SEST-6577

Geographic Information Systems for Security Studies

Lab 09 (+ Walk Through 1)

Yuri M. Zhukov
Associate Professor
Georgetown University

November 4, 2025

Goal: explain geographic variation in Islamic State violence

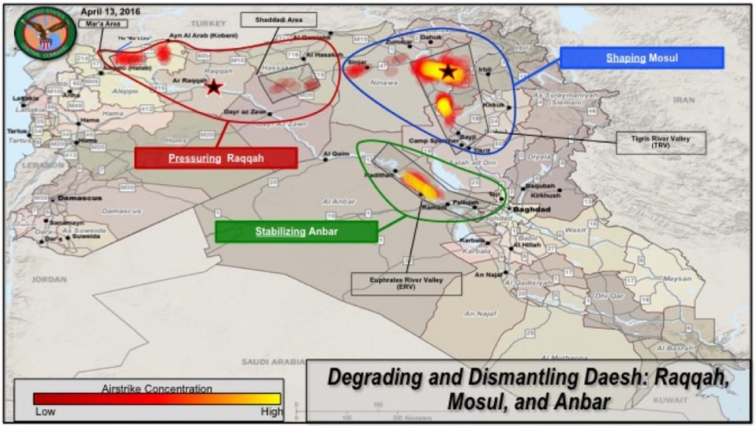


Figure 1: Operation Inherent Resolve

There is no formal Problem Set this week, but if you would like to replicate this in QGIS, follow along (for walk-through in R or Python, see replication code on Canvas)

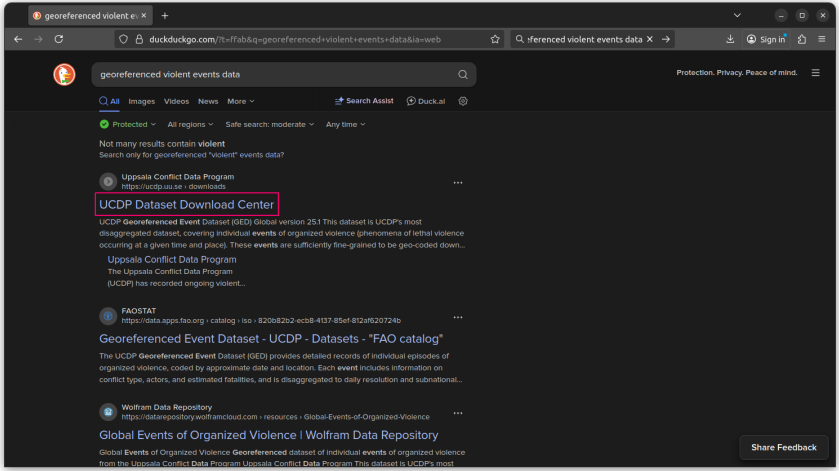
Overview of lab exercise

1. Data
   a) Collect raw data from open sources online
   b) Pre-processing to integrate data to common set of spatial units
2. Analysis
   a) Export pre-processed data as comma-separated file
   b) Run regression models in R (or Python)

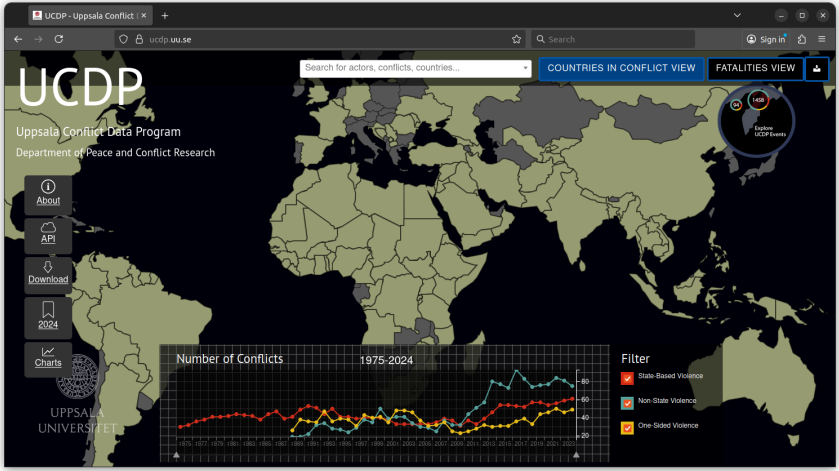
Data

Collection

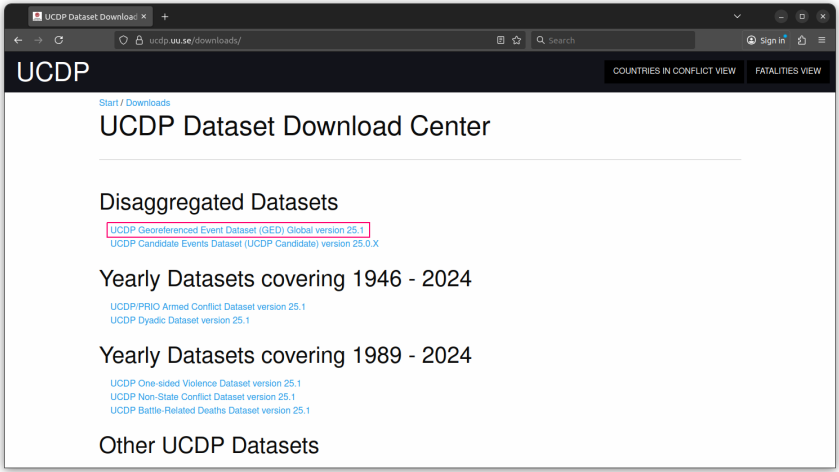
Data on **political violence** are easily found online. If we search for `georeferenced violent events data`, a top result is for "UCDP Dataset Download Center"



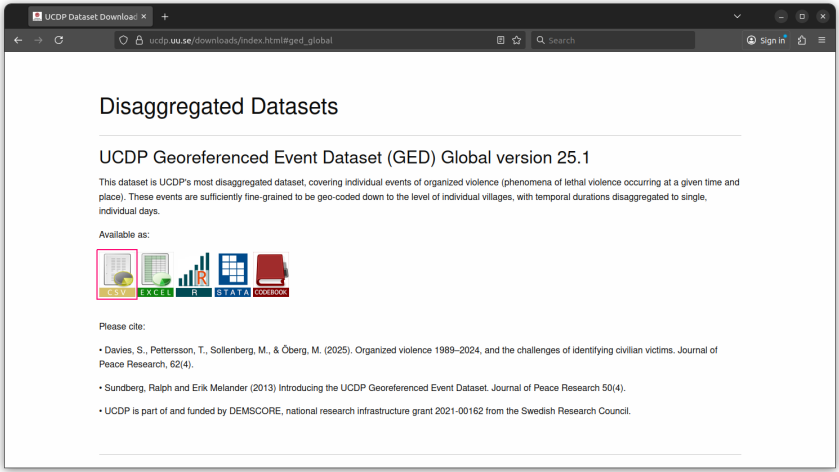
The Uppsala Conflict Data Program (UCDP) is one of the world's leading providers of open-source data on violence and armed conflict, based at Uppsala University



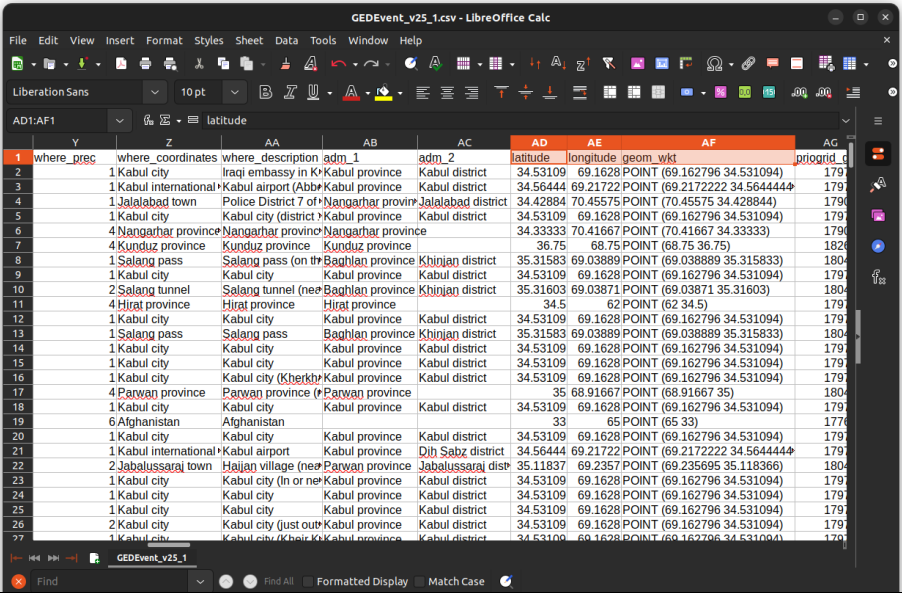
Download-able data are located at `ucdp.uu.se/downloads`.
Click on "Georeferenced Event Dataset (GED)"



We have used GED before, for Problem Set 02 (Afghanistan and Yemen).
Download GED as a `csv` file and extract it from the `zip` archive

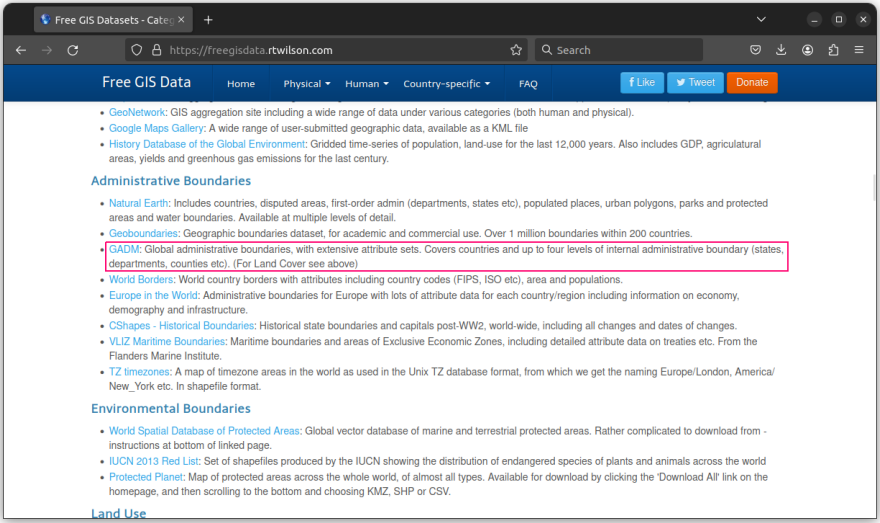


There are about 317K events here, each of which has point coordinates

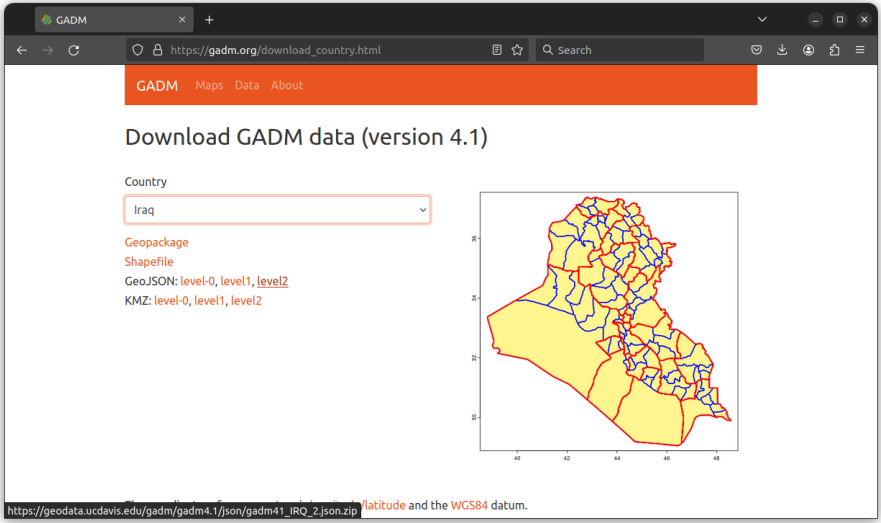


	Y	Z	AA	AB	AC	AD	AE	AF	AG
1	where_prec	where_coordinates	where_description	adm_1	adm_2	latitude	longitude	geom_wkt	priogrid_g
2	1	Kabul city	Iraqi embassy in K	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
3	1	Kabul international	Kabul airport (Abb	Kabul province	Kabul district	34.56444	69.21722	POINT (69.2172222 34.5644444	179
4	1	Jalalabad town	Police District 7 of	Nangarhar provin	Jalalabad district	34.42884	70.45575	POINT (70.45575 34.428844)	179
5	1	Kabul city	Kabul city (district	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
6	4	Nangarhar province	Nangarhar provinc	Nangarhar province		34.33333	70.41667	POINT (70.41667 34.33333)	179
7	4	Kunduz province	Kunduz province	Kunduz province		36.75	68.75	POINT (68.75 36.75)	182
8	1	Salang pass	Salang pass (on th	Baghlan province	Khinjan district	35.31583	69.03889	POINT (69.038889 35.315833)	180
9	1	Kabul city	Kabul city	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
10	2	Salang tunnel	Salang tunnel (nea	Baghlan province	Khinjan district	35.31603	69.03871	POINT (69.03871 35.31603)	180
11	4	Hirat province	Hirat province	Hirat province		34.5	62	POINT (62 34.5)	179
12	1	Kabul city	Kabul city	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
13	1	Salang pass	Salang pass	Baghlan province	Khinjan district	35.31583	69.03889	POINT (69.038889 35.315833)	180
14	1	Kabul city	Kabul city	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
15	1	Kabul city	Kabul city	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
16	1	Kabul city	Kabul city (Kherkh	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
17	4	Parwan province	Parwan province (P	Parwan province		35	68.91667	POINT (68.91667 35)	180
18	1	Kabul city	Kabul city	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
19	6	Afghanistan	Afghanistan			33	65	POINT (65 33)	177
20	1	Kabul city	Kabul city	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
21	1	Kabul international	Kabul airport	Kabul province	Dih Sabz district	34.56444	69.21722	POINT (69.2172222 34.5644444	179
22	2	Jabalussaraj town	Hajjan village (nea	Parwan province	Jabalussaraj distr	35.11837	69.2357	POINT (69.235695 35.118366)	180
23	1	Kabul city	Kabul city (In or ne	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
24	1	Kabul city	Kabul city	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
25	1	Kabul city	Kabul city	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
26	2	Kabul city	Kabul city (just out	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179
27	1	Kabul city	Kabul city (Kheir Kh	Kabul province	Kabul district	34.53109	69.1628	POINT (69.162796 34.531094)	179

