09	DIETER LE...	JAMES A. T...	TURNER	JAMES	1236 GIRA...	A.	NULL	NULL	

Create a new field called `smd_area` of type `Decimal number (real)`.
For the Expression, type `$area`.
This will give us the area calculation for destination polygons (in square kilometers)

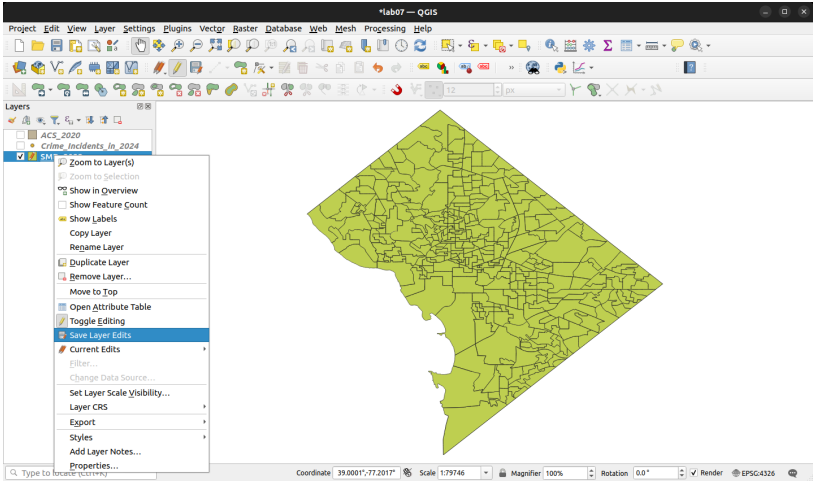


The new field `smd_area` should appear in the Attribute Table for `SMD_2022`



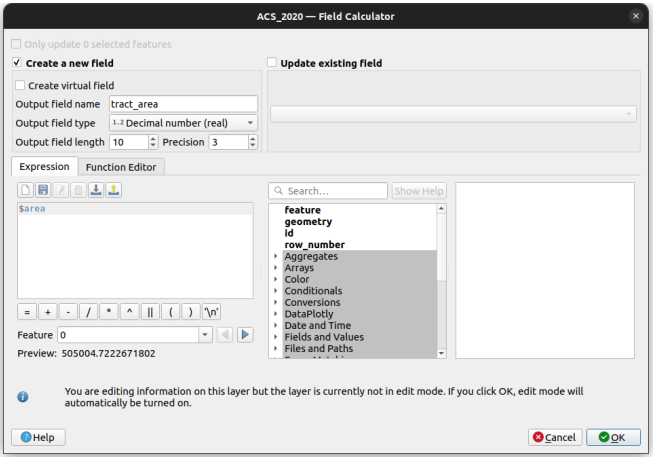
	Date	ElectionName	ContestNumbe	ContestName	WardNumber	votes_1st	votes_2nd	votes_cast	voteshare_1st	voteshare_2nc	votemargin	mpetitive_top	mpetitive_top	smd_area
1	22 ...	General Ele...	18	ANC - 1A01...	1	481	7	488	98.565573...	1.4344262...	97.131147...	0.0286885...	0.0143442...	85605.951
2	22 ...	General Ele...	19	ANC - 1A02...	1	406	10	416	97.596153...	2.4038461...	95.192307...	0.0480769...	0.0240384...	90463.969
3	22 ...	General Ele...	20	ANC - 1A03...	1	291	8	299	97.324414...	2.6755852...	94.648829...	0.0535117...	0.0267558...	65689.035
4	22 ...	General Ele...	21	ANC - 1A04...	1	505	15	520	97.115384...	2.8846153...	94.230769...	0.0576923...	0.0288461...	137899.862
5	22 ...	General Ele...	22	ANC - 1A05...	1	346	6	352	98.295454...	1.7045454...	96.590909...	0.0340909...	0.0170454...	139006.208
6	22 ...	General Ele...	23	ANC - 1A06...	1	146	0	146	100	0	100	0	0	114353.210
7	22 ...	General Ele...	24	ANC - 1A07...	1	498	18	516	96.511627...	3.4883720...	93.023255...	0.0697674...	0.0348837...	120606.540
8	22 ...	General Ele...	25	ANC - 1A08...	1	352	11	363	96.969696...	3.0303030...	93.939393...	0.0606060...	0.0303030...	81005.655

Remember to save your layer edits after every operation like this! Right-click on `SMD_2022` → `Save Layer Edits` (simply entering Ctrl-S or Cmd-S will save the project file, but not the underlying data layers we are editing)



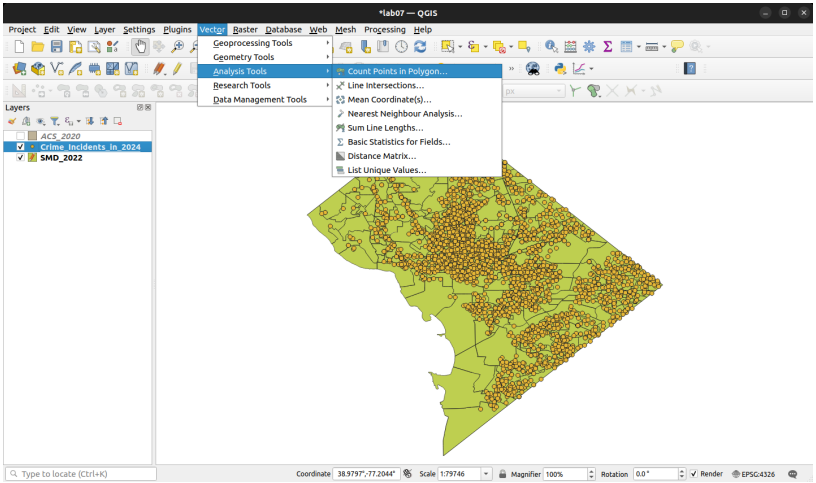
Let's also calculate the area of source polygons (a_i). Open the Field Calculator for `ACS_2020` and create a new field called `tract_area` of type `Decimal number (real)`.

For the Expression, type `$area`.

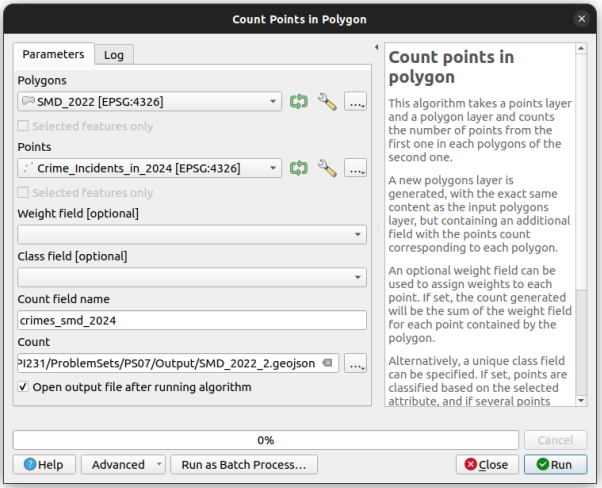


Let's calculate the number of crimes in each SMD (the "nominator" of our future interpolated variable).

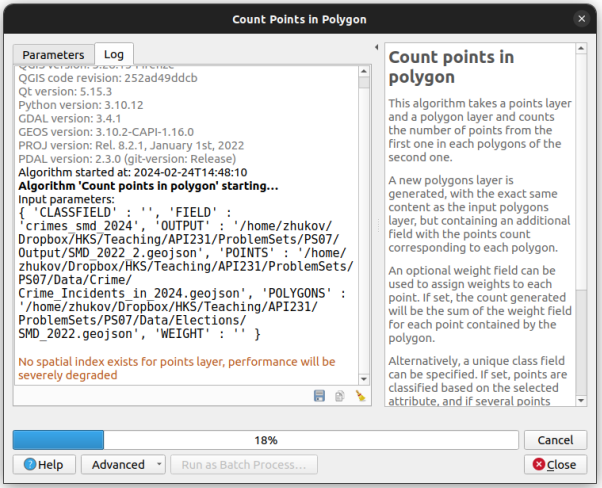
Go to `Vector` menu → `Analysis Tools` → `Count Points in Polygon...`



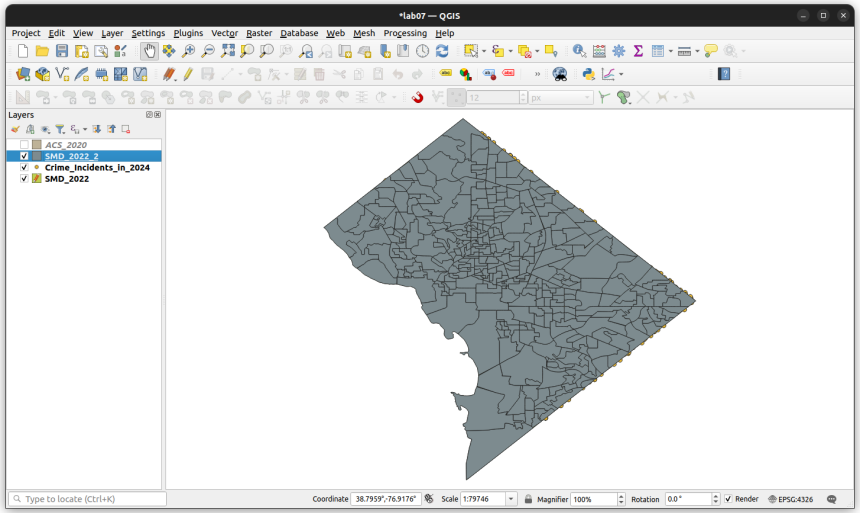
Select `SMD_2022` as the Polygons layer and `Crime_Incidents_in_2024` as the Points layer. Name the count field `crimes_smd_2024` and save the output to a new file called `SMD_2022_2.geojson`. Click `Run`



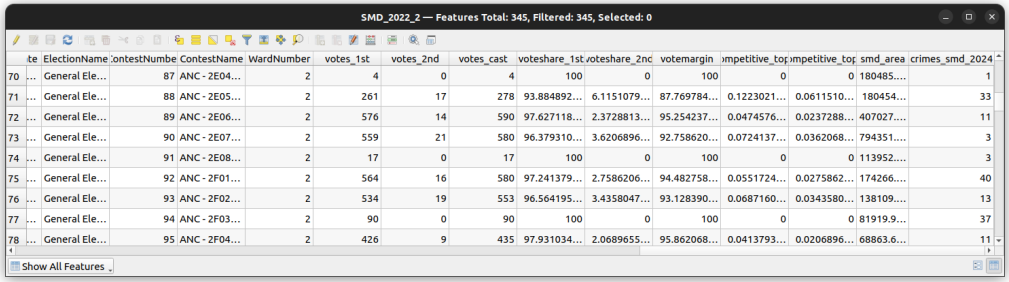
You may see a `No spatial index...` warning message in the log, but this won't affect the calculation (only the processing speed). Just be patient and wait for the algorithm to finish



The newly-created SMD_2022_2 layer should appear in the project window. You can hide the other two layers for now, if you like

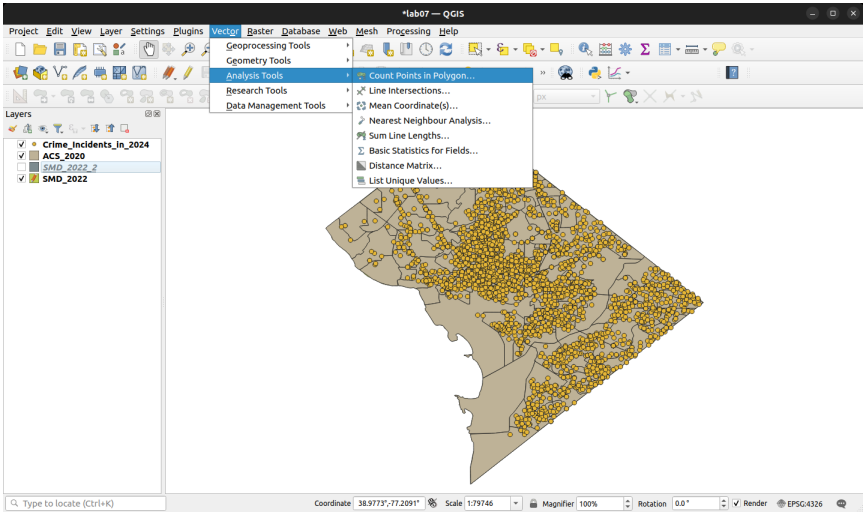


Check the Attribute Table for `SMD_2022_2` to make sure the `crimes_smd_2024` variable is there

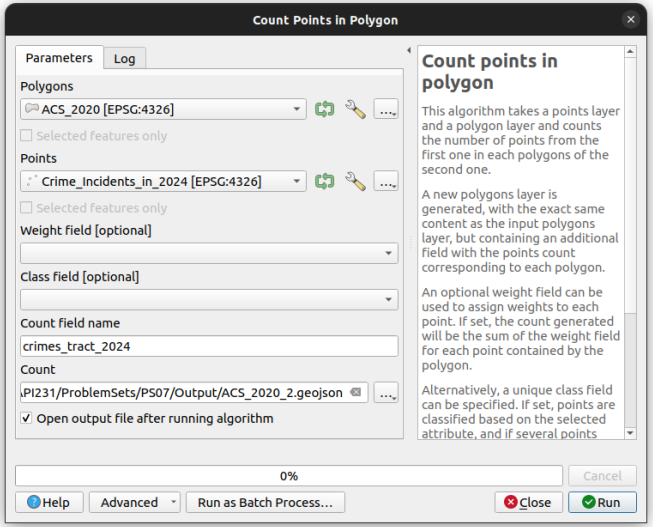


	te	ElectionName	ontestNumbe	ContestName	WardNumber	votes_1st	votes_2nd	votes_cast	voteshare_1st	voteshare_2nc	votemargin	mpetitive_top	mpetitive_top	smd_area	crimes_smd_2024
70	...	General Ele...	87	ANC - 2E04...	2	4	0	4	100	0	100	0	0	180485....	1
71	...	General Ele...	88	ANC - 2E05...	2	261	17	278	93.884892...	6.1151079...	87.769784...	0.1223021...	0.0611510...	180454....	33
72	...	General Ele...	89	ANC - 2E06...	2	576	14	590	97.627118...	2.3728813...	95.254237...	0.0474576...	0.0237288...	407027....	11
73	...	General Ele...	90	ANC - 2E07...	2	559	21	580	96.379310...	3.6206896...	92.758620...	0.0724137...	0.0362068...	794351....	3
74	...	General Ele...	91	ANC - 2E08...	2	17	0	17	100	0	100	0	0	113952....	3
75	...	General Ele...	92	ANC - 2F01...	2	564	16	580	97.241379...	2.7586206...	94.482758...	0.0551724...	0.0275862...	174266....	40
76	...	General Ele...	93	ANC - 2F02...	2	534	19	553	96.564195...	3.4358047...	93.128390...	0.0687160...	0.0343580...	138109....	13
77	...	General Ele...	94	ANC - 2F03...	2	90	0	90	100	0	100	0	0	81919.9...	37
78	...	General Ele...	95	ANC - 2F04...	2	426	9	435	97.931034...	2.0689655...	95.862068...	0.0413793...	0.0206896...	68863.6...	11

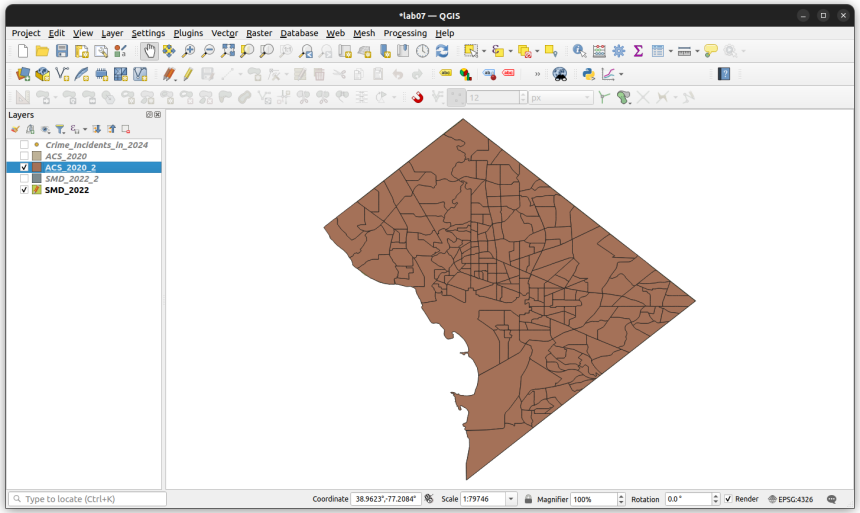
To facilitate a "whole vs. parts" comparison (i.e. transforming a variable directly versus reconstructing it from transformed components) we can do the same kind of points-in-polygons calculation with the ACS_2020 layer...