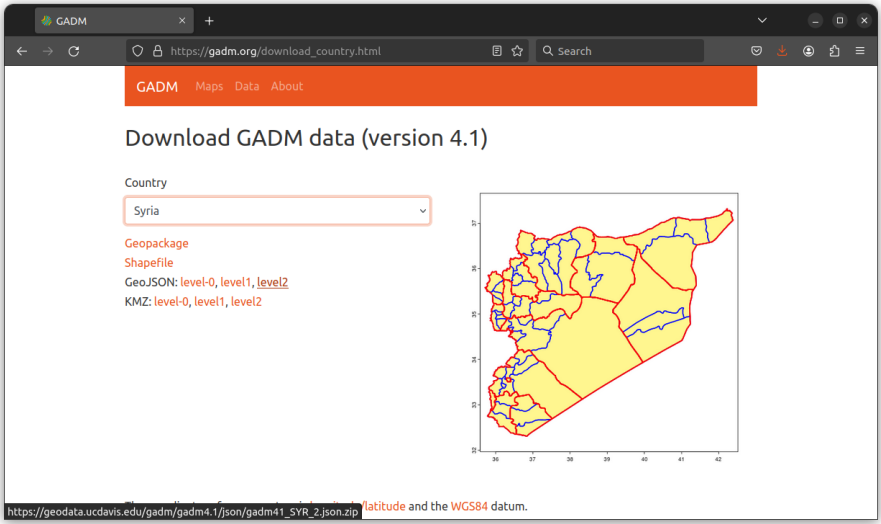
For data on **administrative boundaries**, Rick Wilson's "Free GIS Data" site has links to multiple data sources. Of these, we will use `gadm.org`



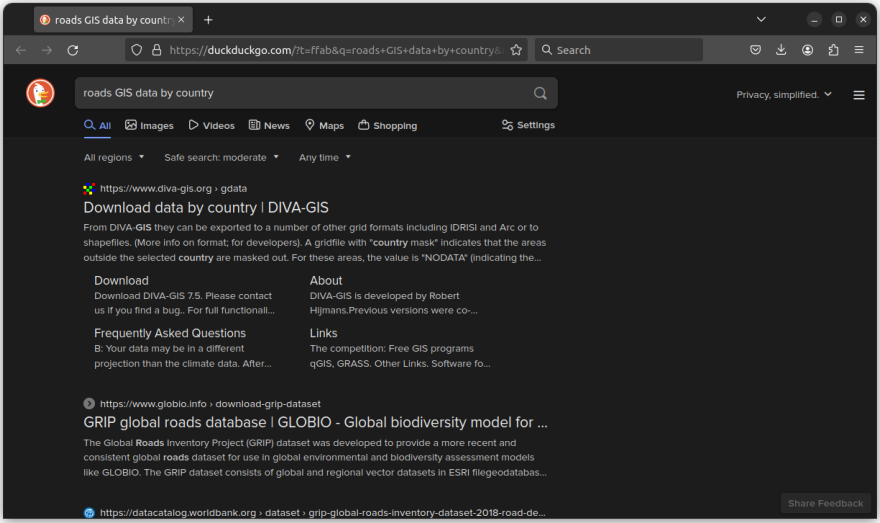
From GADM's website, we can get data on sub-national administrative boundaries. Download the `level2` `GeoJSON` file for Iraq...



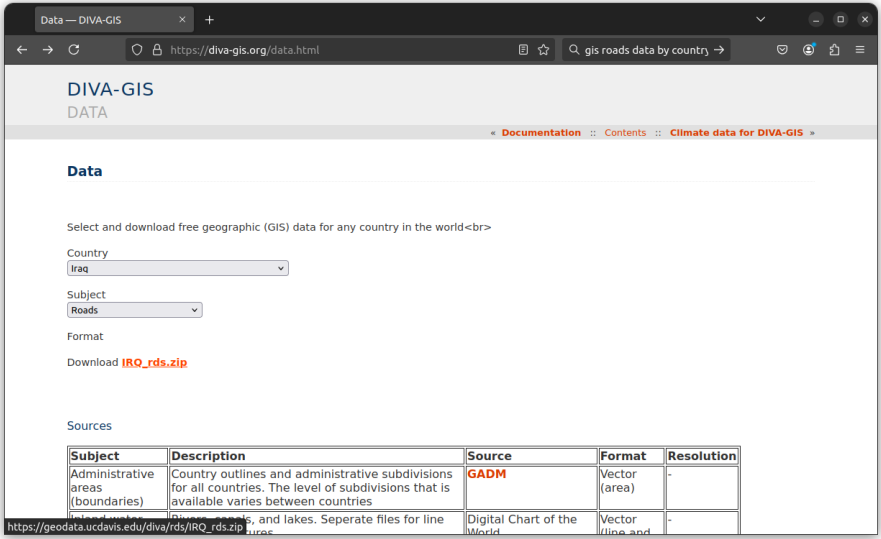
. . . and for Syria



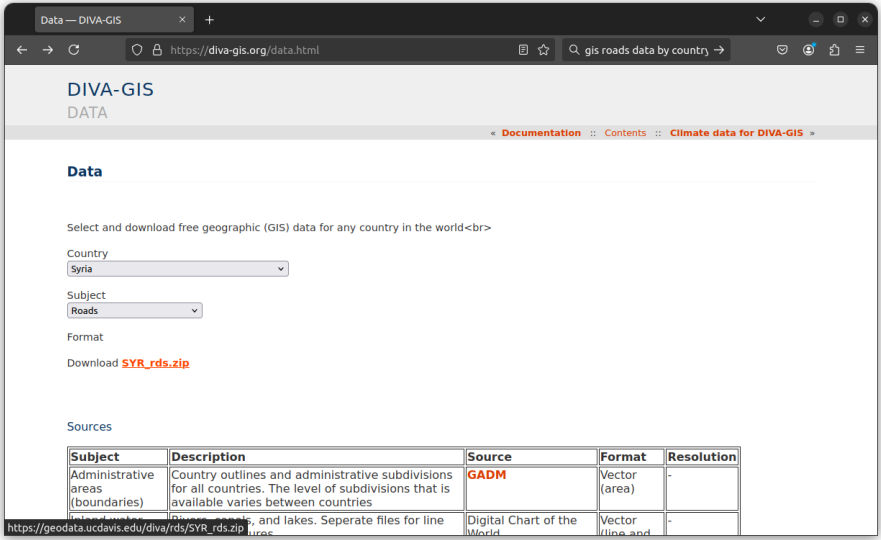
Multiple data sources exist for **roads**. Among the easiest to get are those from `diva-gis.org/gData`



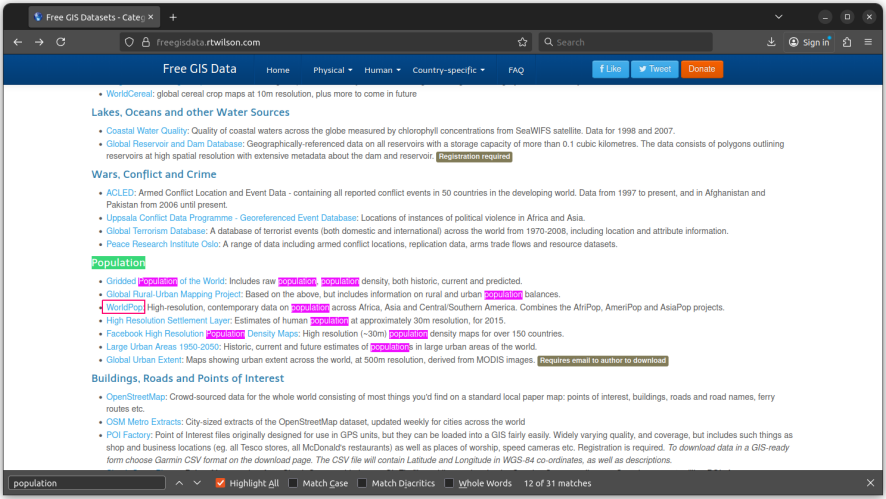
These are country-level road layers from “Digital Chart of the World”, download-able as zipped shapefiles



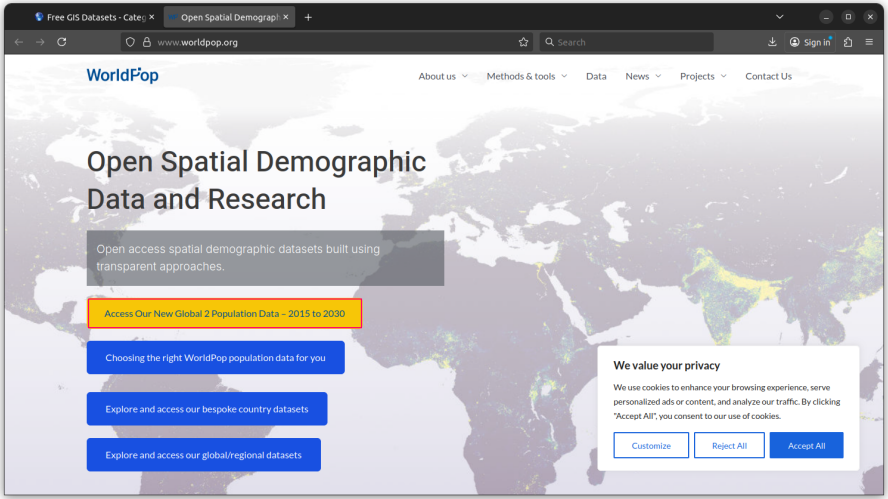
Download the roads data for both Iraq and Syria here



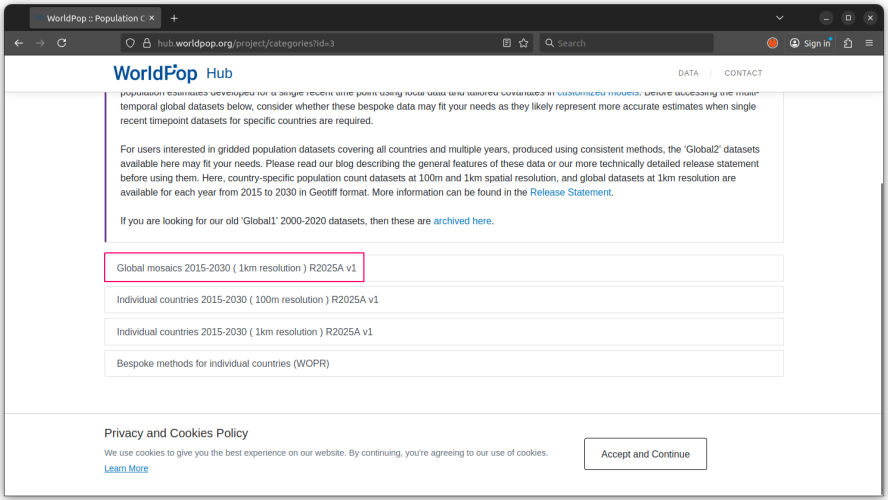
For **population data**, we can go back to Rick Wilson's page. There are many options, but let's try "WorldPop"



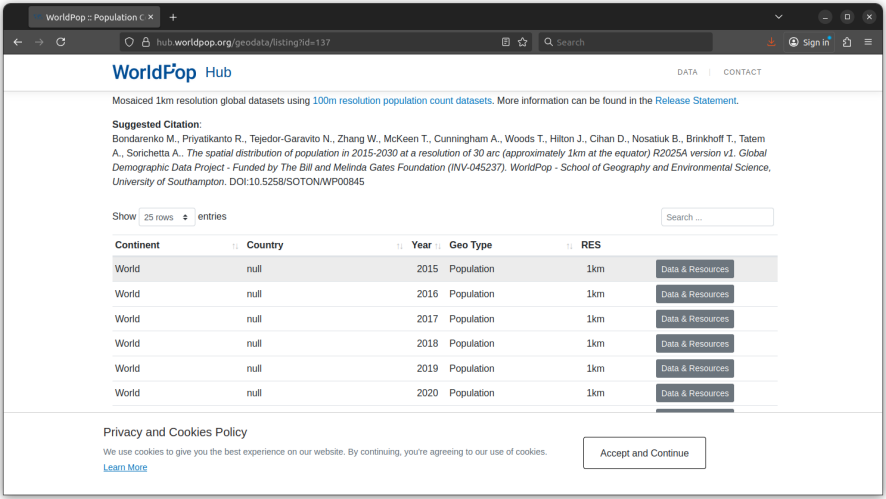
On the landing page, go to “Access... data”.



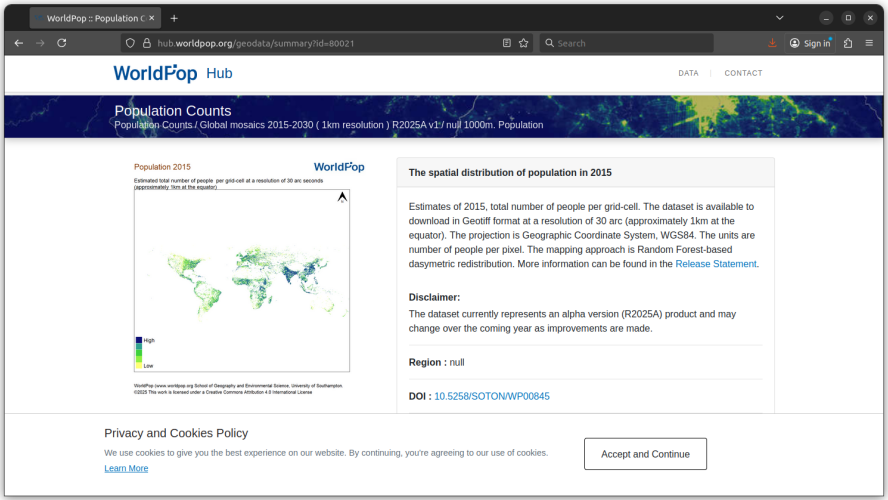
Let's grab the global mosaic file.



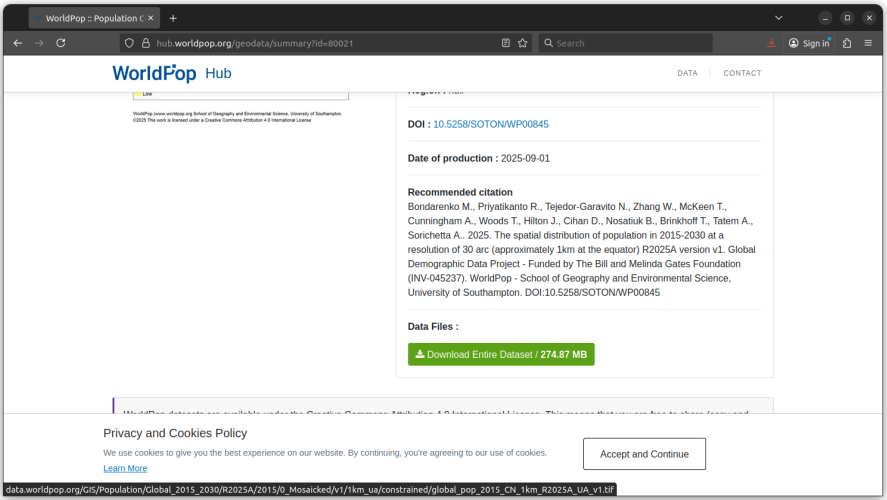
The earliest population raster they have is from 2015 (1 year after IS campaign began). Suboptimal, but we'll take it. Click on `Data and Resources`.



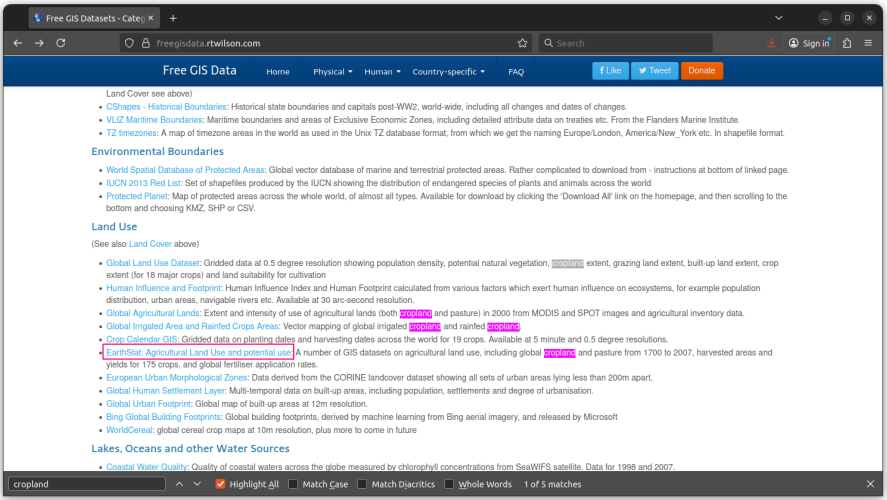
Here we see a preview of the data and, if we scroll down. . .



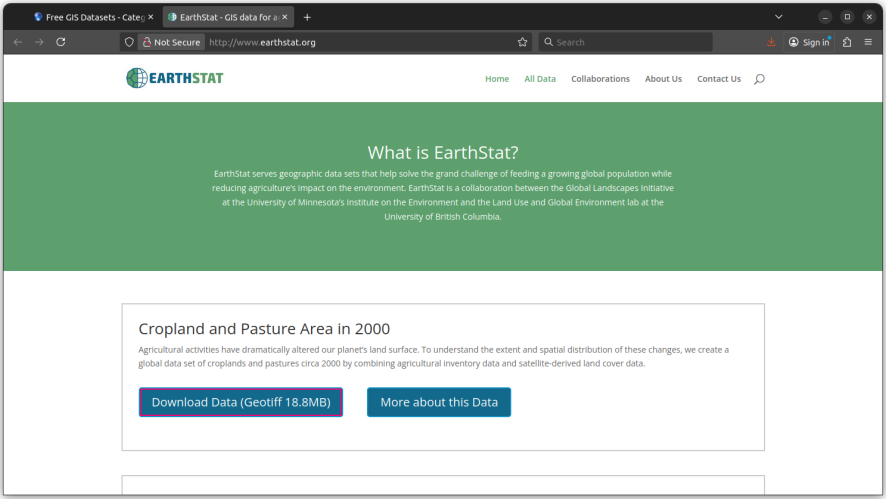
. . . a link to download the data.



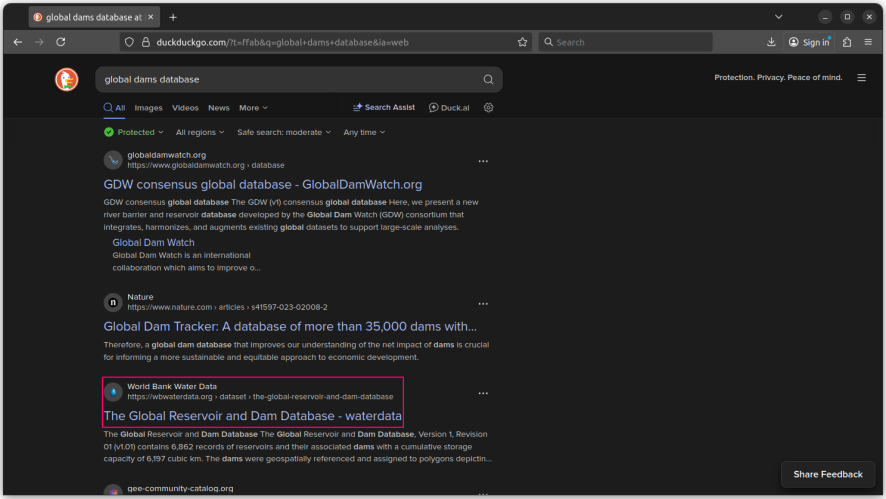
Also at the RT Wilson page, we can find data on **cropland** (e.g., "Earthstat")



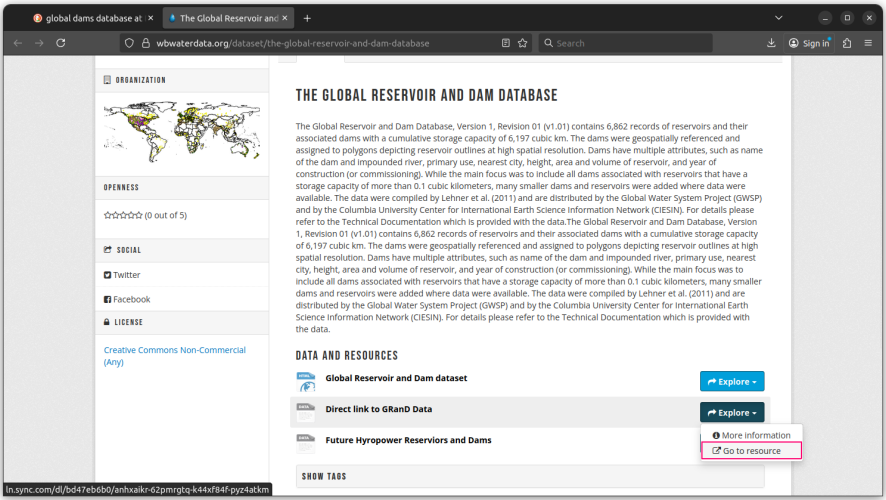
The first dataset we see here is a global raster from 2000. This pre-dates the IS campaign (good) by about 14 years (less good). This likely overstates how much cropland was still active in 2014. Suboptimal, but OK for an exploratory analysis.



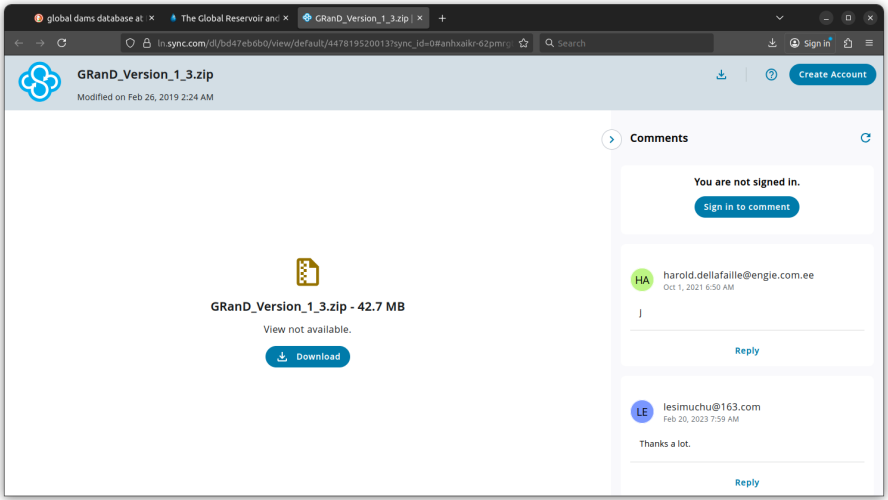
For data on **hydroelectric dams**, we can do a quick web search. This quickly points us to a dataset called "Global Reservoir and Dam (GRanD)".



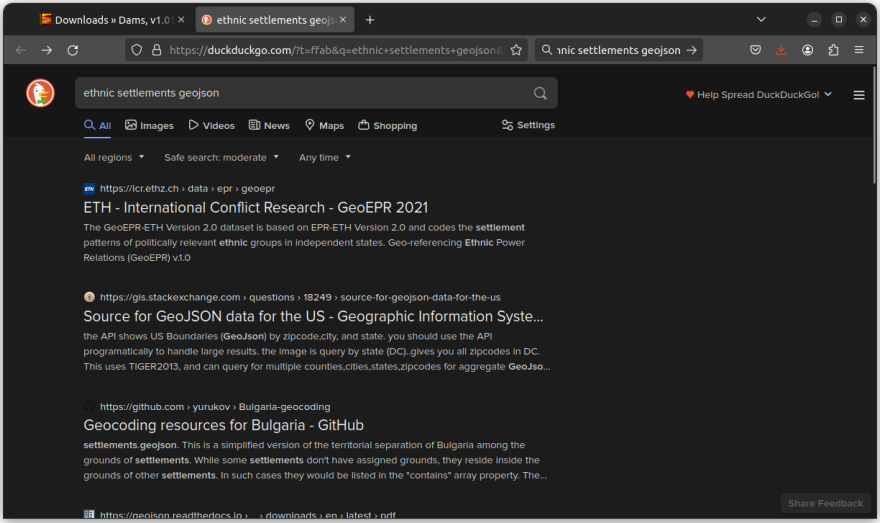
Look for a link to download the data.