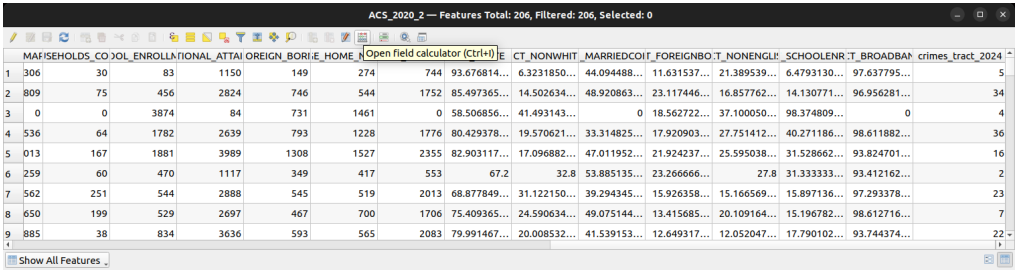
Same approach as before, except with `ACS_2020` as polygon layer, `crimes_tract_2024` as field name, and `ACS_2020_2.geojson` as output file



The new ACS_2020_2 layer should appear in the project window.
Let's open its Attribute Table

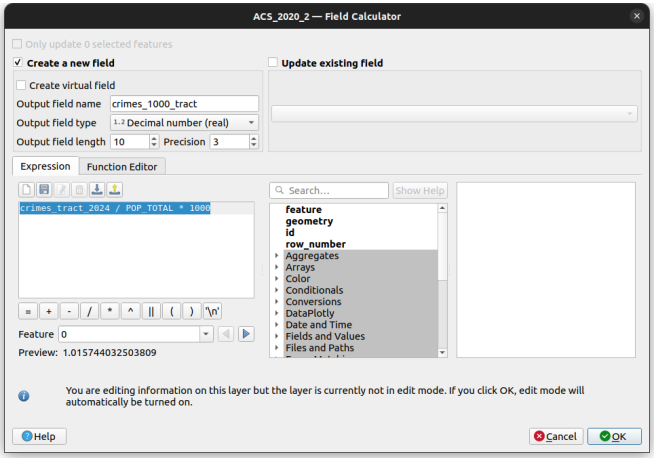


Make sure the `crimes_tract_2024` variable is there.
Then launch the Field Calculator

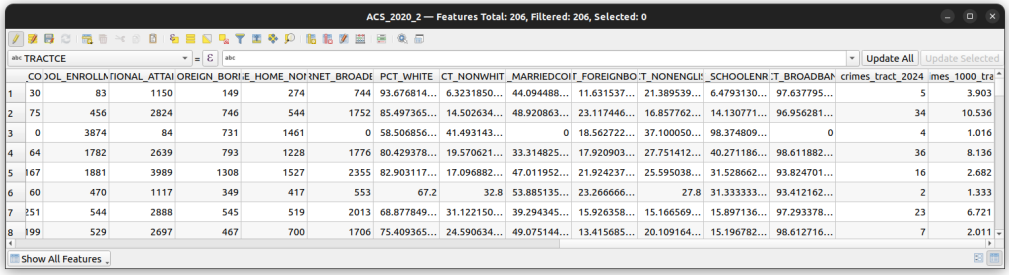


	MAR	SEHOLDS_CO	OL_ENROLLM	TIONAL_ATTAI	OREIGN_BORI	E_HOME_N	E	CT_NONWHIT	_MARRIEDCOU	T_FOREIGNBO	T_NONENGLI	_SCHOOLENR	T_BROADBAN	crimes_tract_2024
1	306	30	83	1150	149	274	744	93.676814…	6.3231850…	44.094488…	11.631537…	21.389539…	6.4793130…	97.637795…	5
2	809	75	456	2824	746	544	1752	85.497365…	14.502634…	48.920863…	23.117446…	16.857762…	14.130771…	96.956281…	34
3	0	0	3874	84	731	1461	0	58.506856…	41.493143…	0	18.562722…	37.100050…	98.374809…	0	4
4	536	64	1782	2639	793	1228	1776	80.429378…	19.570621…	33.314825…	17.920903…	27.751412…	40.271186…	98.611882…	36
5	013	167	1881	3989	1308	1527	2355	82.903117…	17.096882…	47.011952…	21.924237…	25.595038…	31.528662…	93.824701…	16
6	259	60	470	1117	349	417	553	67.2	32.8	53.885135…	23.266666…	27.8	31.333333…	93.412162…	2
7	562	251	544	2888	545	519	2013	68.877849…	31.122150…	39.294345…	15.926358…	15.166569…	15.897136…	97.293378…	23
8	650	199	529	2697	467	700	1706	75.409365…	24.590634…	49.075144…	13.415685…	20.109164…	15.196782…	98.612716…	7
9	885	38	834	3636	593	565	2083	79.991467…	20.008532…	41.539153…	12.649317…	12.052047…	17.790102…	93.744374…	22

Calculate the "crimes per 1000 residents" variable for `ACS_2020_2`. Name the field `crimes_1000_tract`, set type to `Decimal number (real)`, and set the Expression to `crimes_tract_2024 / POP_TOTAL * 1000`

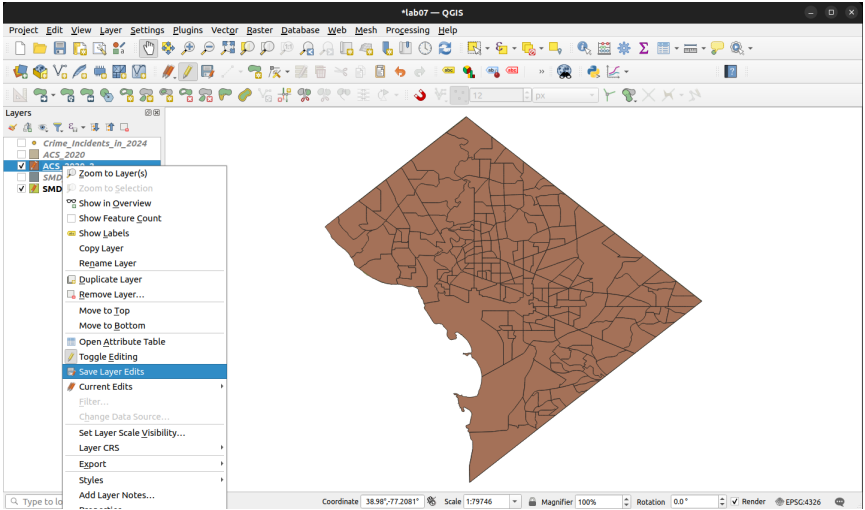


The new `crimes_1000_tract` variable should appear in the Attribute Table

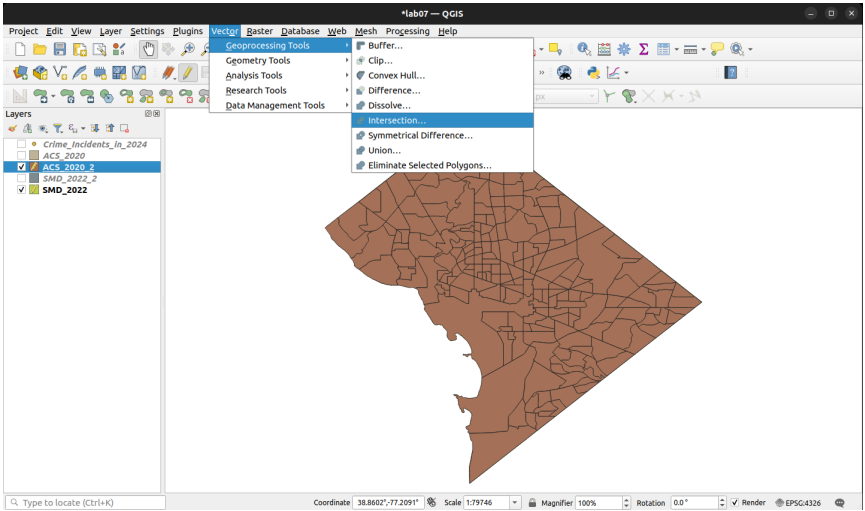


	CO	OL_ENROLLN	TIONAL_ATTAI	OREIGN_BORI	E_HOME_NOI	RNET_BROADE	PCT_WHITE	CT_NONWHIT	_MARRIEDCOI	T_FOREIGNBO	T_NONENGLI	_SCHOOLENR	T_BROADBAI	crimes_tract_2024	mes_1000_tra
1	30	83	1150	149	274	744	93.676814…	6.3231850…	44.094488…	11.631537…	21.389539…	6.4793130…	97.637795…	5	3.903
2	75	456	2824	746	544	1752	85.497365…	14.502634…	48.920863…	23.117446…	16.857762…	14.130771…	96.956281…	34	10.536
3	0	3874	84	731	1461	0	58.506856…	41.493143…	0	18.562722…	37.100050…	98.374809…	0	4	1.016
4	64	1782	2639	793	1228	1776	80.429378…	19.570621…	33.314825…	17.920903…	27.751412…	40.271186…	98.611882…	36	8.136
5	167	1881	3989	1308	1527	2355	82.903117…	17.096882…	47.011952…	21.924237…	25.595038…	31.528662…	93.824701…	16	2.682
6	60	470	1117	349	417	553	67.2	32.8	53.885135…	23.266666…	27.8	31.333333…	93.412162…	2	1.333
7	251	544	2888	545	519	2013	68.877849…	31.122150…	39.294345…	15.926358…	15.166569…	15.897136…	97.293378…	23	6.721
8	199	529	2697	467	700	1706	75.409365…	24.590634…	49.075144…	13.415685…	20.109164…	15.196782…	98.612716…	7	2.011

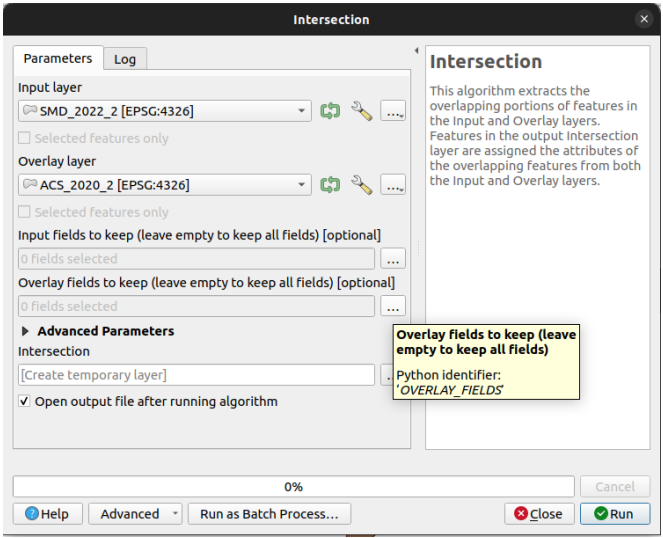
Remember to save your layer edits!



The next step is to calculate the area of each intersection between source and destination polygons, or $a_{i \cap j}$. To do this, we first need to create the intersection. Go to `Vector` menu → `Geoprocessing Tools` → `Intersection...`



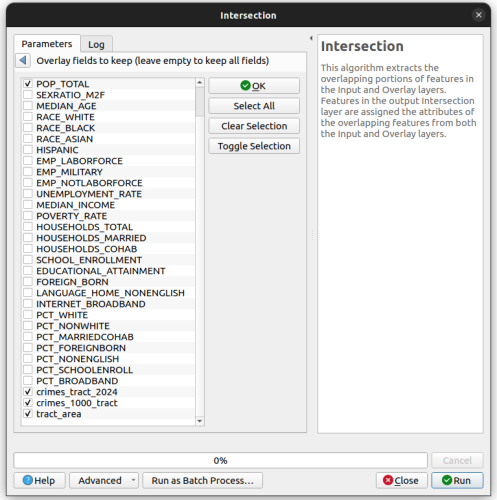
For the intersection, select `SMD_2022_2` as Input Layer, `ACS_2020_2` as Overlay layer. Then click the `[...]` button next to "Overlay fields to keep"



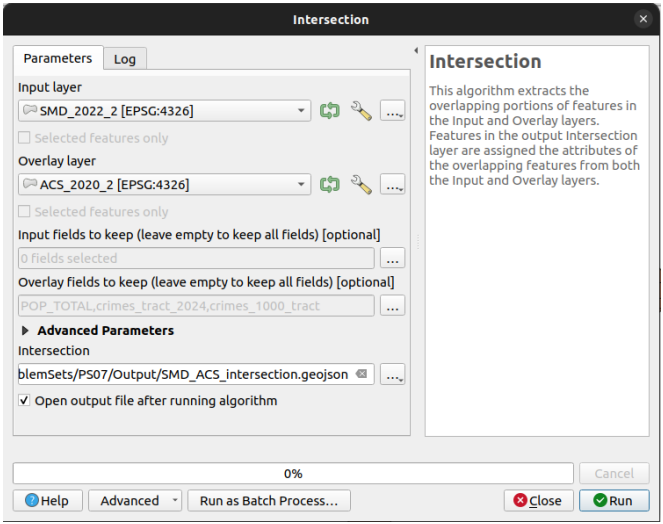
On the next screen check the boxes next to