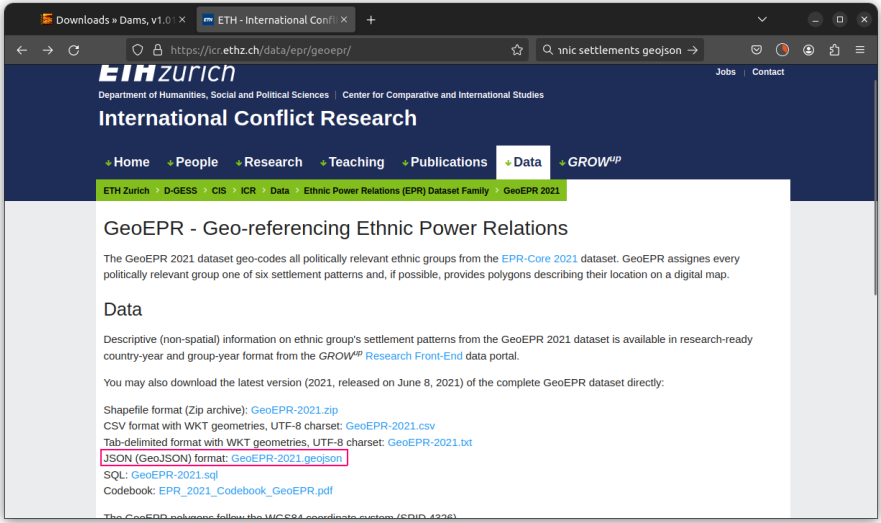
Seems legit... (scan ZIPs for viruses before extracting)



For data on **sectarian divisions**, a relatively reliable global source is the "Ethnic Power Relations" (geoEPR) dataset — we used this in Problem Set 02



Specifically, we will download the `GeoEPR-2021.geojson` file, which contains polygons for each ethno-religious group (`icr.ethz.ch/data/epr/geoepr`)



Here is the full list of data sources and links:

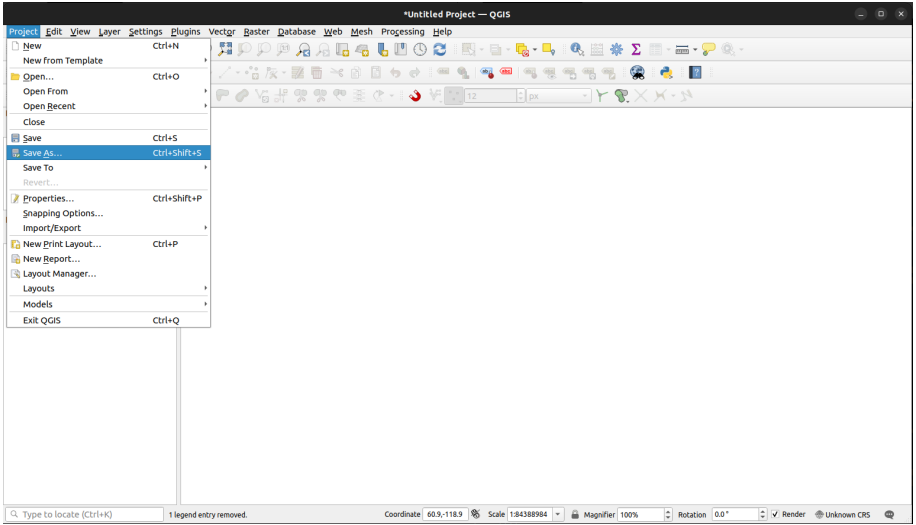
Category	Type	Format	Data source
ISIS violence	Table (non-geo)	`.csv`	UCDP GED
Administrative units	Vector (polygons)	`.geojson`	GADM
Roads	Vector (polylines)	`.shp`	DIVA-GIS
Population	Raster	`.tif`	WorldPop
Cropland	Raster	`.tif`	EarthStat
Dams	Vector (points)	`.shp`	GRanD
Sectarian divisions	Vector (polygons)	`.geojson`	ETH-Zurich

These are all in the `WT01.zip` file posted on Canvas.

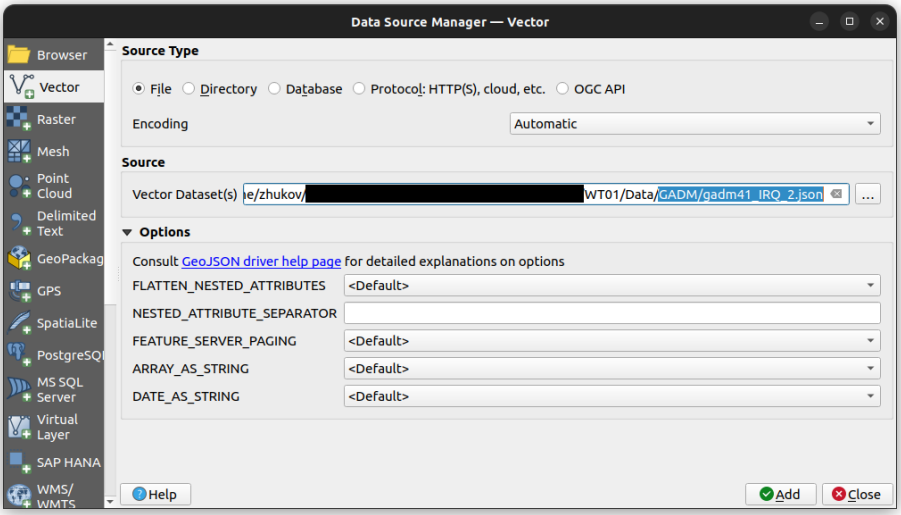
Pre-processing

Always save your progress!

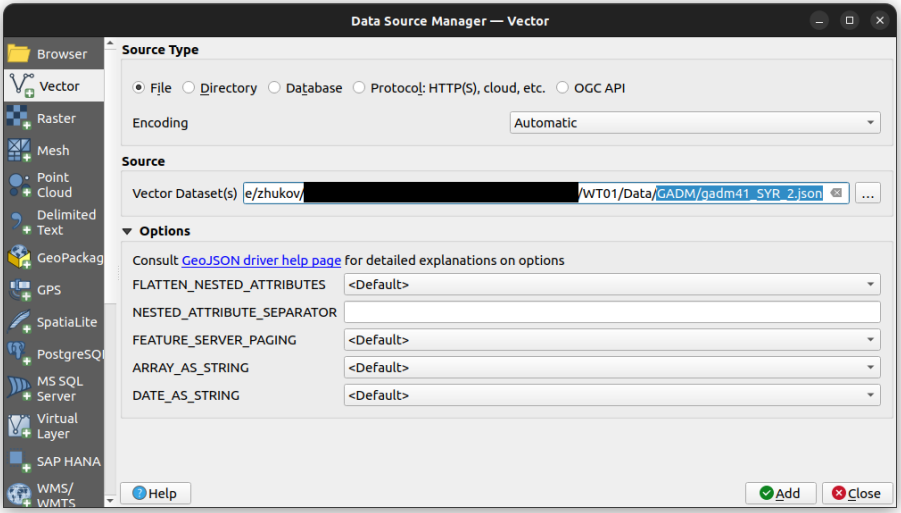
Go to `Project` → `Save As...`



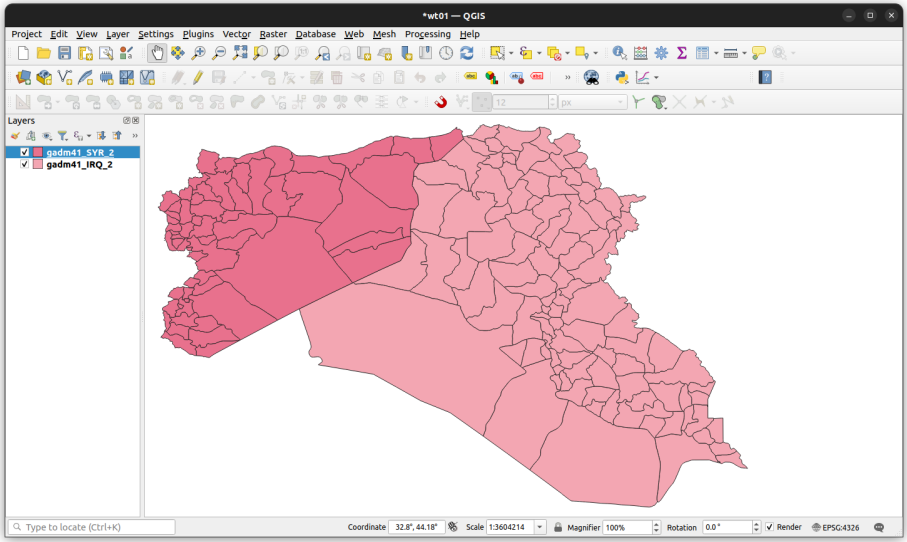
Load *administrative boundaries* (`Layer` → `Add Layer` → `Add Vector Layer`).
For **Iraq**: `gadm41_IRQ_2.json` file in `Data/GADM` folder.



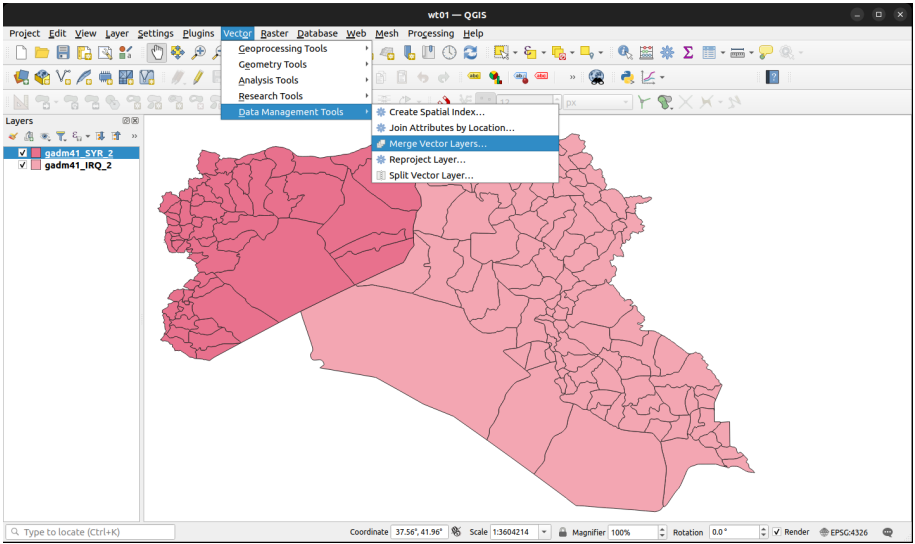
For **Syria**: `gadm41_SYR_2.json` file in `Data/GADM` folder.



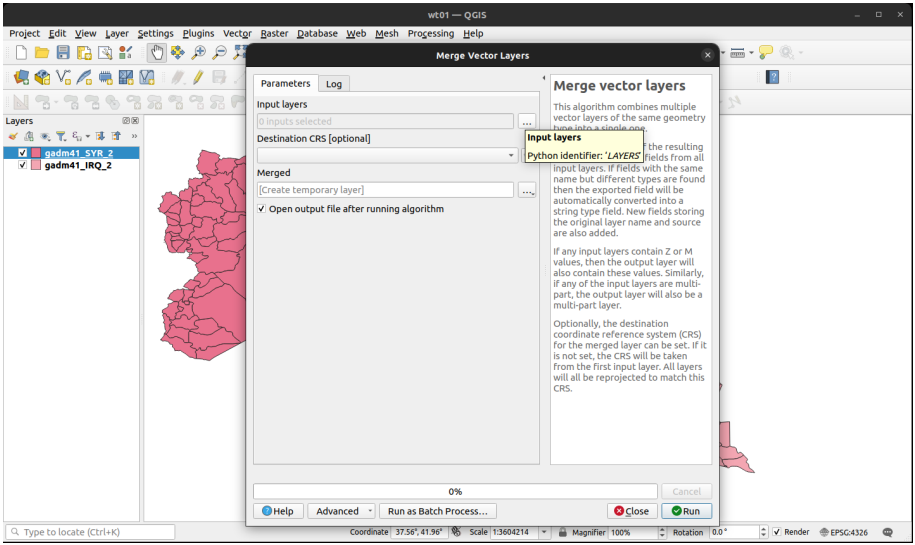
The two sets of district boundaries should be visible in the project window. Let's **merge them into a single layer**, with districts for both countries.



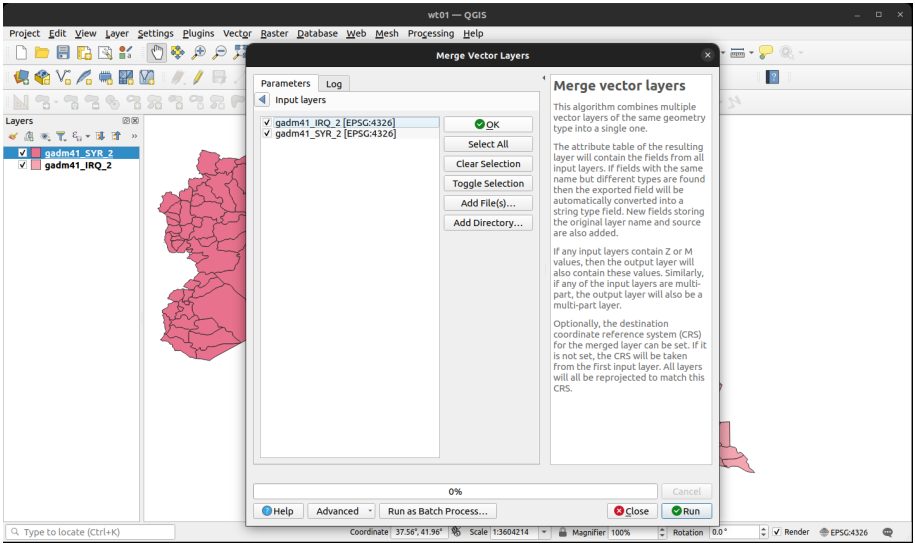
Open the `Merge Vector Layers` tool (`Vector` → `Data Management Tool` → `Merge Vector Layers`).



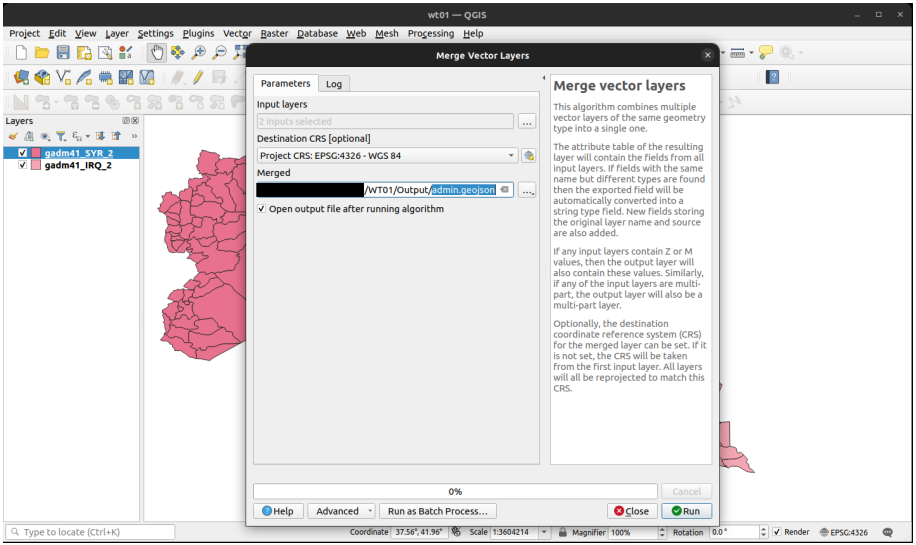
Click on the `[...]` box next to `Input layers`



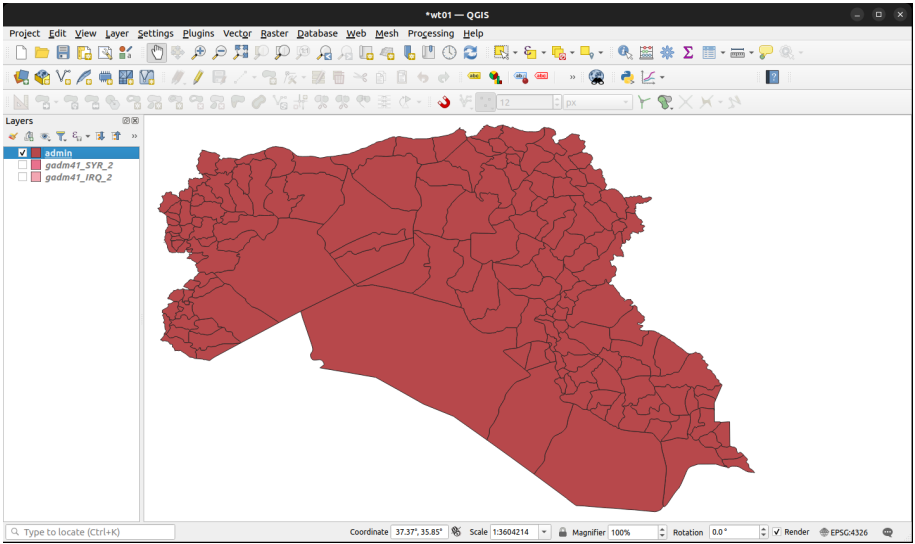
Check the boxes next to gadm41_IRQ_2 and gadm41_SYR_2. Click OK.



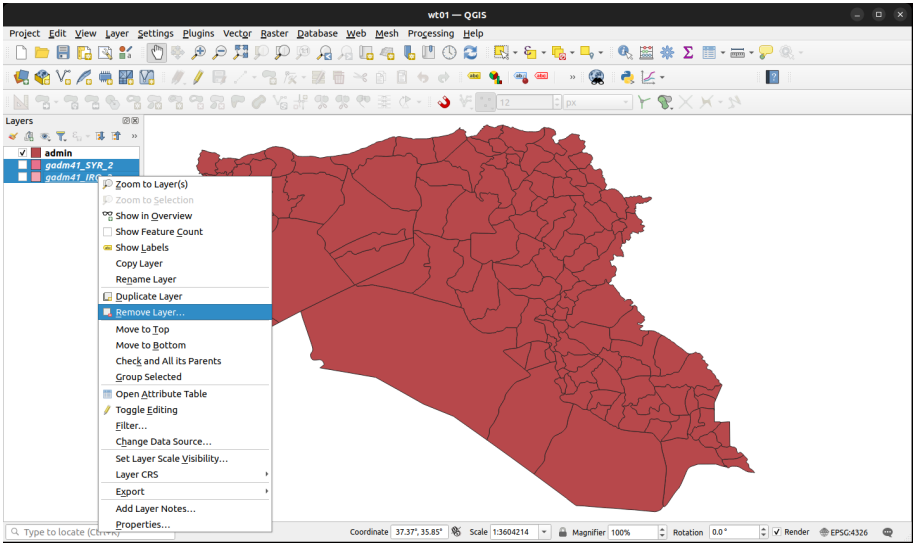
Click on the `[...]` box next to `Merged`. Find a location in which to save the output, and name the file `admin.geojson`. Click `Run`



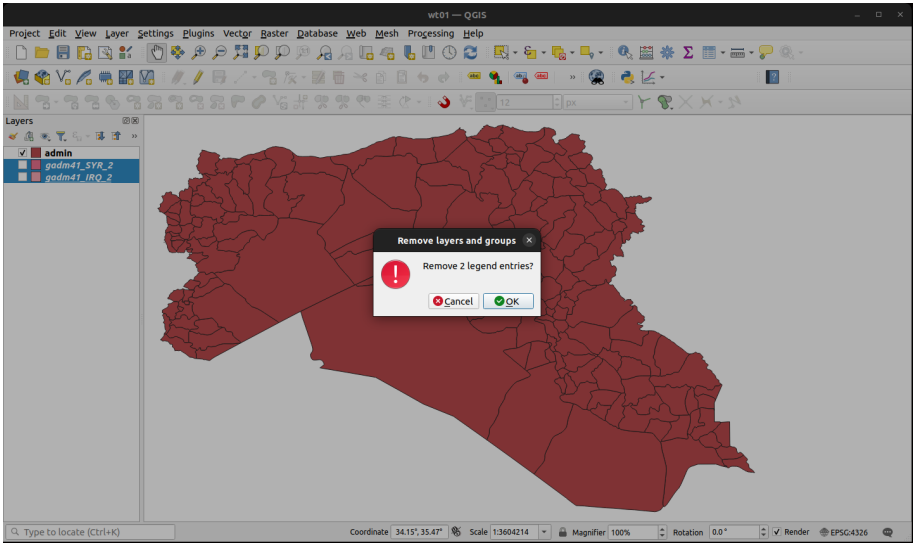
The merged layer should appear in the main project window.



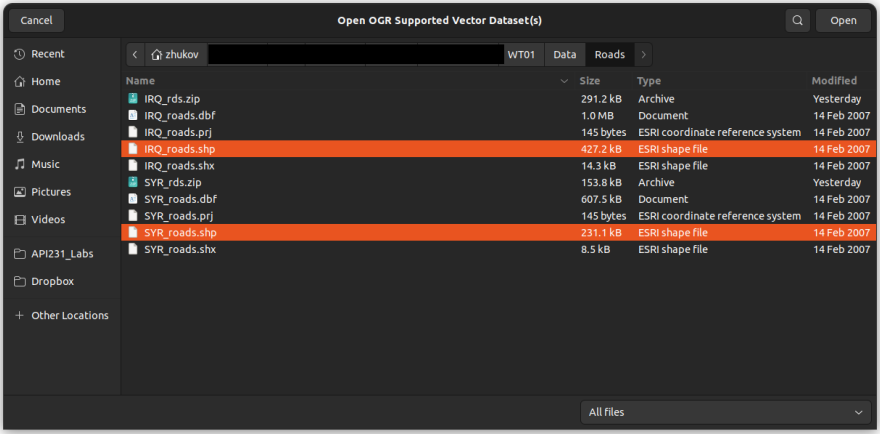
You can remove the `gadm41_IRQ_2` and `gadm41_SYR_2` layers from the project (`Right-click` on layer(s) in Layers menu → `Remove Layer...`)



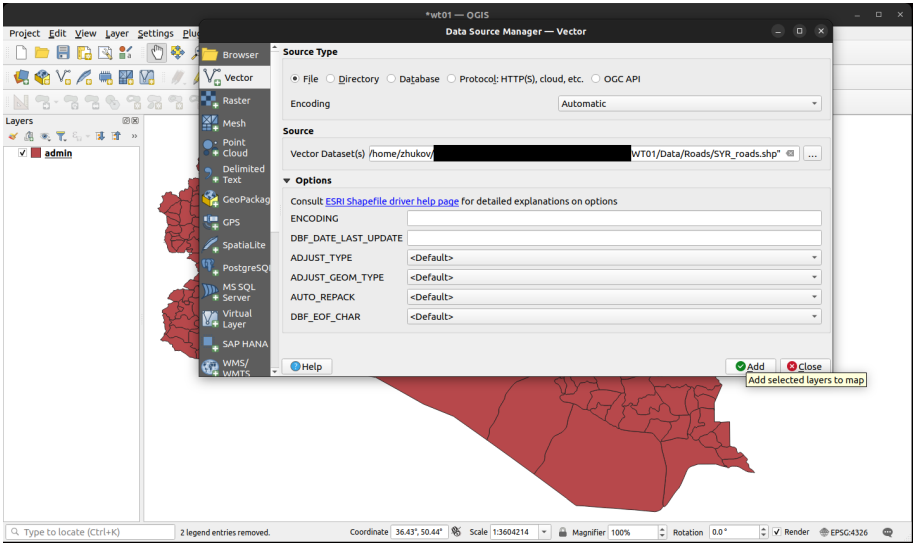
To conserve memory, it's good practice to remove data objects were are not actively using



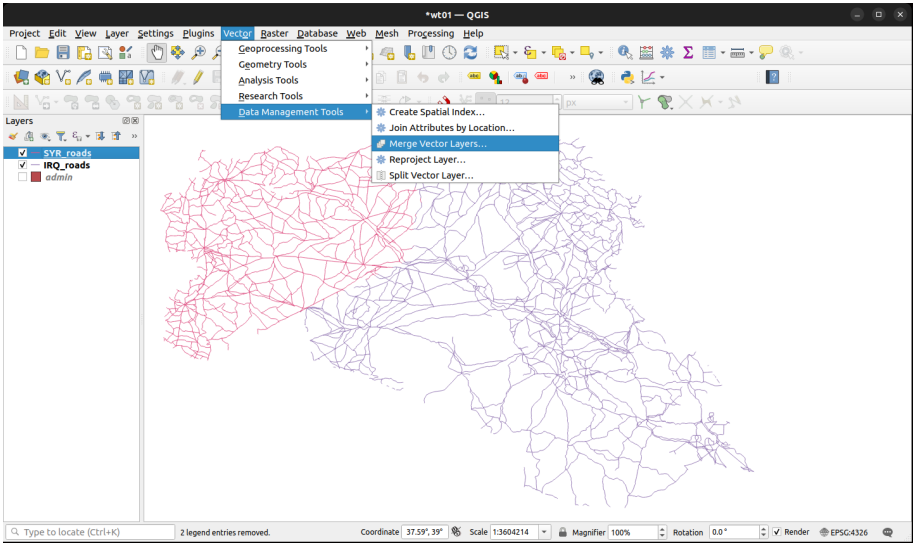
Load *roads data* (`Layer` → `Add Layer` → `Add Vector Layer`).
Try opening **two datasets at the same time**: `IRQ_roads.shp` and `SYR_roads.shp` files in `Data/Roads`



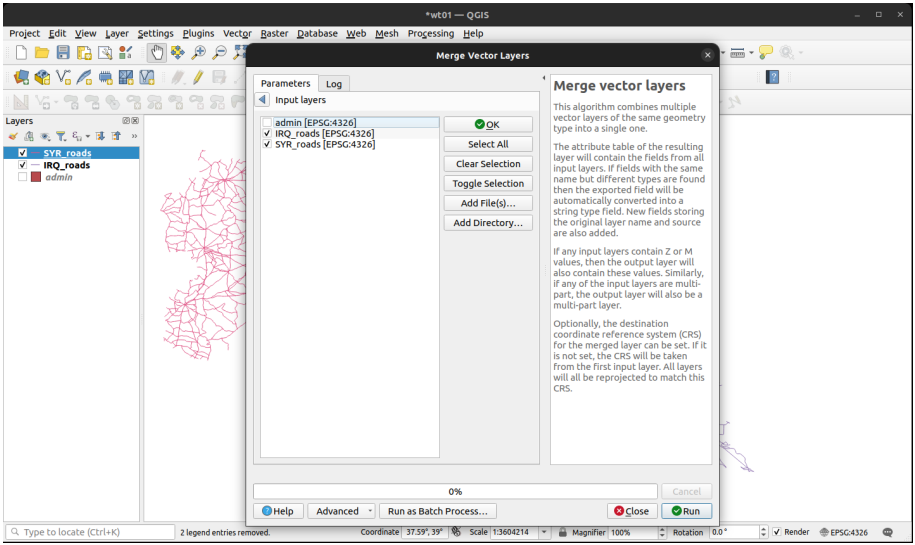
Paths to both files should appear in the `Source` box. Click `Add`



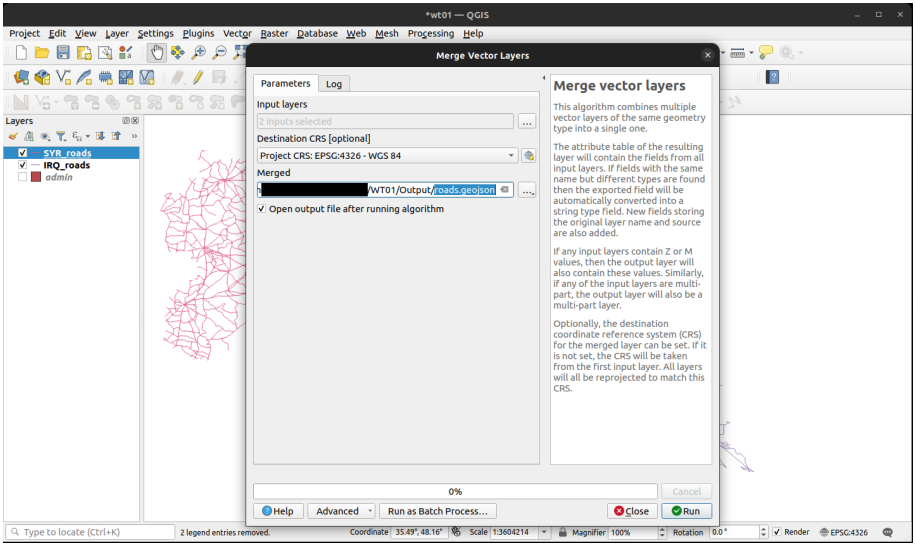
Let's repeat the `Merge Vector Layers` procedure for these two layers (`Vector` → `Data Management Tool` → `Merge Vector Layers`).