- ✓ `POP_TOTAL`
- ✓ `crimes_tract_2024`
- ✓ `crimes_1000_tract`
- ✓ `tract_area`

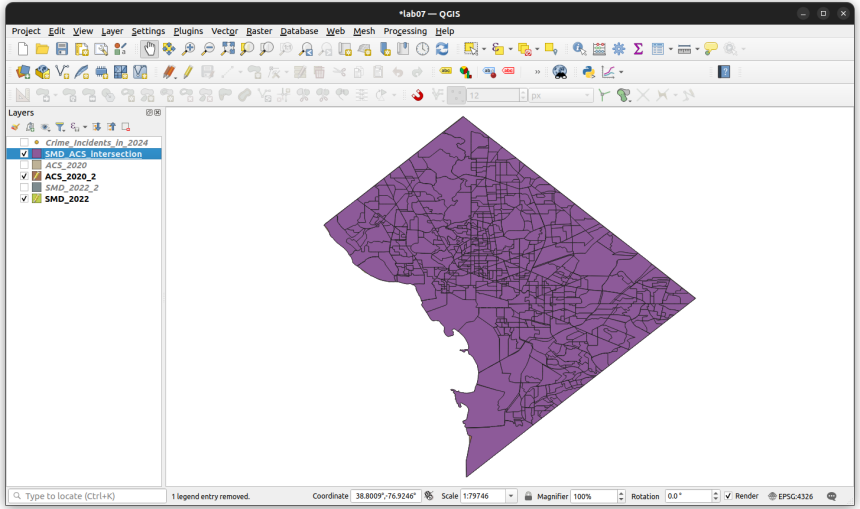
Click `OK`



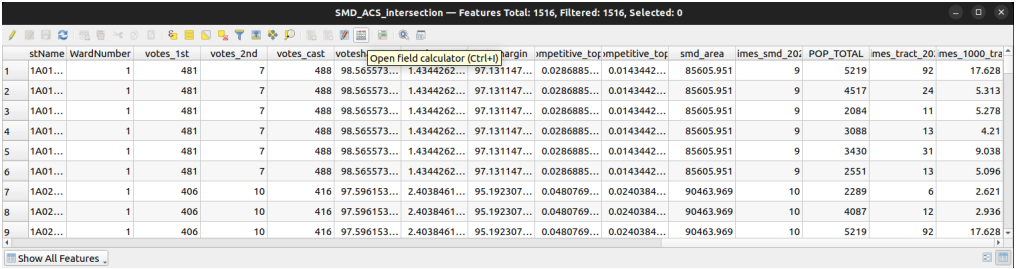
Save the output file to `SMD_ACS_intersection.geojson`, click `Run`



The `SMD_ACS_intersection` layer should appear in your project window

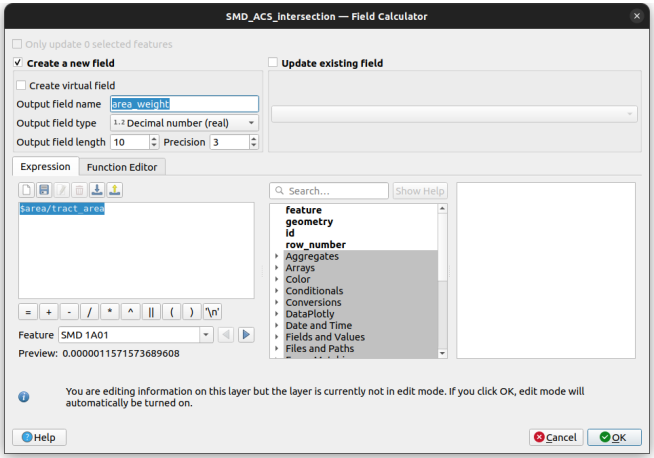


Open the Attribute Table for `SMD_ACS_intersection` and launch the Field Calculator

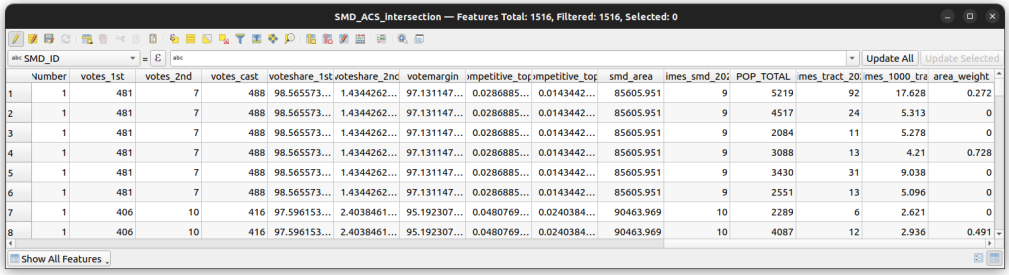


	stName	WardNumber	votes_1st	votes_2nd	votes_cast	votesh…	…argin	…mpetitive_top	…mpetitive_top	smd_area	…imes_smd_202	POP_TOTAL	…mes_tract_20…	…mes_1000_tra…
1	1A01…	1	481	7	488	98.565573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	5219	92	17.628
2	1A01…	1	481	7	488	98.565573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	4517	24	5.313
3	1A01…	1	481	7	488	98.565573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	2084	11	5.278
4	1A01…	1	481	7	488	98.565573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	3088	13	4.21
5	1A01…	1	481	7	488	98.565573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	3430	31	9.038
6	1A01…	1	481	7	488	98.565573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	2551	13	5.096
7	1A02…	1	406	10	416	97.596153…	2.4038461…	95.192307…	0.0480769…	0.0240384…	90463.969	10	2289	6	2.621
8	1A02…	1	406	10	416	97.596153…	2.4038461…	95.192307…	0.0480769…	0.0240384…	90463.969	10	4087	12	2.936
9	1A02…	1	406	10	416	97.596153…	2.4038461…	95.192307…	0.0480769…	0.0240384…	90463.969	10	5219	92	17.628

Create a new field called `area_weight` of type `Decimal Number (real)`.
Set Expression to `$area / tract_area` (this is equivalent to $w_{i \cap j}^{(\text{ext})} = \frac{a_{i \cap j}}{a_i}$)

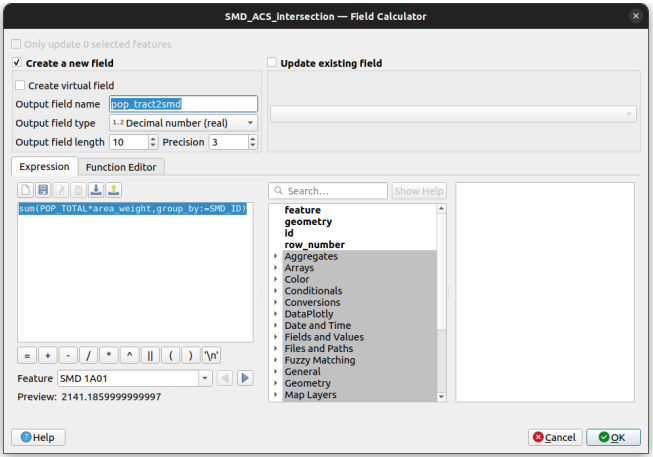


The `area_weight` field should appear in the Attribute Table.
Now go back to the Field Calculator, where we will create weighted population variables for destination polygons

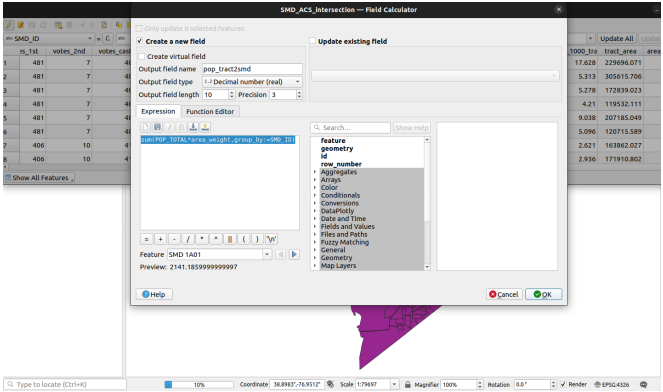


	Number	votes_1st	votes_2nd	votes_cast	voteshare_1st	voteshare_2nd	votemargin	mpetitive_top	mpetitive_top	smd_area	imes_smd_202	POP_TOTAL	mes_tract_20	mes_1000_tra	area_weight
1	1	481	7	488	98.565573...	1.4344262...	97.131147...	0.0286885...	0.0143442...	85605.951	9	5219	92	17.628	0.272
2	1	481	7	488	98.565573...	1.4344262...	97.131147...	0.0286885...	0.0143442...	85605.951	9	4517	24	5.313	0
3	1	481	7	488	98.565573...	1.4344262...	97.131147...	0.0286885...	0.0143442...	85605.951	9	2084	11	5.278	0
4	1	481	7	488	98.565573...	1.4344262...	97.131147...	0.0286885...	0.0143442...	85605.951	9	3088	13	4.21	0.728
5	1	481	7	488	98.565573...	1.4344262...	97.131147...	0.0286885...	0.0143442...	85605.951	9	3430	31	9.038	0
6	1	481	7	488	98.565573...	1.4344262...	97.131147...	0.0286885...	0.0143442...	85605.951	9	2551	13	5.096	0
7	1	406	10	416	97.596153...	2.4038461...	95.192307...	0.0480769...	0.0240384...	90463.969	10	2289	6	2.621	0
8	1	406	10	416	97.596153...	2.4038461...	95.192307...	0.0480769...	0.0240384...	90463.969	10	4087	12	2.936	0.491

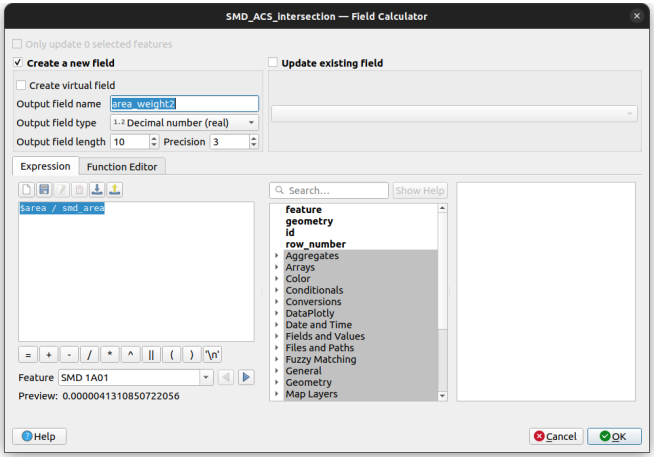
Now let's sum the weighted population numbers for each SMD. Create a new field called `pop_tract2smd` of type `Decimal Number (real)`. Set Expression to `sum(POP_TOTAL*area_weight,group_by:=SMD_ID)` (this is equivalent to $\sum_{i \cap j} w_{i \cap j}^{(ext)} x_{i \cap j}$)



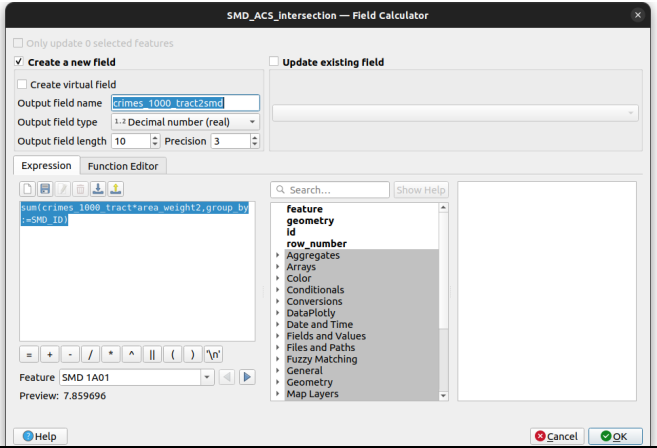
This will take several minutes to compute (you can follow the progress bar at bottom, or use this time to stretch your legs)



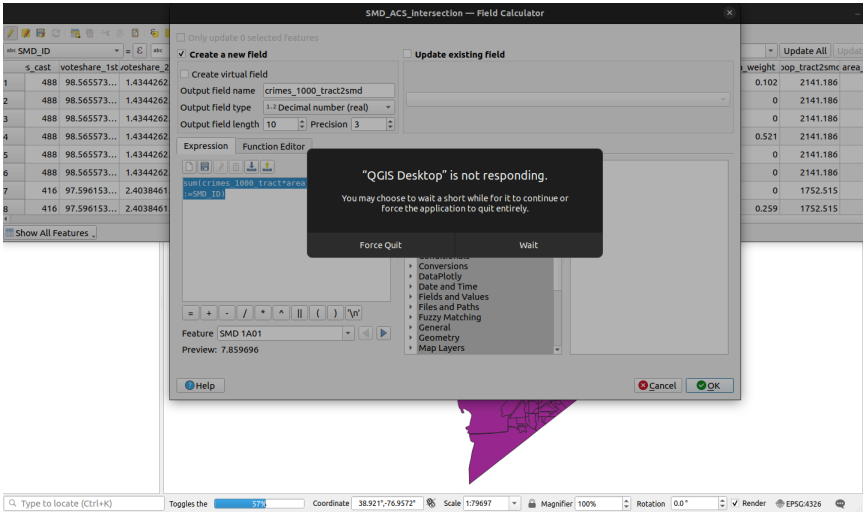
(OPTIONAL) Create a new field called `area_weight2` of type `Decimal Number (real)`.
Set Expression to `$area / smd_area` (this is equivalent to $w_{i \cap j}^{(\text{int})} = \frac{a_{i \cap j}}{a_j}$)



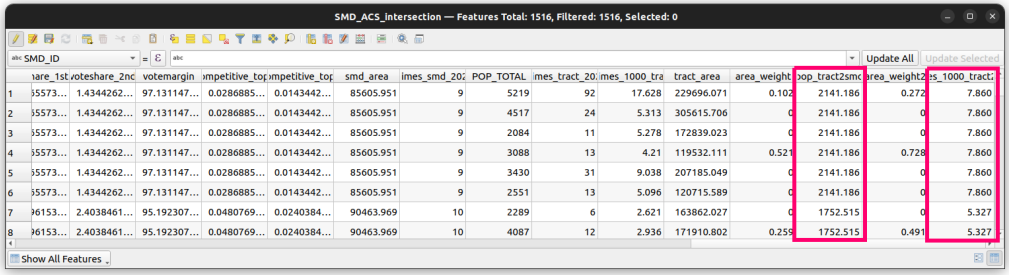
Now let's do a weighted average (intensive) of crimes per 1000 residents. Create a new field called `crimes_1000_tract2smd` of type `Decimal Number (real)`. Set Expression to `sum(crimes_1000_tract*area_weight2,group_by:=SMD_ID)` (this is equivalent to $\sum_{i \cap j} w_{i \cap j}^{(int)} x_{i \cap j}$)



(OPTIONAL) This will also take a while to run. I even received a "program not responding" message, which I ignored (but check the progress bar to make sure the program is actually still running)

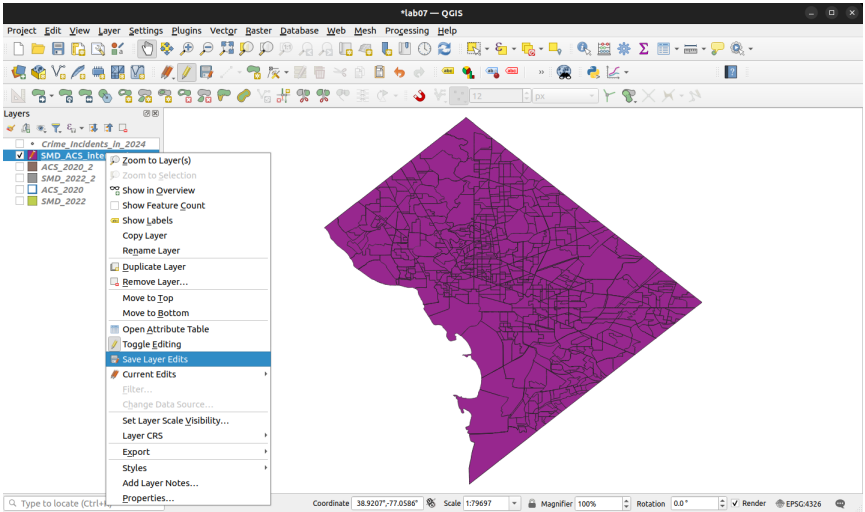


When it finishes running, you will see both new variables in the Attribute Table. Note that these values should be constant within SMDs (i.e. values will be repeated for intersections within each destination polygon)

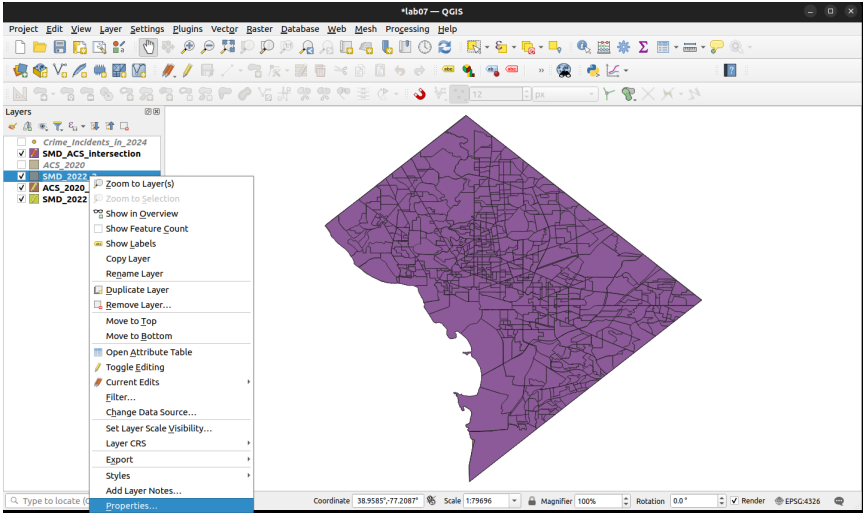


	are_1st	voteshare_2nd	votemargin	mpetitive_top	mpetitive_top	smd_area	imes_smd_202	POP_TOTAL	mes_tract_20	mes_1000_tra	tract_area	area_weight	pop_tract2smd	area_weight2	s_1000_tract2
1	55573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	5219	92	17.628	229696.071	0.102	2141.186	0.272	7.860
2	55573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	4517	24	5.313	305615.706	0	2141.186	0	7.860
3	55573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	2084	11	5.278	172839.023	0	2141.186	0	7.860
4	55573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	3088	13	4.21	119532.111	0.521	2141.186	0.728	7.860
5	55573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	3430	31	9.038	207185.049	0	2141.186	0	7.860
6	55573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	2551	13	5.096	120715.589	0	2141.186	0	7.860
7	96153…	2.4038461…	95.192307…	0.0480769…	0.0240384…	90463.969	10	2289	6	2.621	163862.027	0	1752.515	0	5.327
8	96153…	2.4038461…	95.192307…	0.0480769…	0.0240384…	90463.969	10	4087	12	2.936	171910.802	0.259	1752.515	0.491	5.327

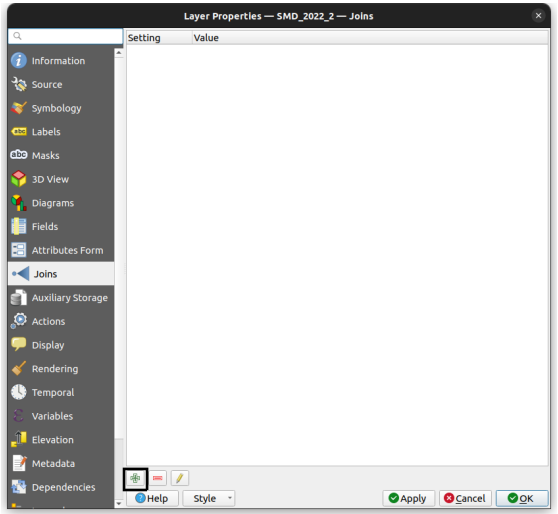
After the algorithm finishes running is a good time to save your layer edits!



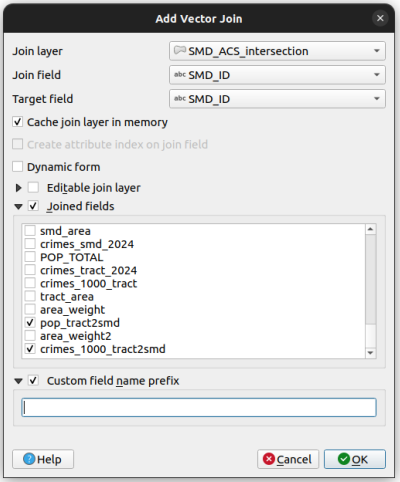
Our next step is to import these weighted values into the `SMD_2022_2` layer, and divide crimes by population size. Right-click on `SMD_2022_2` → `Properties`



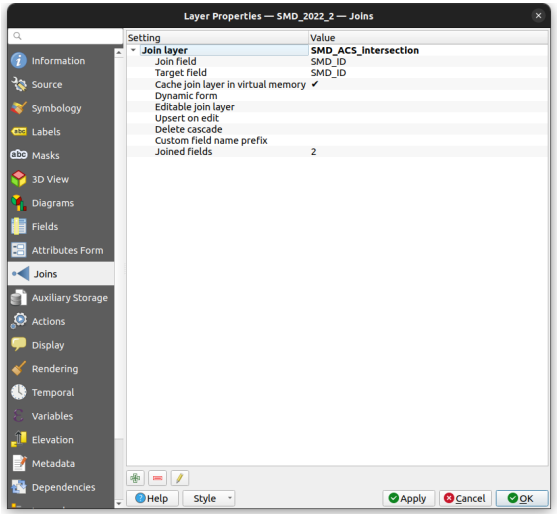
Go to `Joins` tab in `SMD_2022_2` Properties and click + button to add a new join



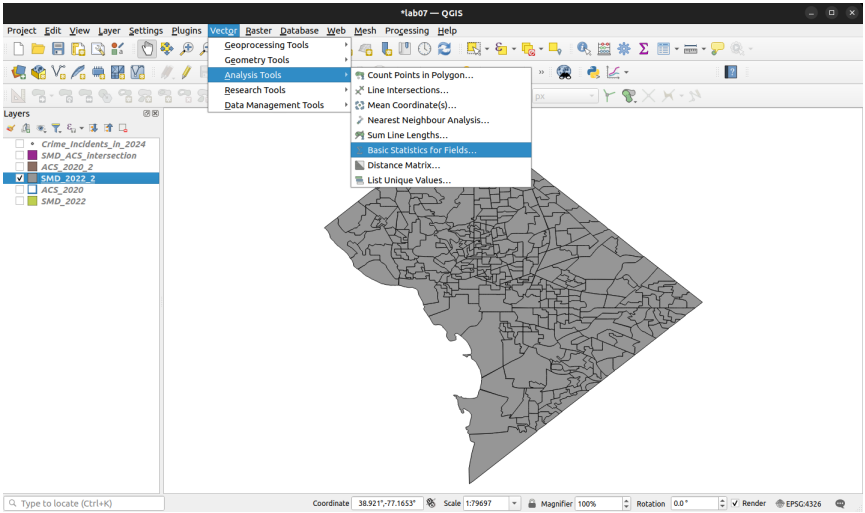
Set `SMD_ACS_intersection` as the "Join layer", `SMD_ID` as the "Join" and "Target" field, and under "Joined fields" select ✓ `pop_tract2smd` and ✓ `crimes_1000_tract2smd` (if you created this variable, too). Check the box next to `Custom field name prefix` and clear the contents (i.e. no prefix). Click `OK`



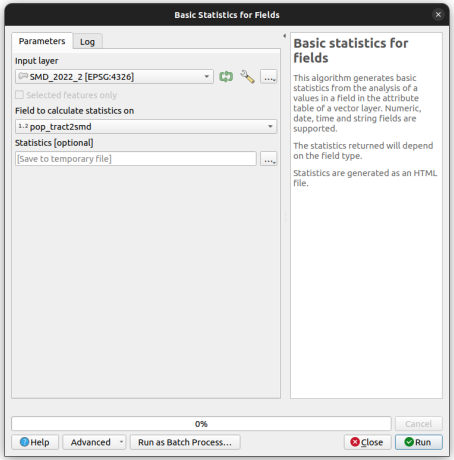
You should now see the new join layer in Properties



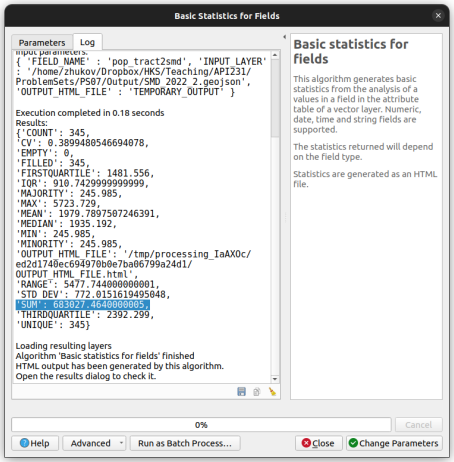
Quality check! Let's see if the sum of interpolated population values is close to the sum of this variable in the original data. Go to `Vector` menu → `Vector Analysis` → `Basic Statistics for Fields`



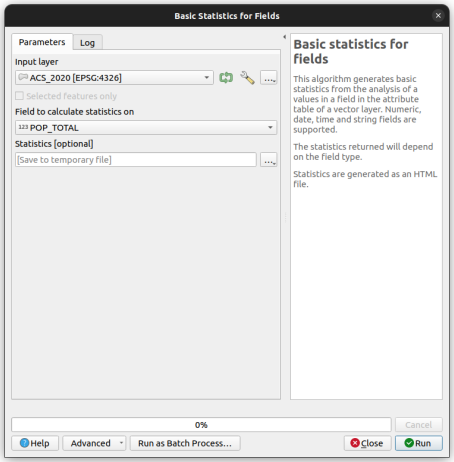
In the next window, set “Input layer” to `SMD_2022_2` and “Field” to `pop_tract2smd`. Click `Run`



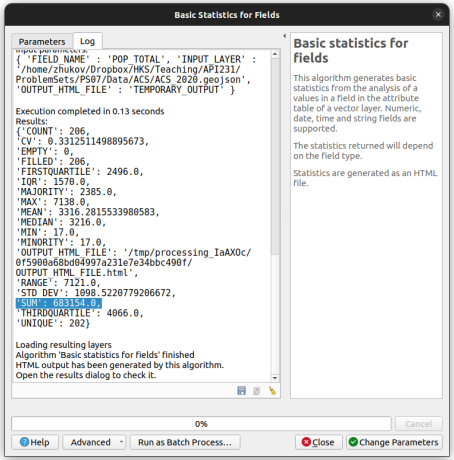
Make note of the value in the log after “SUM:”



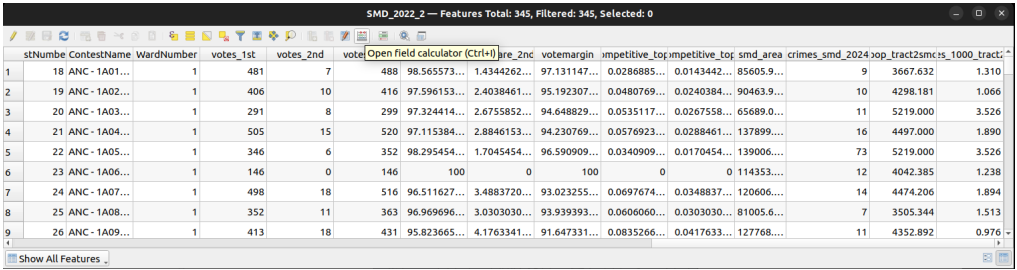
Now do the same with “Input layer” `ACS_2020` and “Field” `POP_TOTAL`



The population sums look numerically close (683,027 vs 683,154).
Interpolation was a success!

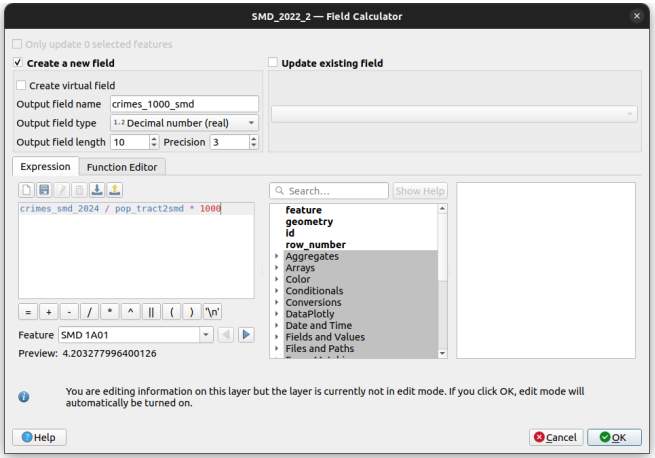


Last step (for interpolation): calculate number of crimes per 1000 residents! Go to the Attribute Table for `SMD_2022_2` and open the Field Calculator

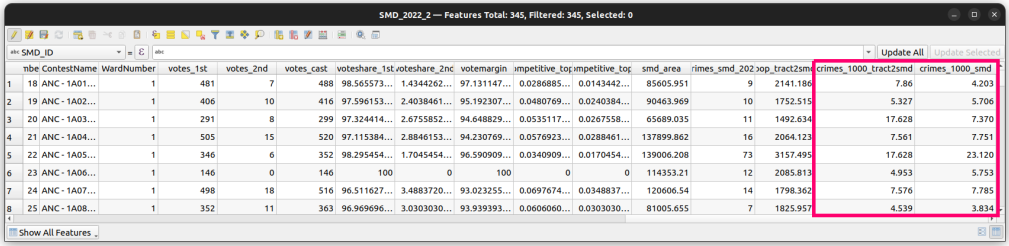


	stNumbe	ContestName	WardNumber	votes_1st	votes_2nd	vote	re_2nc	votemargin	mpetitive_top	mpetitive_top	smd_area	crimes_smd_2024	op_tract2smc	s_1000_tract2
1	18	ANC - 1A01...	1	481	7	488	98.565573...	1.4344262...	97.131147...	0.0286885...	0.0143442...	85605.9....	9	3667.632	1.310
2	19	ANC - 1A02...	1	406	10	416	97.596153...	2.4038461...	95.192307...	0.0480769...	0.0240384...	90463.9....	10	4298.181	1.066
3	20	ANC - 1A03...	1	291	8	299	97.324414...	2.6755852...	94.648829...	0.0535117...	0.0267558...	65689.0....	11	5219.000	3.526
4	21	ANC - 1A04...	1	505	15	520	97.115384...	2.8846153...	94.230769...	0.0576923...	0.0288461...	137899....	16	4497.000	1.890
5	22	ANC - 1A05...	1	346	6	352	98.295454...	1.7045454...	96.590909...	0.0340909...	0.0170454...	139006....	73	5219.000	3.526
6	23	ANC - 1A06...	1	146	0	146	100	0	100	0	0	114353....	12	4042.385	1.238
7	24	ANC - 1A07...	1	498	18	516	96.511627...	3.4883720...	93.023255...	0.0697674...	0.0348837...	120606....	14	4474.206	1.894
8	25	ANC - 1A08...	1	352	11	363	96.969696...	3.0303030...	93.939393...	0.0606060...	0.0303030...	81005.6....	7	3505.344	1.513
9	26	ANC - 1A09...	1	413	18	431	95.823665...	4.1763341...	91.647331...	0.0835266...	0.0417633...	127768....	11	4352.892	0.976