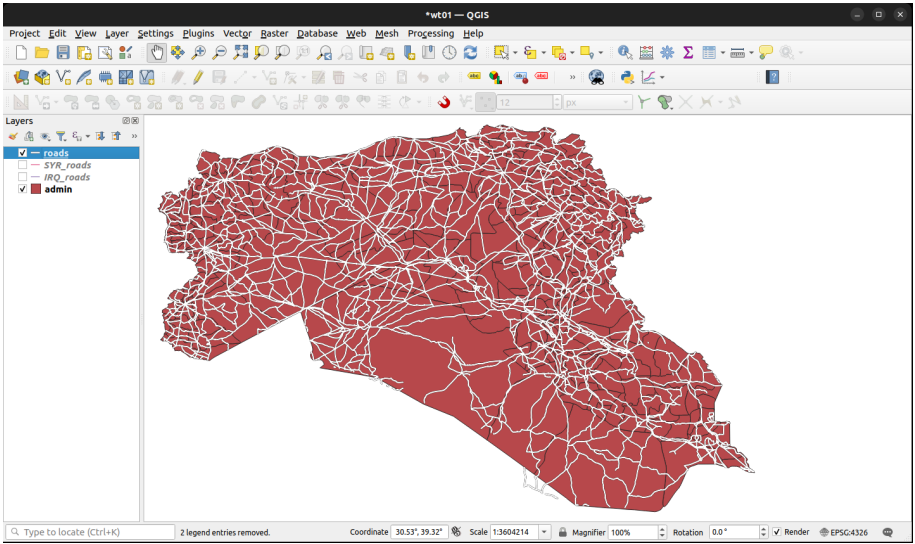
This time, perform the merge on the `IRQ_roads` and `SYR_roads` layers



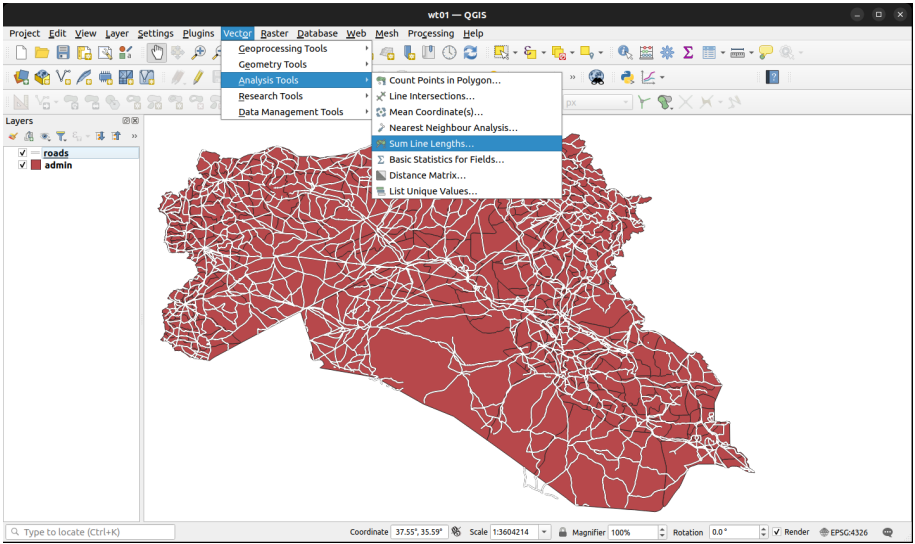
Save the output file as `roads.geojson`



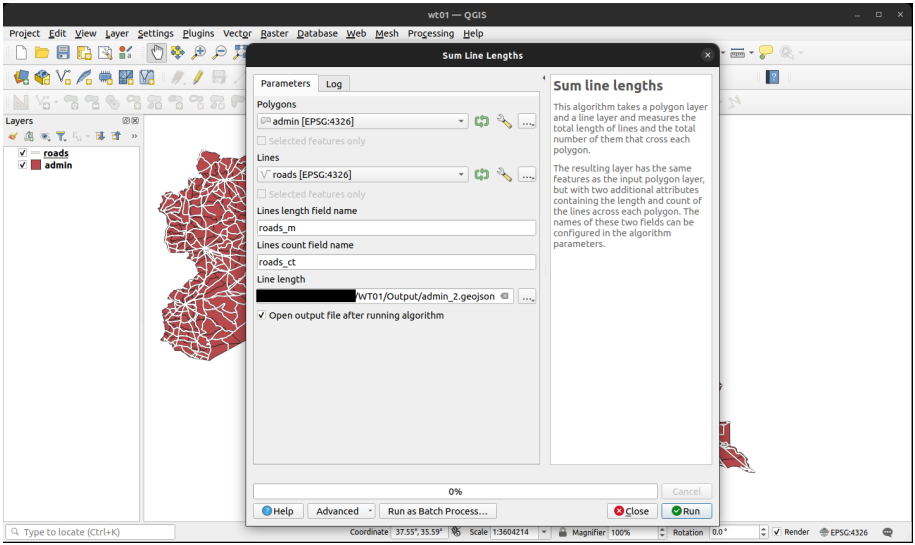
As before, you can remove the two country-specific roads layers (we no longer need them)



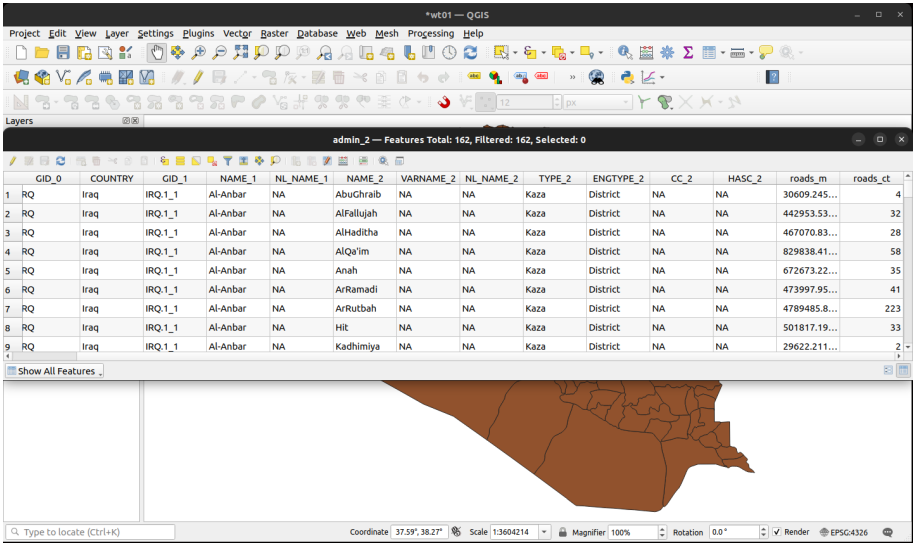
Let's **calculate each district's road density** with the `Sum Line Lengths` tool (`Vector` → `Analysis` → `Sum Line Lengths`)



Set `Polygons` = `admin`, `Lines` = `roads`, `Line length field name` = `roads_m`, `Lines count field name` = `roads_ct`. Save as `admin_2.geojson`

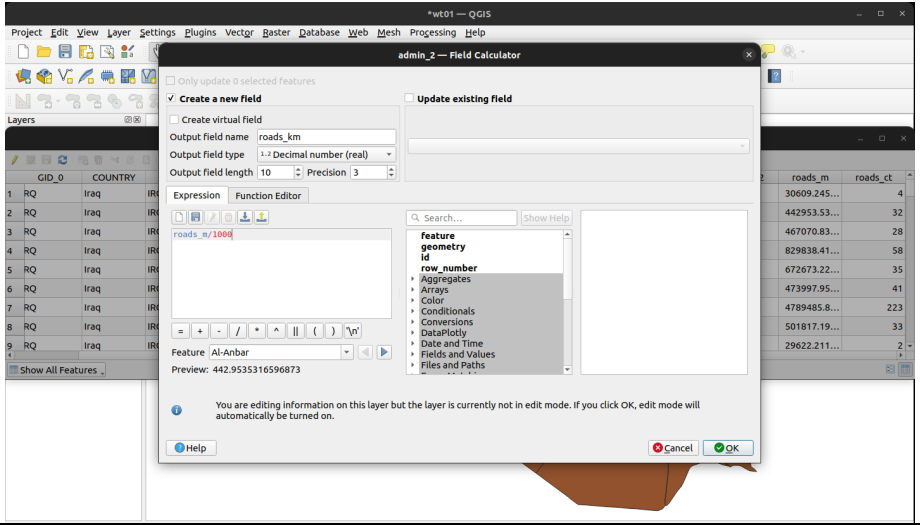


Open the attribute table for `admin_2`. The `roads_m` and `roads_ct` variables should be in the table. Let's convert from **meters to kilometers**

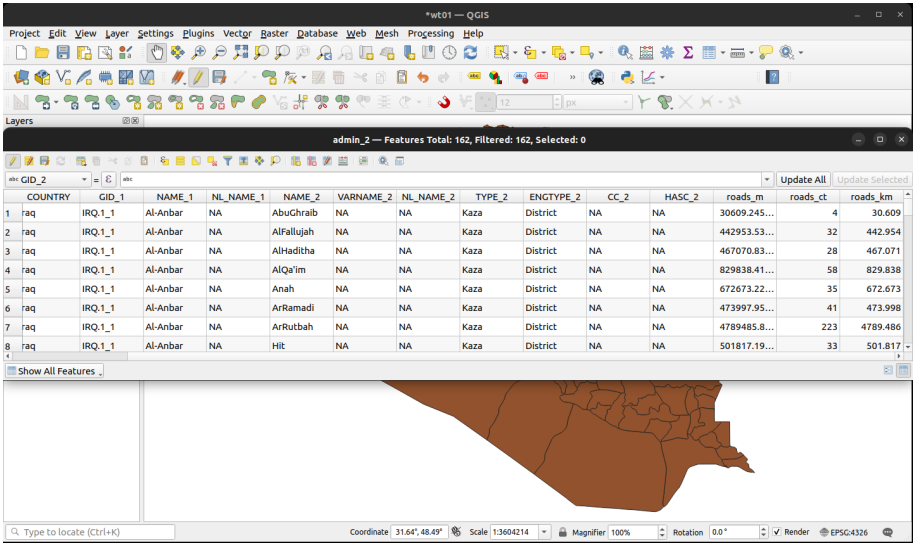


	GID_0	COUNTRY	GID_1	NAME_1	NL_NAME_1	NAME_2	VARNAME_2	NL_NAME_2	TYPE_2	ENGTYPE_2	CC_2	HASC_2	roads_m	roads_ct
1	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	AbuGhraib	NA	NA	Kaza	District	NA	NA	30609.245…	4
2	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	AlFallujah	NA	NA	Kaza	District	NA	NA	442953.53…	32
3	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	AlHaditha	NA	NA	Kaza	District	NA	NA	467070.83…	28
4	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	AlQa'im	NA	NA	Kaza	District	NA	NA	829838.41…	58
5	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	Anah	NA	NA	Kaza	District	NA	NA	672673.22…	35
6	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	ArRamadi	NA	NA	Kaza	District	NA	NA	473997.95…	41
7	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	ArRutbah	NA	NA	Kaza	District	NA	NA	4789485.8…	223
8	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	Hit	NA	NA	Kaza	District	NA	NA	501817.19…	33
9	RQ	Iraq	IRQ.1_1	Al-Anbar	NA	Kadhimiya	NA	NA	Kaza	District	NA	NA	29622.211…	2

Open `Field Calculator` for `admin_2`. Create new field, `roads_km`, of type `Decimal number (real)`. For the `Expression`, write `roads_m/1000`. The `Output preview` should show a number with decimals. Click `OK`