Create a new field called `crimes_1000_smd` of type `Decimal number (real)`.
Set Expression to `crimes_smd_2024 / pop_tract2smd * 1000`

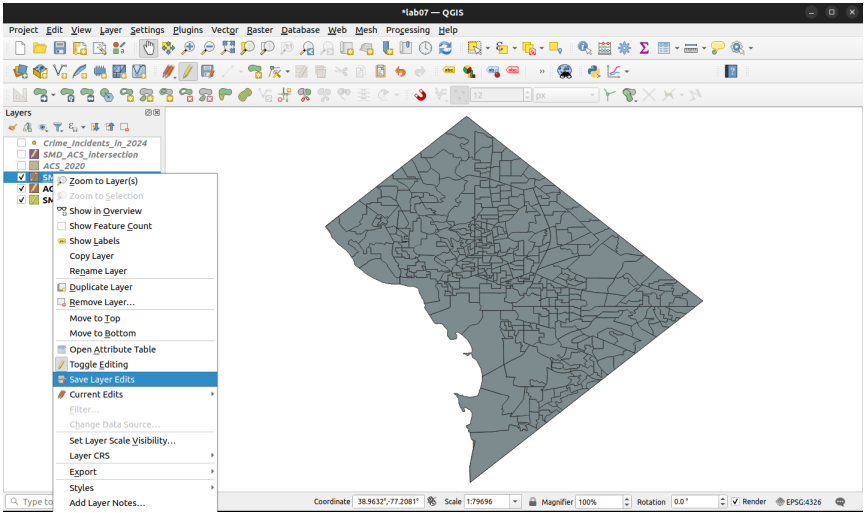


(OPTIONAL) Whole vs. parts: compare the values for the field we just constructed (`crimes_1000_smd`) from components to the `crimes_1000_tract2smd` variable we interpolated whole-cloth.

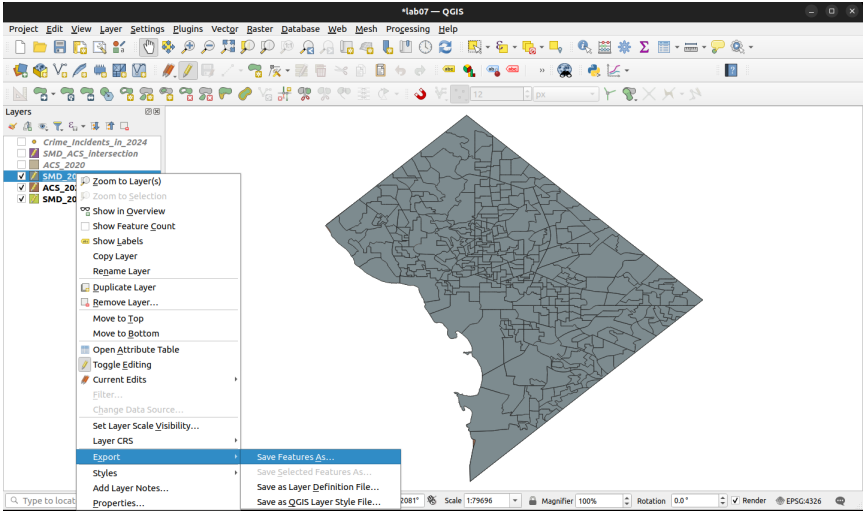


	nbe	ContestName	WardNumber	votes_1st	votes_2nd	votes_cast	voteshare_1st	voteshare_2nd	votemargin	mpetitive_top	mpetitive_top	smd_area	rimes_smd_202	op_tract2smd	crimes_1000_tract2smd	crimes_1000_smd
1	18	ANC - 1A01…	1	481	7	488	98.565573…	1.4344262…	97.131147…	0.0286885…	0.0143442…	85605.951	9	2141.186	7.86	4.203
2	19	ANC - 1A02…	1	406	10	416	97.596153…	2.4038461…	95.192307…	0.0480769…	0.0240384…	90463.969	10	1752.515	5.327	5.706
3	20	ANC - 1A03…	1	291	8	299	97.324414…	2.6755852…	94.648829…	0.0535117…	0.0267558…	65689.035	11	1492.634	17.628	7.370
4	21	ANC - 1A04…	1	505	15	520	97.115384…	2.8846153…	94.230769…	0.0576923…	0.0288461…	137899.862	16	2064.123	7.561	7.751
5	22	ANC - 1A05…	1	346	6	352	98.295454…	1.7045454…	96.590909…	0.0340909…	0.0170454…	139006.208	73	3157.495	17.628	23.120
6	23	ANC - 1A06…	1	146	0	146	100	0	100	0	0	114353.21	12	2085.813	4.953	5.753
7	24	ANC - 1A07…	1	498	18	516	96.511627…	3.4883720…	93.023255…	0.0697674…	0.0348837…	120606.54	14	1798.362	7.576	7.785
8	25	ANC - 1A08…	1	352	11	363	96.969696…	3.0303030…	93.939393…	0.0606060…	0.0303030…	81005.655	7	1825.957	4.539	3.834

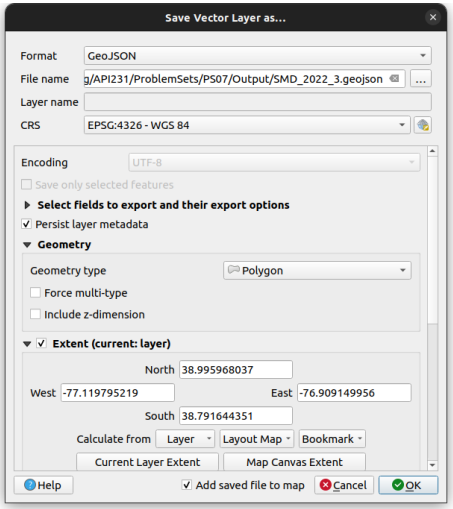
Save the edits to the SMD_2022_2 layer!



To be extra-safe, let's export the layer to a new geojson file (to preserve the join).
Right-click `SMD_2022_2` in layer menu → `Export` → `Save Features As...`

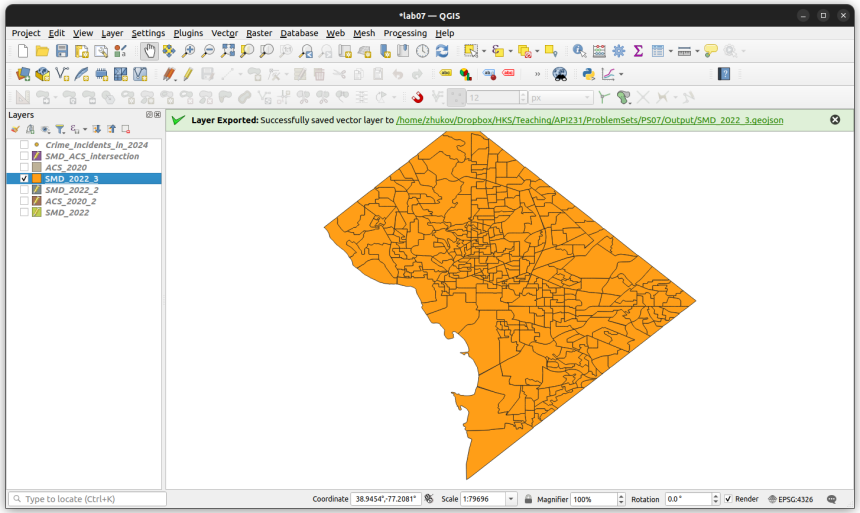


Save the layer as `SMD_2022_3.geojson` with Geometry type: `Polygon`. Check the box next to `Extent (current: layer)`. Click `OK`



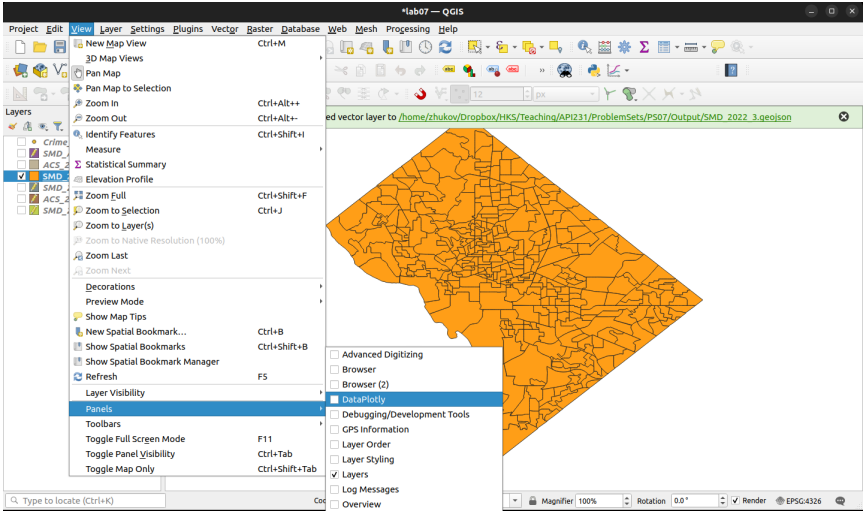
The new layer should appear in your project window.

Now we're ready to *make a scatterplot* of electoral competitiveness and crime!

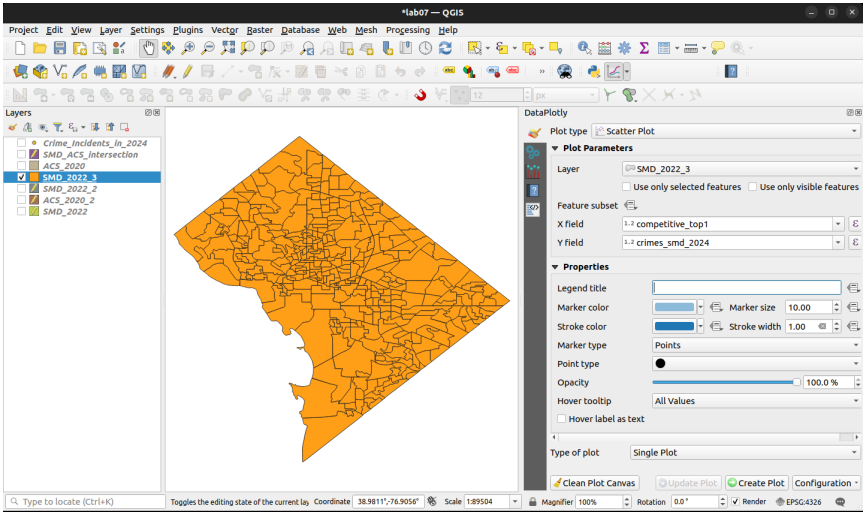


Scatterplot

To *make the scatterplots*, we will be using the `DataPlotly` plugin.
Go to `View` menu → `Panels` → ✓ `DataPlotly`

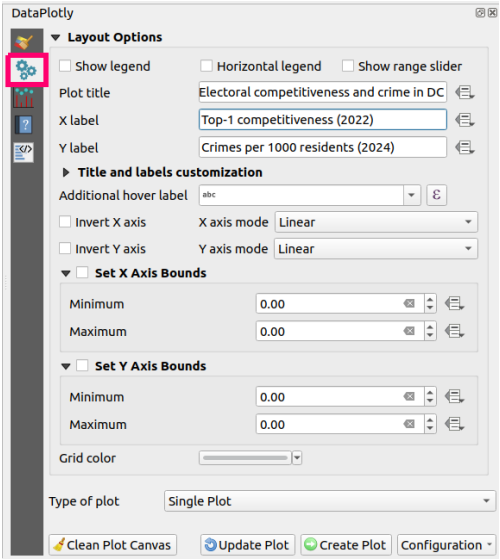


In the `DataPlotly` panel, set the following parameters:
Plot type: `Scatter Plot`; Layer: `SMD_2022_3`; X field: `competitiveness_top1`; Y field: `crimes_smd_2024`; Legend title: `[empty]`

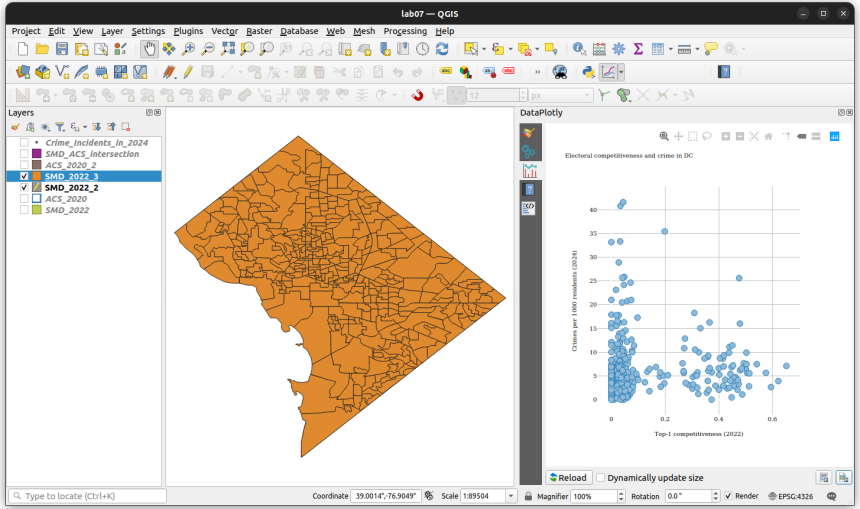


In the `Layout Options` tab, set the following parameters:

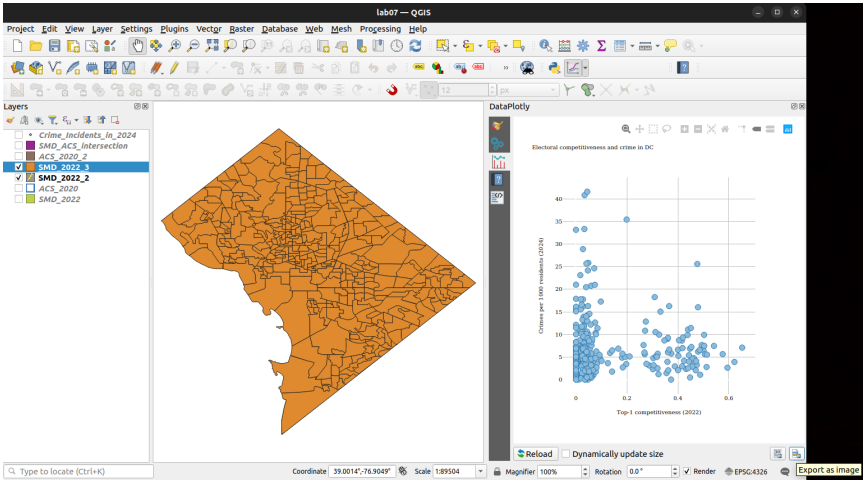
- Show legend: ☐ (off)
- Plot title: `Electoral competitiveness and crime in DC`
- X label: `Top-1 competitiveness (2022)`
- Y label: `Crimes per 1000 residents (2024)`
- Click `Create Plot`



The scatterplot should appear on the next screen.