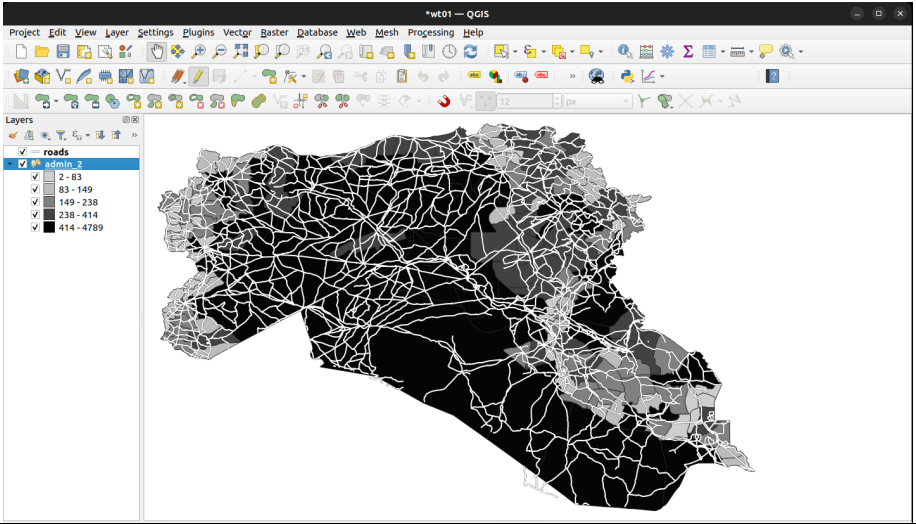


The resulting `roads_km` field should be visible in the attribute table

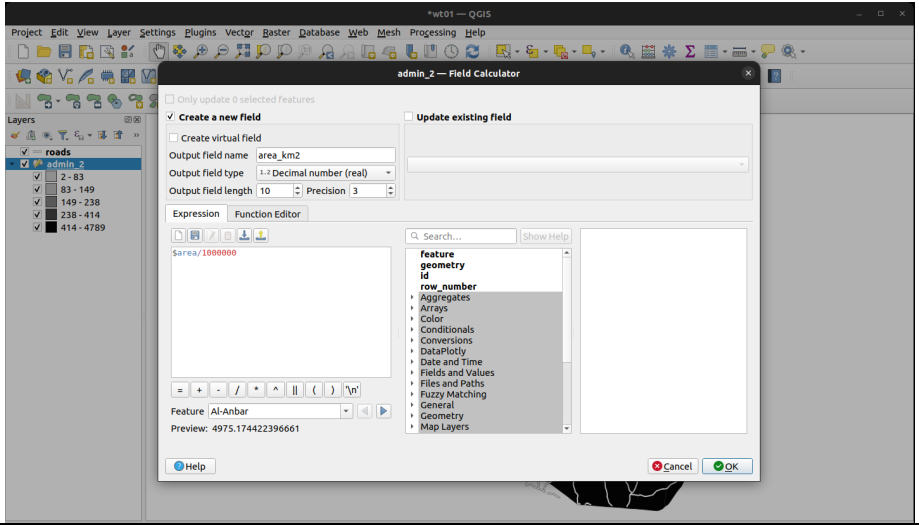


	COUNTRY	GID_1	NAME_1	NL_NAME_1	NAME_2	VARNAME_2	NL_NAME_2	TYPE_2	ENGTYPE_2	CC_2	HASC_2	roads_m	roads_ct	roads_km
1	raq	IRQ.1_1	Al-Anbar	NA	AbuGhraib	NA	NA	Kaza	District	NA	NA	30609.245…	4	30.609
2	raq	IRQ.1_1	Al-Anbar	NA	AlFallujah	NA	NA	Kaza	District	NA	NA	442953.53…	32	442.954
3	raq	IRQ.1_1	Al-Anbar	NA	AlHaditha	NA	NA	Kaza	District	NA	NA	467070.83…	28	467.071
4	raq	IRQ.1_1	Al-Anbar	NA	AlQa'im	NA	NA	Kaza	District	NA	NA	829838.41…	58	829.838
5	raq	IRQ.1_1	Al-Anbar	NA	Anah	NA	NA	Kaza	District	NA	NA	672673.22…	35	672.673
6	raq	IRQ.1_1	Al-Anbar	NA	ArRamadi	NA	NA	Kaza	District	NA	NA	473997.95…	41	473.998
7	raq	IRQ.1_1	Al-Anbar	NA	ArRutbah	NA	NA	Kaza	District	NA	NA	4789485.8…	223	4789.486
8	raq	IRQ.1_1	Al-Anbar	NA	Hit	NA	NA	Kaza	District	NA	NA	501817.19…	33	501.817

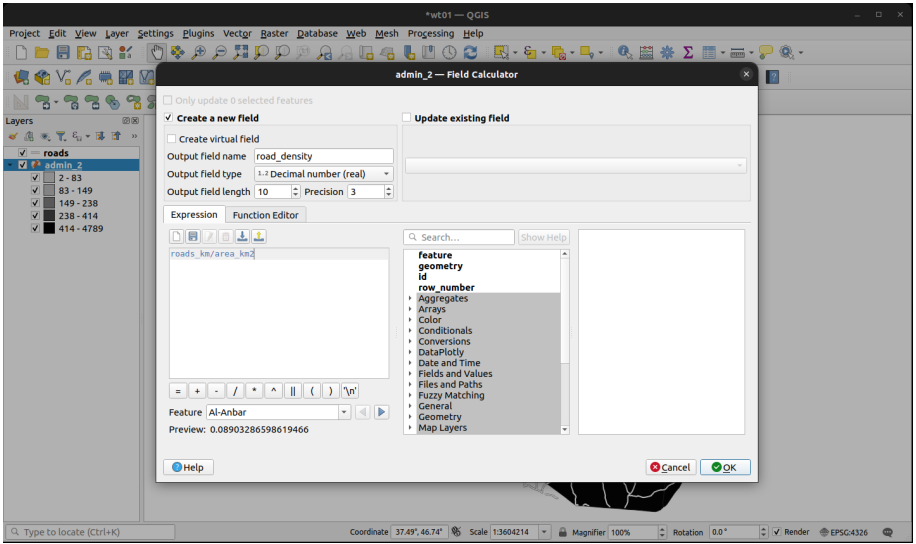
If we plot this variable, we can see its main limitations: smaller districts have less road length, so this becomes a proxy for district size. Let's create a **road density** measure that takes into account district size



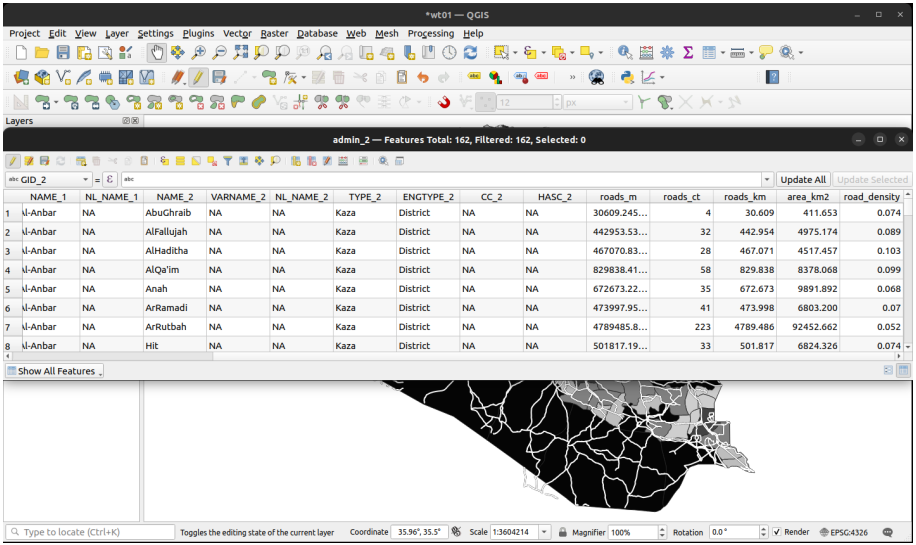
Back in the `Field Calculator`, create a new field called `area_km2` of type `Decimal number (real)`. For `Expression`, write `$area/1000000` (divide by 1M to convert from m^2 to km^2). Click `OK`



Open the `Field Calculator` again. Create a new field called `road_density` of type `Decimal number (real)`. For `Expression`, write `roads_km/area_km2`

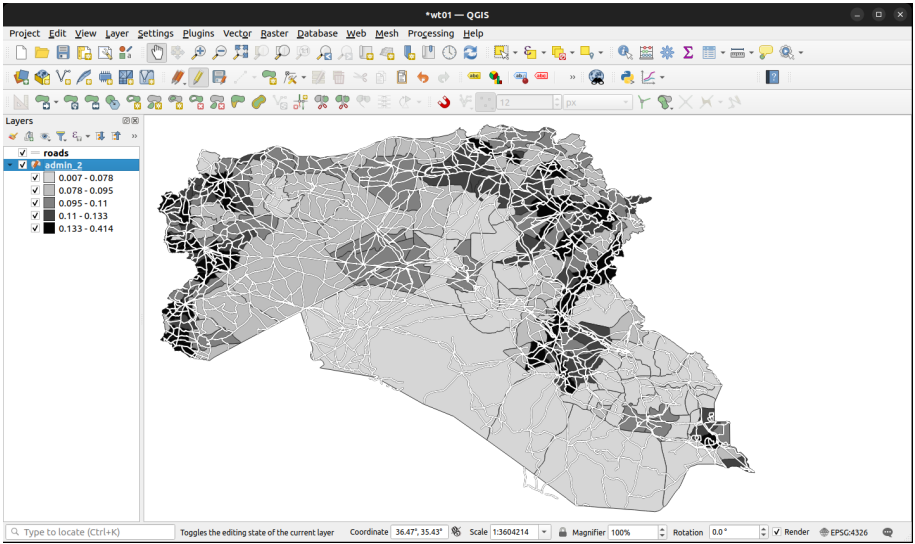


Check to make sure `road_density` is added to attribute table

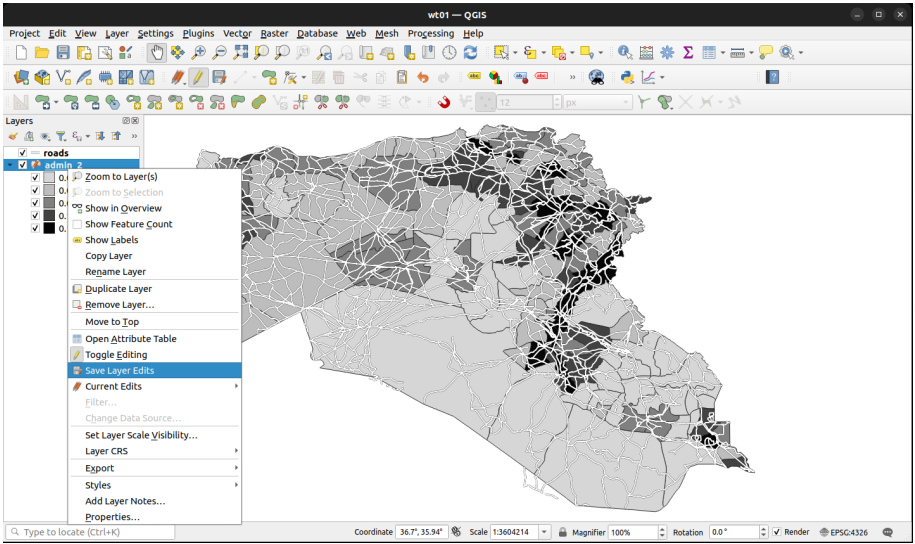


	NAME_1	NL_NAME_1	NAME_2	VARNAME_2	NL_NAME_2	TYPE_2	ENGTYPE_2	CC_2	HASC_2	roads_m	roads_ct	roads_km	area_km2	road_density
1	al-Anbar	NA	AbuGhraib	NA	NA	Kaza	District	NA	NA	30609.245…	4	30.609	411.653	0.074
2	al-Anbar	NA	AlFallujah	NA	NA	Kaza	District	NA	NA	442953.53…	32	442.954	4975.174	0.089
3	al-Anbar	NA	AlHaditha	NA	NA	Kaza	District	NA	NA	467070.83…	28	467.071	4517.457	0.103
4	al-Anbar	NA	AlQa'im	NA	NA	Kaza	District	NA	NA	829838.41…	58	829.838	8378.068	0.099
5	al-Anbar	NA	Anah	NA	NA	Kaza	District	NA	NA	672673.22…	35	672.673	9891.892	0.068
6	al-Anbar	NA	ArRamadi	NA	NA	Kaza	District	NA	NA	473997.95…	41	473.998	6803.200	0.07
7	al-Anbar	NA	ArRutbah	NA	NA	Kaza	District	NA	NA	4789485.8…	223	4789.486	92452.662	0.052
8	al-Anbar	NA	Hit	NA	NA	Kaza	District	NA	NA	501817.19…	33	501.817	6824.326	0.074

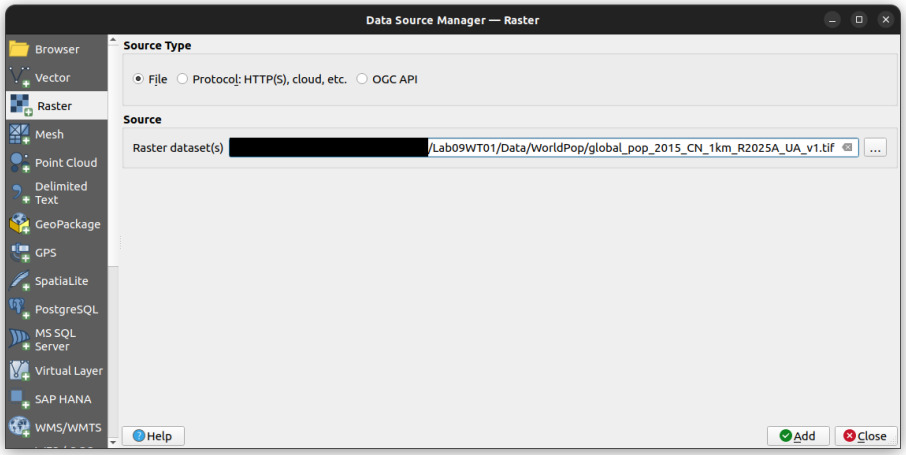
If we plot this variable, we see the opposite pattern from before (smaller, urban districts have higher road density)