To save the scatterplot, click the `Export as image` button in lower-right corner



Name the file `scatterplot_competitiveness_crime.png`.
It should look something like this:

Problem Set 7

Your assignment (if using QGIS): create a scatterplot of school performance and race

- perform areal interpolation:
  - source polygons: ACS census tracts (`AFS_2020.geojson`)
    x variables: neighborhood population size and racial makeup
  - destination polygons: elementary school attendance zones (`SAZ_Elementary.geojson`)
    y variables: STAR school performance scores
- make and export a scatterplot:
  - percent non-white on x-axis
  - school STAR score on y-axis
  - name the file `scatterplot_race_schools.png`
- upload plot to Canvas

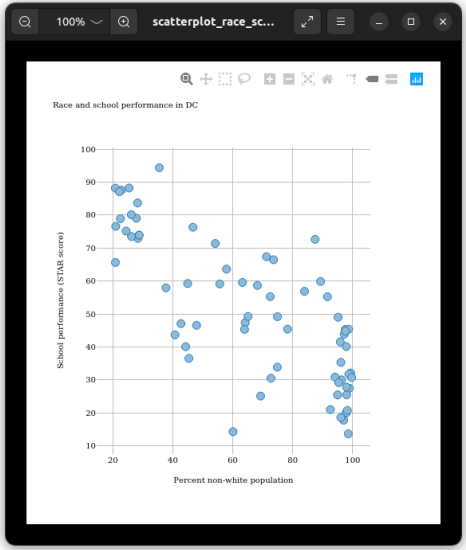
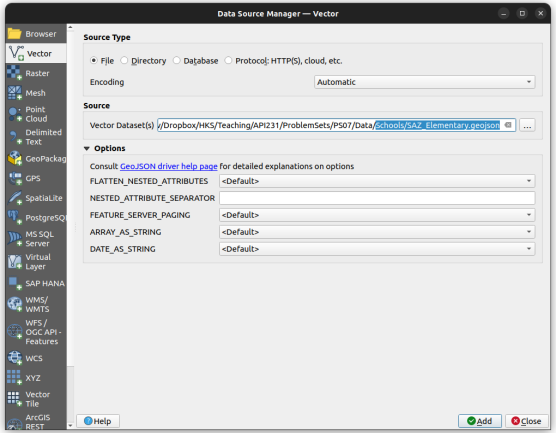
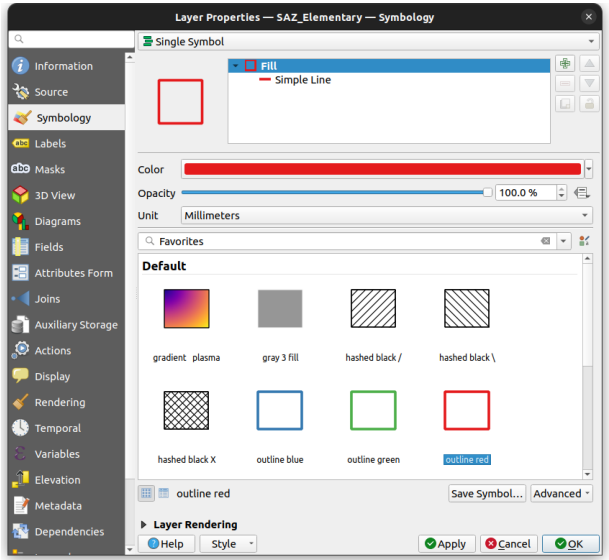


Figure 16: Can you make this?

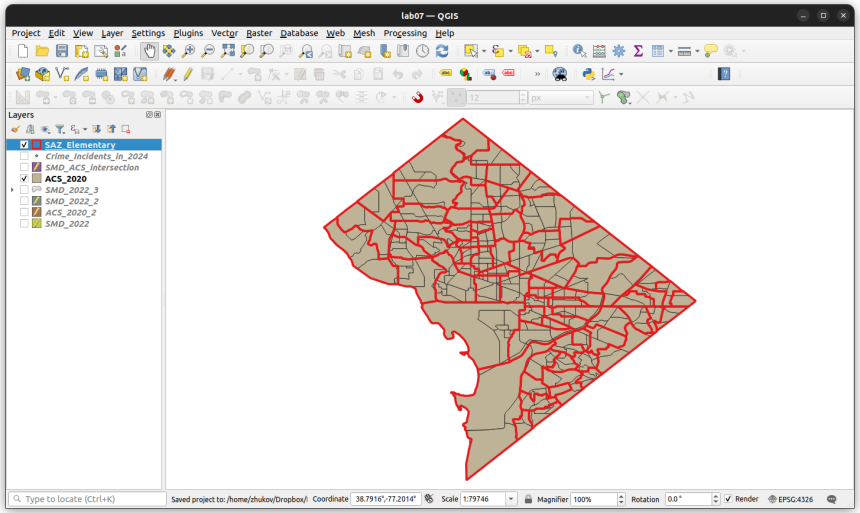
Here's a *quick preview* of the assignment.
Load the vector data file `SAZ_Elementary.geojson` from `Data/Schools`



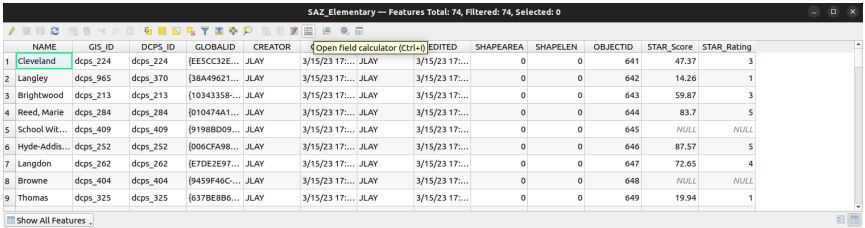
To see how `SAZ_Elementary` overlays with census tracts, select a symbology with bright thick borders and a transparent inside, like `outline_red`



Because SAZs are (mostly) larger and fewer in number than tracts (74 vs 206), it seems more appropriate for SAZs to be the destination units

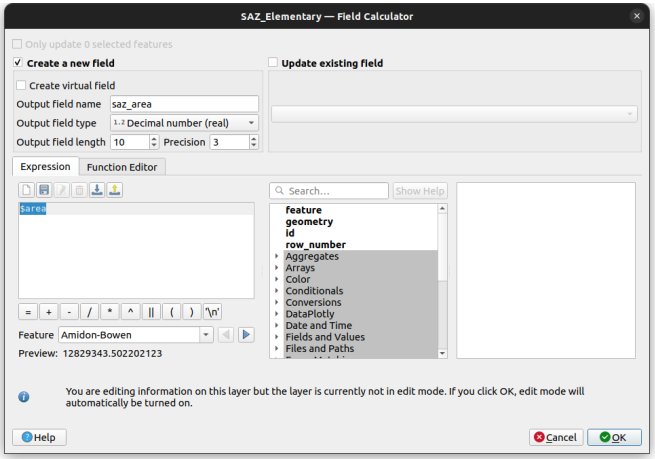


Let's construct the area weights, starting with the area of destination polygons, a_j. Recall that we already calculated the area of source polygons, a_i (`tract_area` in `ACS_2020`). Go to the Field Calculator for `SAZ_Elementary`

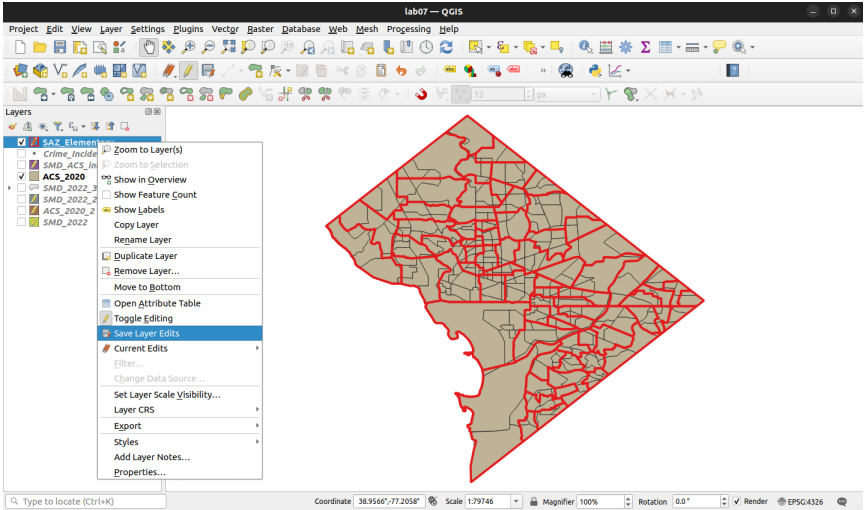


	NAME	GIS_ID	DCPS_ID	GLOBALID	CREATOR	EDITED	SHAPEAREA	SHAPELEN	OBJECTID	STAR_Score	STAR_Rating
1	Cleveland	dcps_224	dcps_224	{EE5CC32E…	JLAY	3/15/23 17:…	0	0	641	47.37	3
2	Langley	dcps_965	dcps_370	{38A49621…	JLAY	3/15/23 17:…	0	0	642	14.26	1
3	Brightwood	dcps_213	dcps_213	{10343358-…	JLAY	3/15/23 17:…	0	0	643	59.87	3
4	Reed, Marie	dcps_284	dcps_284	{010474A1…	JLAY	3/15/23 17:…	0	0	644	83.7	5
5	School Wit…	dcps_409	dcps_409	{9198BD09…	JLAY	3/15/23 17:…	0	0	645	NULL	NULL
6	Hyde-Addis…	dcps_252	dcps_252	{006CFA98…	JLAY	3/15/23 17:…	0	0	646	87.57	5
7	Langdon	dcps_262	dcps_262	{E7DE2E97…	JLAY	3/15/23 17:…	0	0	647	72.65	4
8	Browne	dcps_404	dcps_404	{9459F46C-…	JLAY	3/15/23 17:…	0	0	648	NULL	NULL
9	Thomas	dcps_325	dcps_325	{637BE8B6…	JLAY	3/15/23 17:…	0	0	649	19.94	1

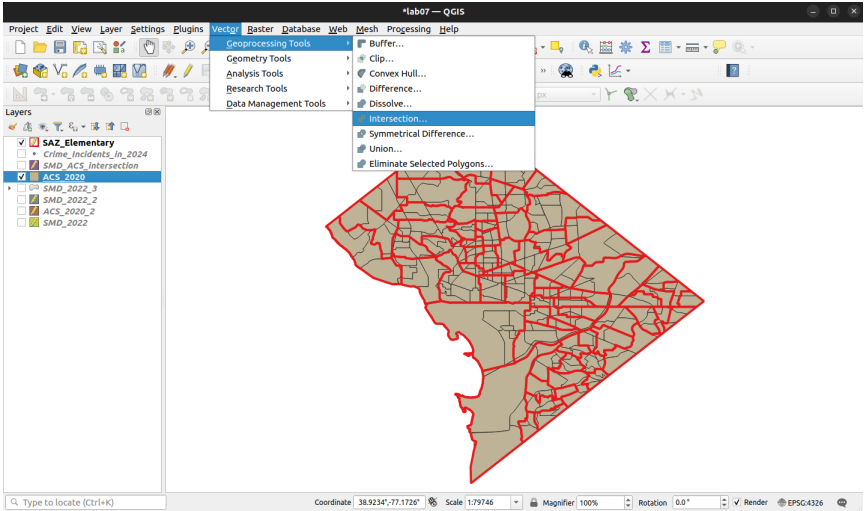
Create a new field named `saz_area` of type `Decimal number`.
Set the expression to `$area`



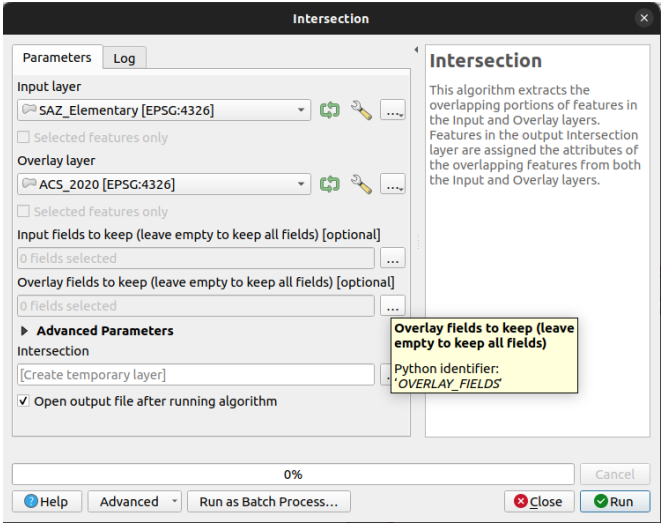
Remember to save your layer edits regularly!



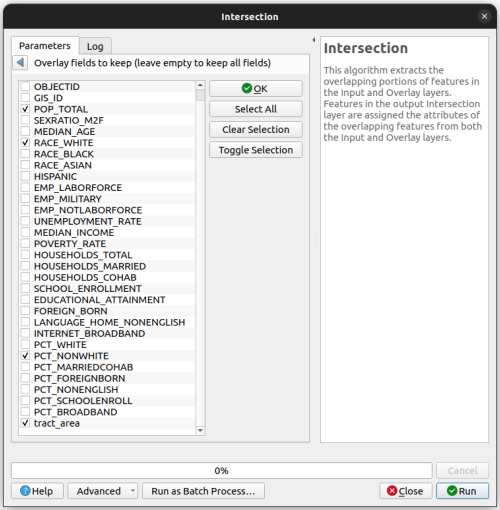
Now let's intersect `SAZ_Elementary` with `ACS_2020` to calculate intersection areas, $a_{i \cap j}$. Go to `Vector` menu → `Geoprocessing Tools` → `Intersection...`



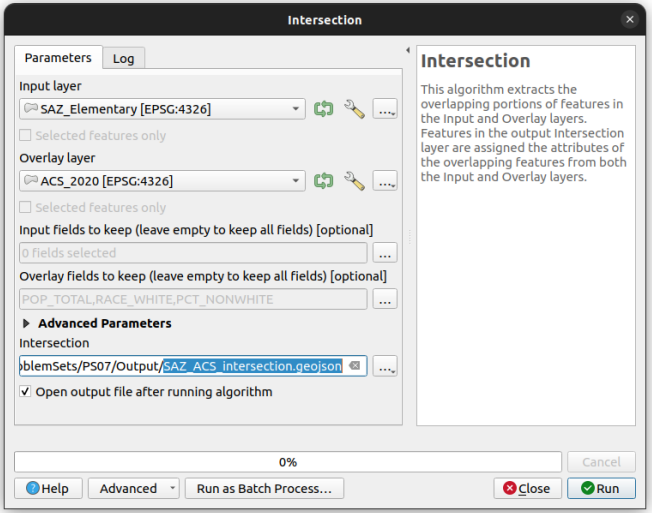
For the intersection, select `SAZ_Elementary` as Input Layer, `ACS_2020` as Overlay layer. Then click the `[...]` button next to "Overlay fields to keep"



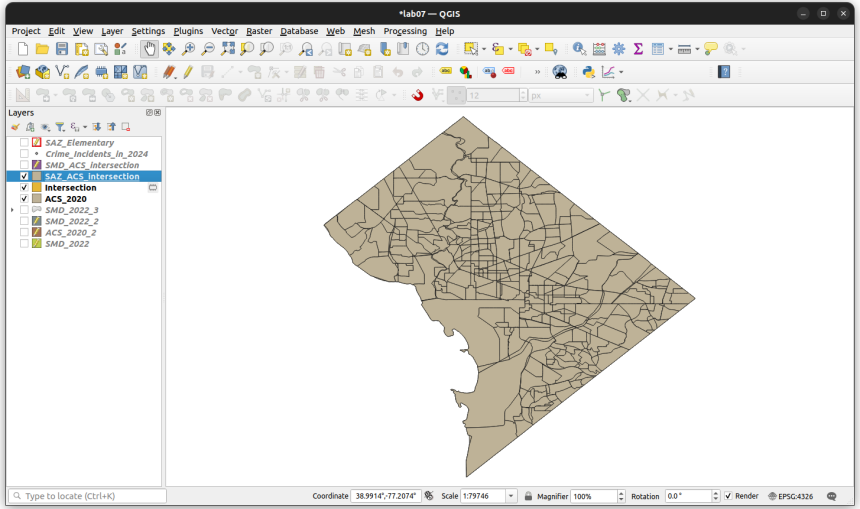
On the next screen check the boxes next to ✓ `POP_TOTAL`, ✓ `RACE_WHITE`, ✓ `PCT_NONWHITE` and ✓ `tract_area`. Click `OK`



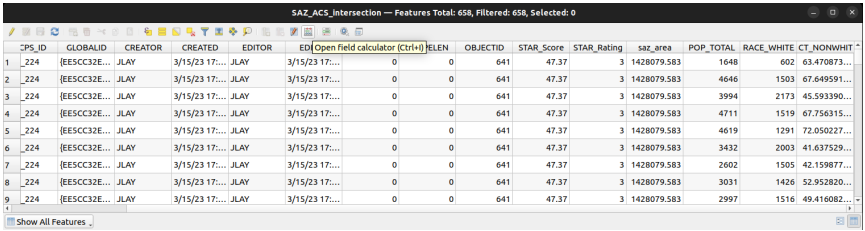
Save the output file to `SAZ_ACS_intersection.geojson`, click `Run`



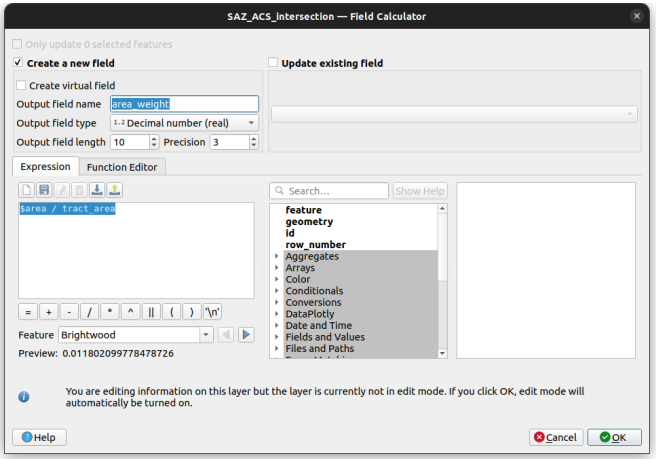
The intersection should appear in your project window



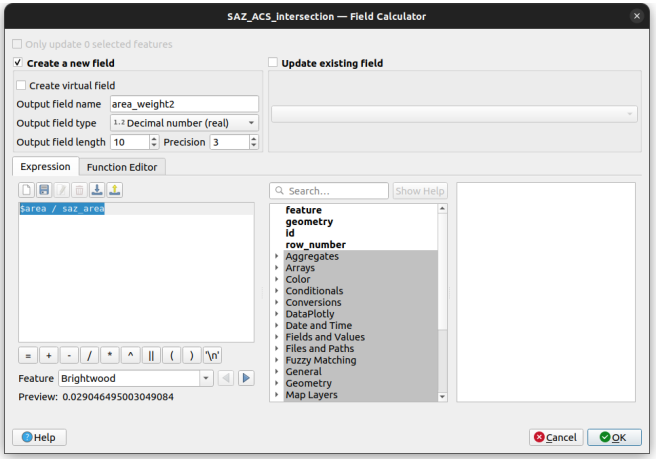
Launch the Field Calculator for `SAZ_ACS_intersection`



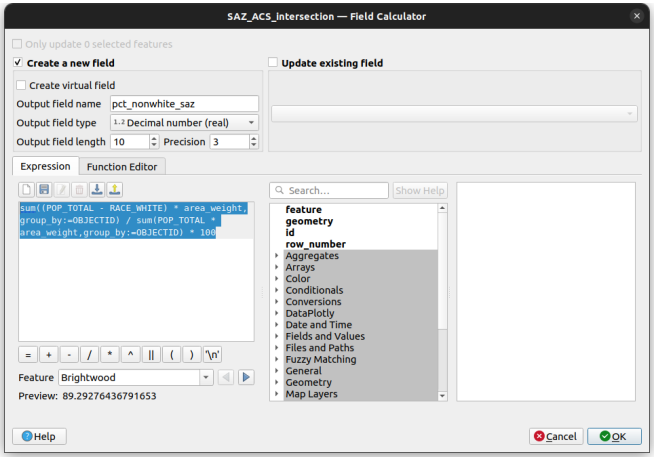
Create a new field `area_weight` of type `Decimal number`.
Set expression to `$area / tract_area`. Click `OK`



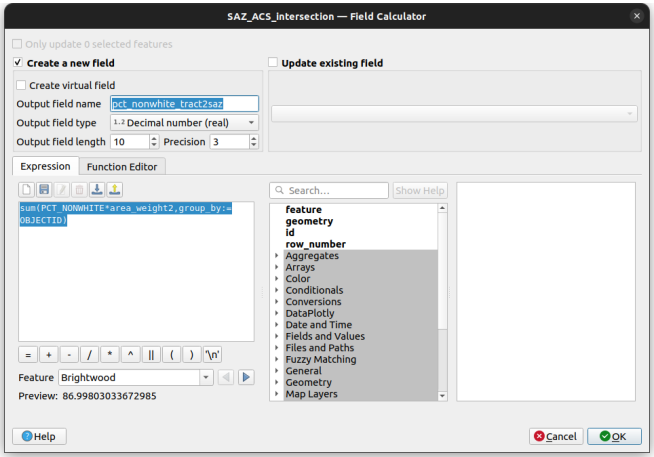
(OPTIONAL) For the intensive transformation, create a new field `area_weight2` of type `Decimal number`. Set expression to `$area / saz_area`. Click `OK`



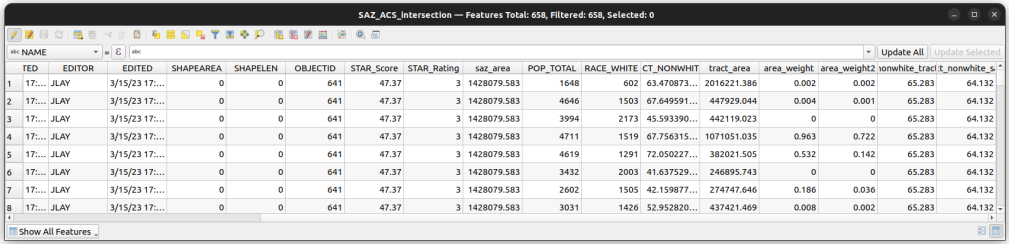
Create a new field `pct_nonwhite_saz` of type `Decimal number`. Set expression to `sum((POP_TOTAL - RACE_WHITE) * area_weight,group_by:=OBJECTID) / sum(POP_TOTAL * area_weight,group_by:=OBJECTID) * 100`



(OPTIONAL) Create a new field `pct_nonwhite_tract2saz` of type `Decimal number`. Set expression to `sum(PCT_NONWHITE * area_weight2,group_by:=OBJECTID)`

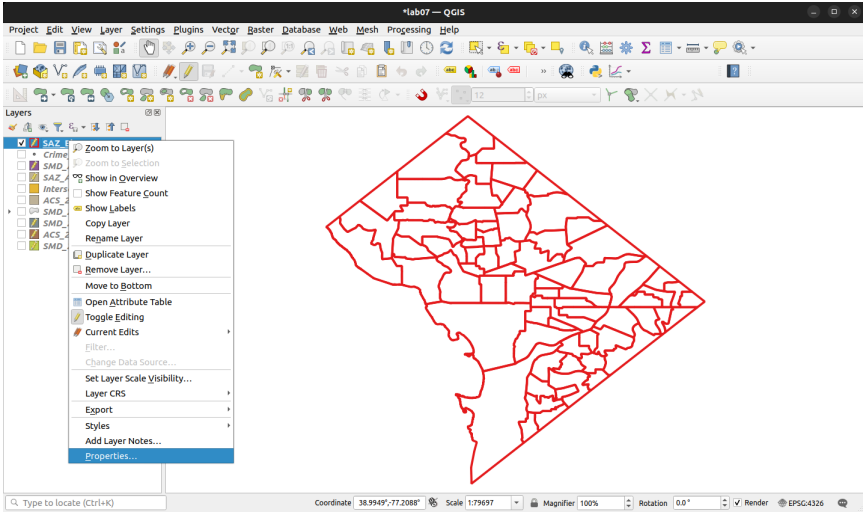


Inspect the `pct_nonwhite_saz` variable in the Attribute Table.
Is it constant within SAZs? If so, we're ready to join it to the SAZs

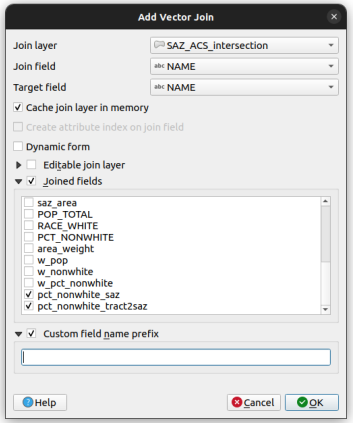


	TED	EDITOR	EDITED	SHAPEAREA	SHAPELEN	OBJECTID	STAR_Score	STAR_Rating	saz_area	POP_TOTAL	RACE_WHITE	CT_NONWHIT	tract_area	area_weight	area_weight2	onwhite_tract	t_nonwhite_s
1	17:...	JLAY	3/15/23 17:...	0	0	641	47.37	3	1428079.583	1648	602	63.470873...	2016221.386	0.002	0.002	65.283	64.132
2	17:...	JLAY	3/15/23 17:...	0	0	641	47.37	3	1428079.583	4646	1503	67.649591...	447929.044	0.004	0.001	65.283	64.132
3	17:...	JLAY	3/15/23 17:...	0	0	641	47.37	3	1428079.583	3994	2173	45.593390...	442119.023	0	0	65.283	64.132
4	17:...	JLAY	3/15/23 17:...	0	0	641	47.37	3	1428079.583	4711	1519	67.756315...	1071051.035	0.963	0.722	65.283	64.132
5	17:...	JLAY	3/15/23 17:...	0	0	641	47.37	3	1428079.583	4619	1291	72.050227...	382021.505	0.532	0.142	65.283	64.132
6	17:...	JLAY	3/15/23 17:...	0	0	641	47.37	3	1428079.583	3432	2003	41.637529...	246895.743	0	0	65.283	64.132
7	17:...	JLAY	3/15/23 17:...	0	0	641	47.37	3	1428079.583	2602	1505	42.159877...	274747.646	0.186	0.036	65.283	64.132
8	17:...	JLAY	3/15/23 17:...	0	0	641	47.37	3	1428079.583	3031	1426	52.952820...	437421.469	0.008	0.002	65.283	64.132

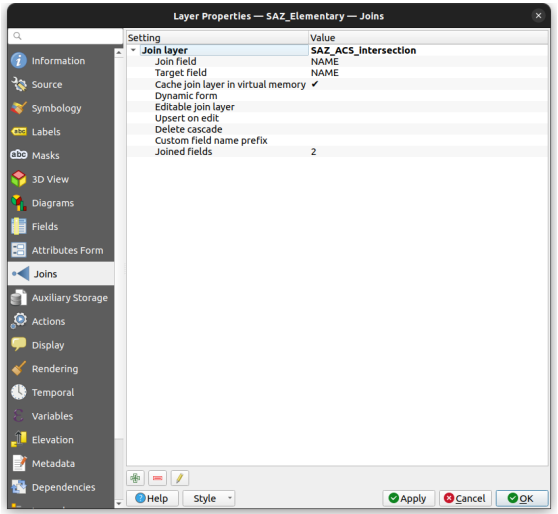
Now let's import these weighted values into the `SAZ_Elementary` layer. Right-click on `SAZ_Elementary` → `Properties`



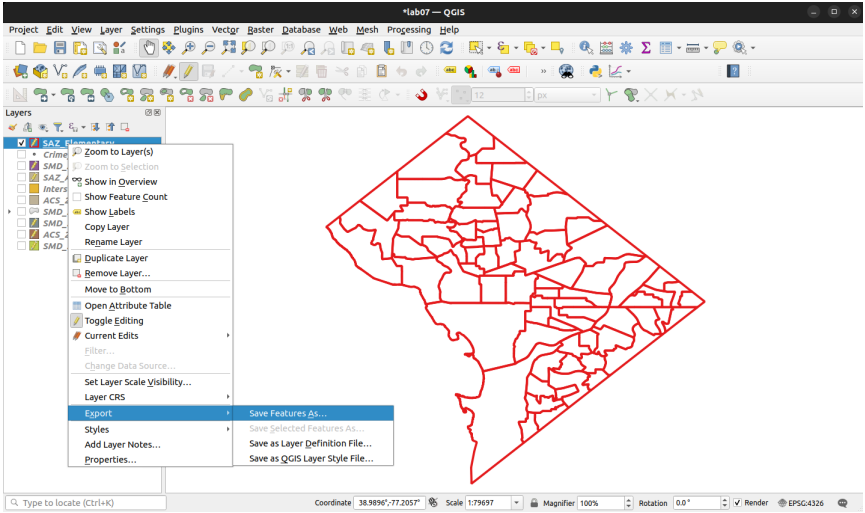
Go to `Joins` tab and click + to add a new join. Set `SAZ_ACS_intersection` as the "Join layer", `NAME` as the "Join" and "Target" field, and under "Joined fields" select ✓ `pct_nonwhite_saz` and ✓ `pct_nonwhite_tract2saz` (if you created this variable, too). Check the box next to `Custom field name prefix` and clear the contents (i.e. no prefix). Click `OK`



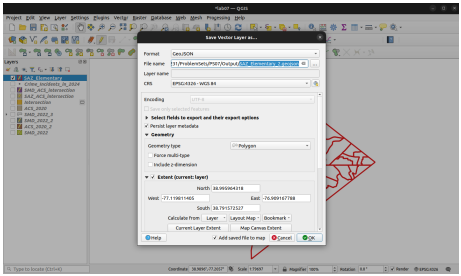
You should now see the new join layer in Properties



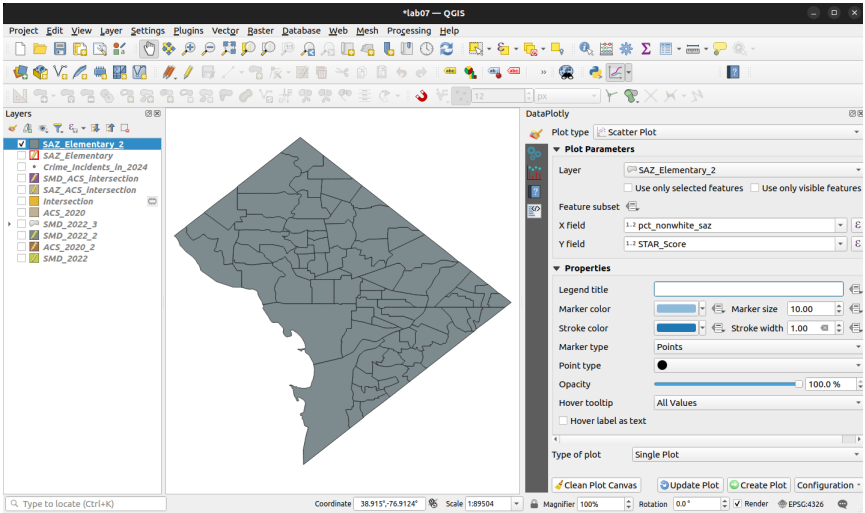
Export the layer to a new geojson file (to preserve the join). Right-click `SAZ_Elementary` in layer menu → `Export` → `Save Features As...`



Save the layer as `SAZ_Elementary_2.geojson` with Geometry type: `Polygon`. Check the box next to `Extent (current: layer)`. Click `OK`

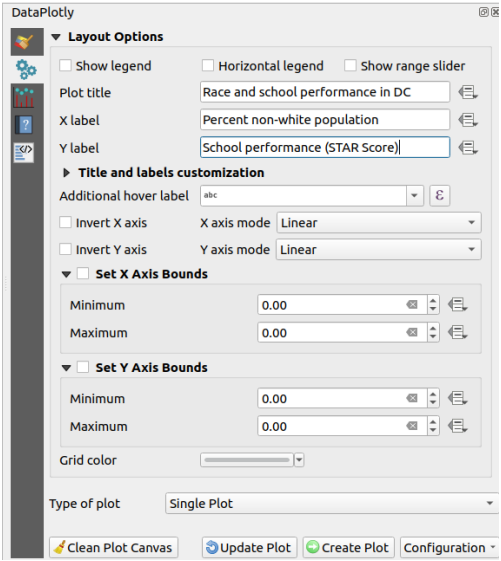


In the DataPlotly panel, set the following parameters:
Plot type: Scatter Plot; Layer: SAZ_Elementary_2; X field: pct_nonwhite_saz; Y field: STAR_Score; Legend title: [empty]

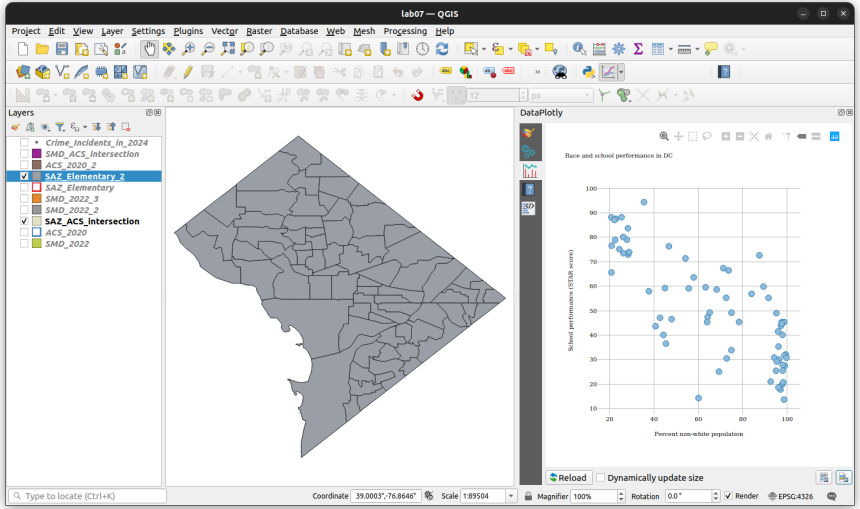


In the `Layout Options` tab, set the following parameters:

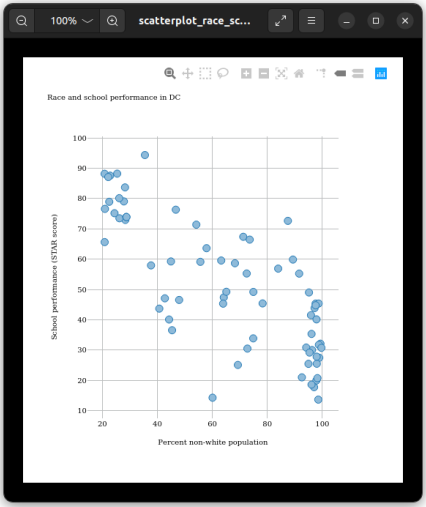
- Show legend: ☐ (off)
- Plot title: `Race and school performance in DC`
- X label: `Percent non-white population`
- Y label: `School performance (STAR Score)`
- Click `Create Plot`



The scatterplot should appear on the next screen.



Export the image as `scatterplot_race_schools.png`.
It should look something like this:



R

Loading R packages

To implement these steps in R, we will be using the `sf` and `SUNGEO` packages

```
library(sf)
library(SUNGEO)
```

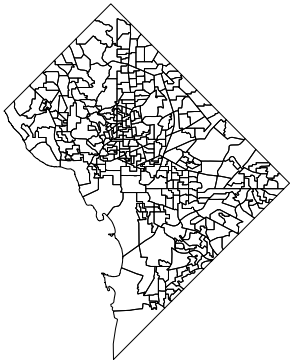
NOTE: The demo code for R is in `ps07_demo.R` on RStudio Cloud, and in `PS07.zip` (posted on Canvas).

Interpolation

Loading spatial data

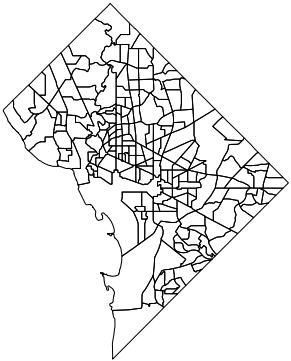
Let's load the *single member district boundaries* into R, using `sf::read_sf()`:

```
smd_2022 = sf::read_sf("Data/Elections/SMD_2022.geojson")
plot(smd_2022["geometry"])
```



Now load the *ACS census tracts* into R:

```
acs_2020 = sf::read_sf("Data/ACS/ACS_2020.geojson")
plot(acs_2020["geometry"])
```



. . . and the *crime incidents data*:

```
crimes_2024 = sf::read_sf("Data/Crime/Crime_Incidents_in_2024.geojson")
plot(crimes_2024["geometry"])
```

Let's calculate *relative scale and nesting* metrics for a change of support from acs_2020 (census tracts) to smd_2022 (single member districts)

```
SUNGEO::nesting(acs_2020,smd_2022,metrix=c("rn","rs"))
```

```
## $rn
## [1] 0.4594585
##
## $rs
## [1] 0.1793275
```

Yikes! What if we switched to (smaller) census block groups as source units?

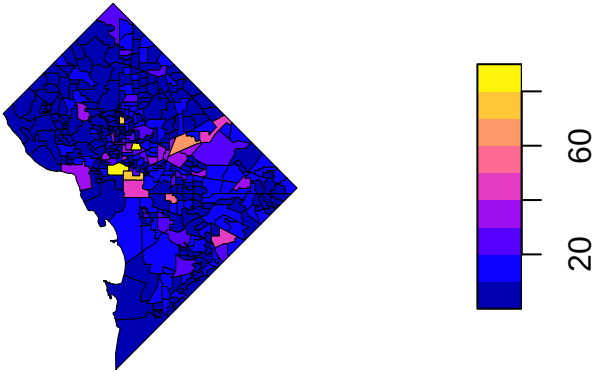
```
census_2020 = sf::read_sf("Data/Census/CENSUS_2020.geojson")
SUNGEO::nesting(census_2020,smd_2022,metrix=c("rn","rs"))
```

```
## $rn
## [1] 0.6907815
##
## $rs
## [1] 0.8217758
```

Much better! But for consistency with QGIS, we'll stick with the tracts here.

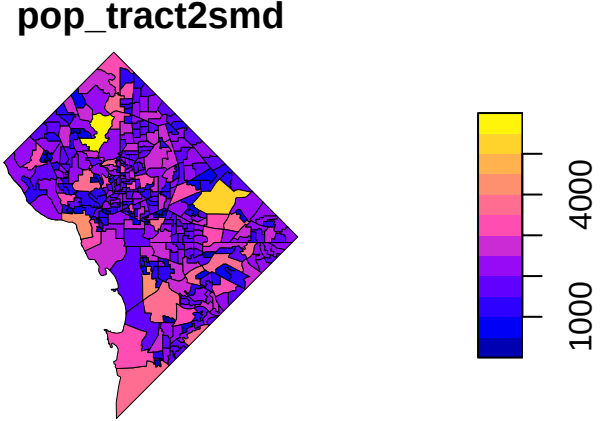
Let's do some point-in-polygon analysis to count crimes per SMD:

```
smd_2022$crimes_smd_2024 = lengths(sf::st_intersects(smd_2022,
                                                     crimes_2024))
plot(smd_2022["crimes_smd_2024"])
```



Let's *interpolate population counts* per SMD (this is a one-step routine in R)

```
smd_2022$pop_tract2smd = sf::st_interpolate_aw(acs_2020["POP_TOTAL"],
                                               smd_2022,extensive=TRUE)$POP_TOTAL
plot(smd_2022["pop_tract2smd"])
```



Let's check the quality of transformation: how close are the population counts?

```
sum(smd_2022$pop_tract2smd) # Interpolated
```

```
## [1] 683070.1
```

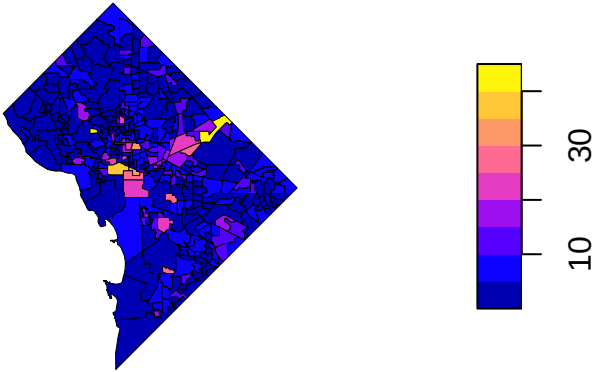
```
sum(acs_2020$POP_TOTAL)     # Original
```

```
## [1] 683154
```

Let's *calculate crimes per 1000 residents*

```
smd_2022$crimes_1000_smd = smd_2022$crimes_smd_2024/
                           smd_2022$pop_tract2smd*1000
plot(smd_2022["crimes_1000_smd"])
```

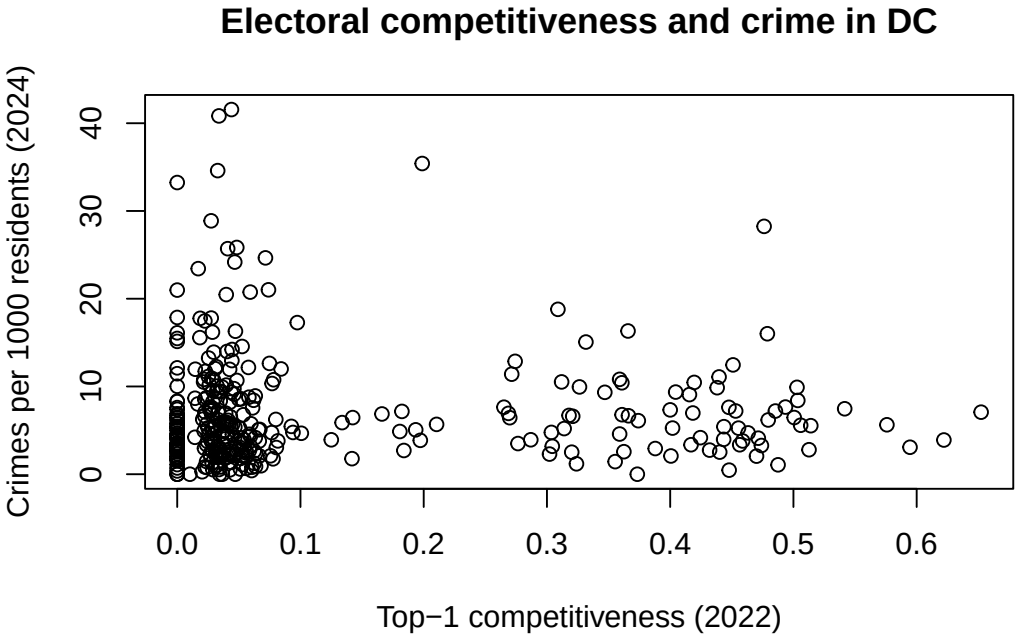
crimes_1000_smd



Scatterplot

Let's *create the scatterplot*

```
plot(x=smd_2022$competitive_top1,
  y=smd_2022$crimes_1000_smd,
  xlab="Top-1 competitiveness (2022)",
  ylab="Crimes per 1000 residents (2024)",
  main="Electoral competitiveness and crime in DC"
  )
```


Electoral competitiveness and crime in DC
Crimes per 1000 residents (2024)
0
10
20
30
40
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Top−1 competitiveness (2022)

Problem Set 7

Your assignment (if using R):

- create a scatterplot of race and school performance in DC!
- perform areal interpolation:
  - source polygons: `AFS_2020.geojson`
  - y variables: `POP_TOTAL` and `RACE_WHITE`
  - destination polygons: `SAZ_Elementary.geojson`
  - y variables: `STAR_Score`
- make and export a scatterplot:
  - percent non-white on x-axis
  - school STAR score on y-axis
  - name the file `scatterplot_race_schools_R.png`
- upload map to Canvas (by next Wednesday)

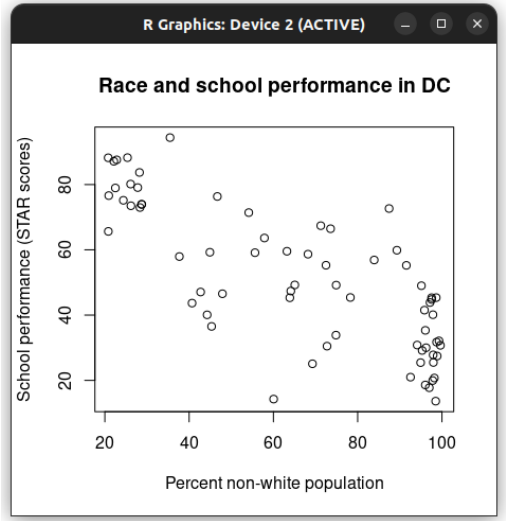


Figure 17: Can you make this?