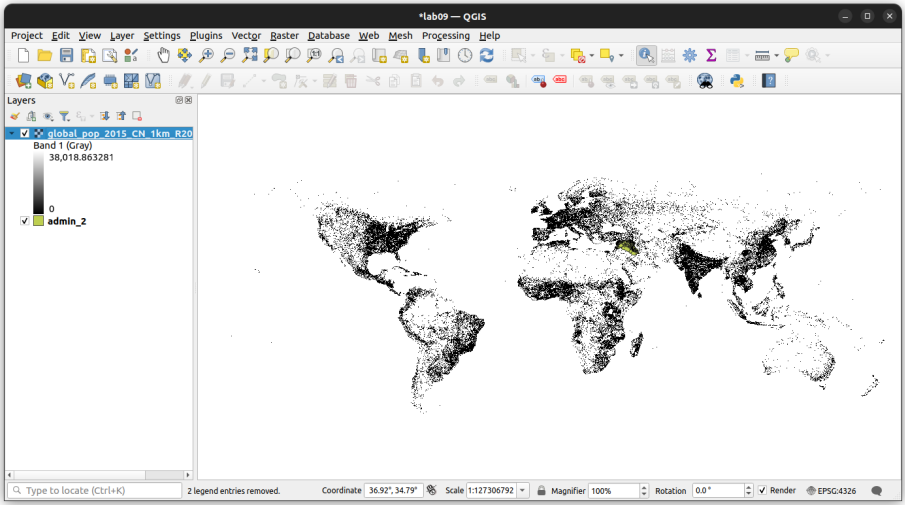
Save the edits you just made to the `admin_2` layer (right click → `Save Layer Edits`)



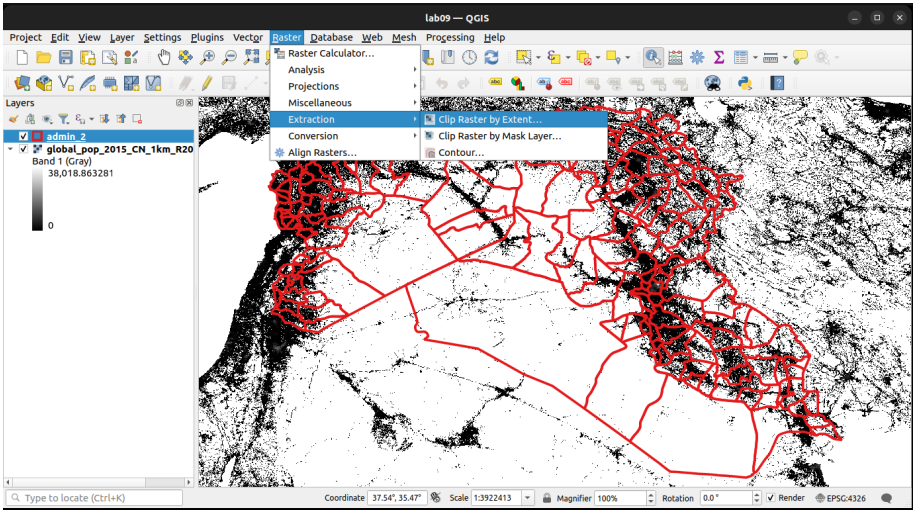
Load *population* raster (`Layer` → `Add Layer` → `Add Raster Layer`).\ Open the `global_pop_2015_CN_1km_R2025A_UA_v1.tif` file in `Data/WorldPop` folder.



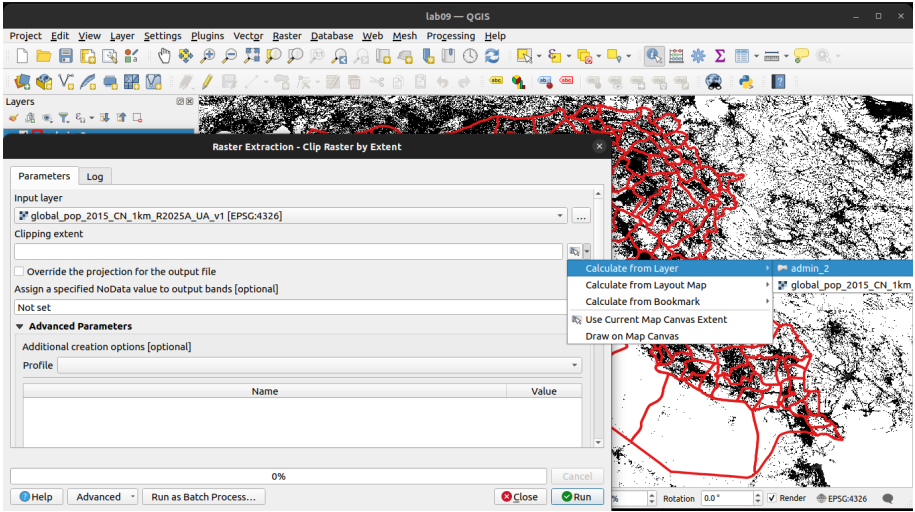
The (global) population layer should now be visible. Let's **extract a subset** of just the part of this layer that overlaps with Syria and Iraq.



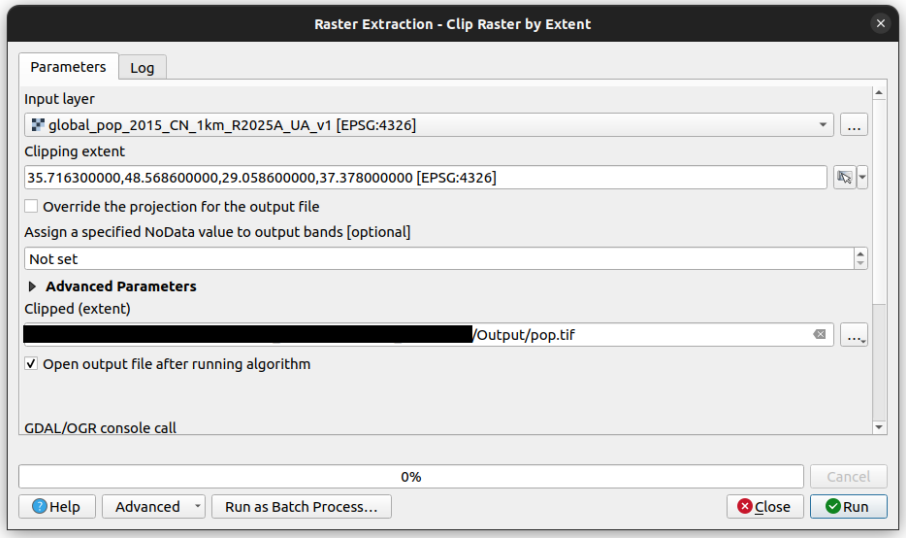
Open the `Clip Raster by Extent` tool (`Raster` → `Extraction` → `Clip Raster by Extent`)



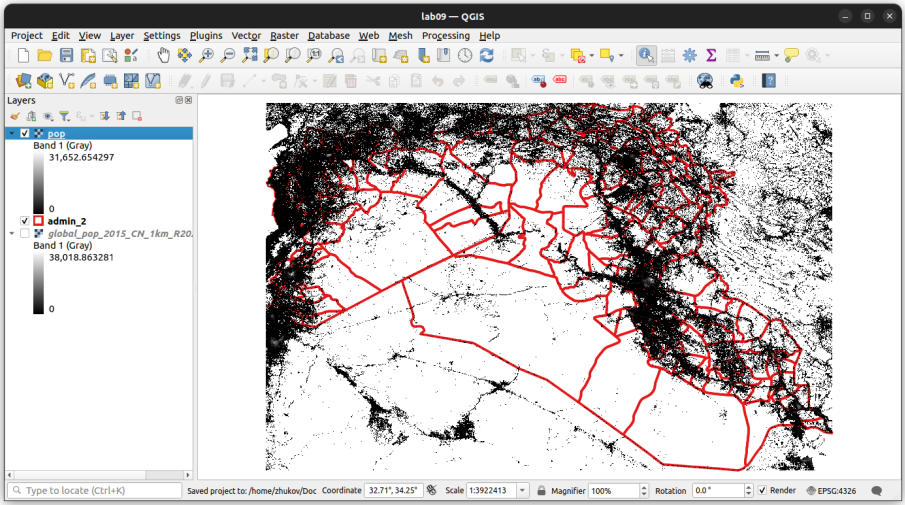
Set `Input layer` = `gpw_v4_population_count_rev11_2010_15_min`. Click the `[...]` button next to `Clipping extent` → `Calculate from Layer` → `admin_2`



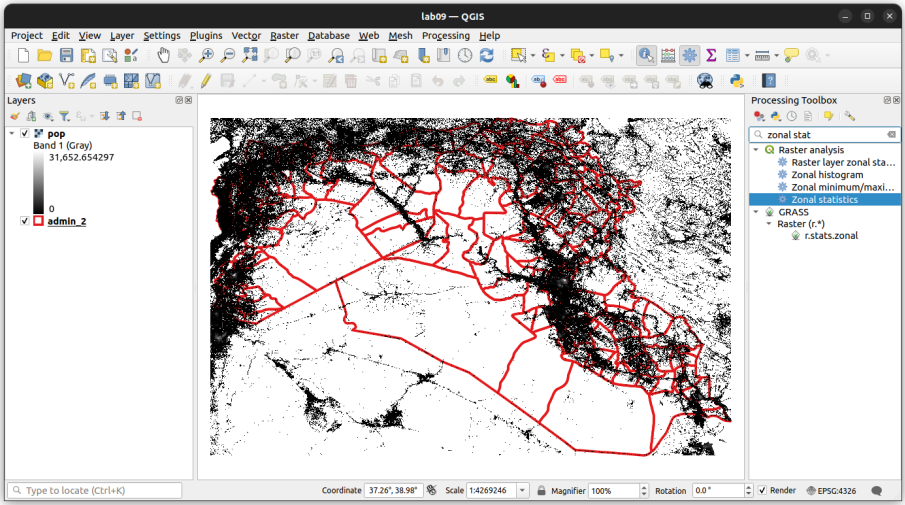
Save the file as `pop.tif`. Accept defaults for all other parameters, and click `Run`



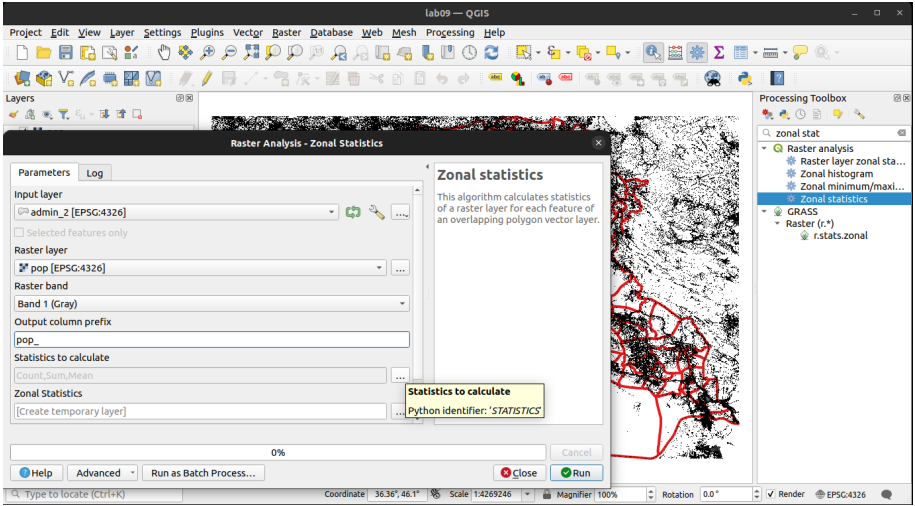
A new layer called pop should appear. You can remove the original, global `gpw_v4...` layer



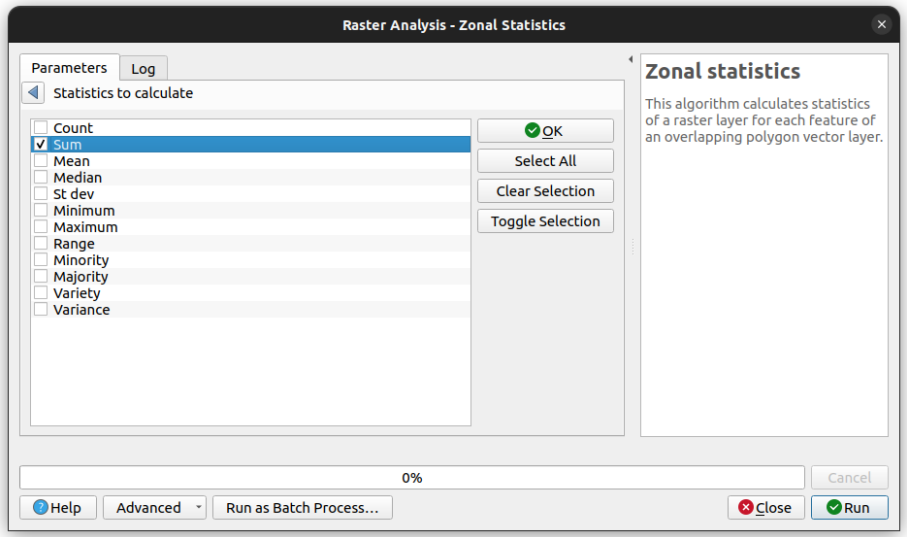
Let's **calculate the population size of each district**. Open the `Processing Toolbox` and open the `Zonal statistics` tool in the `Raster Analysis` menu



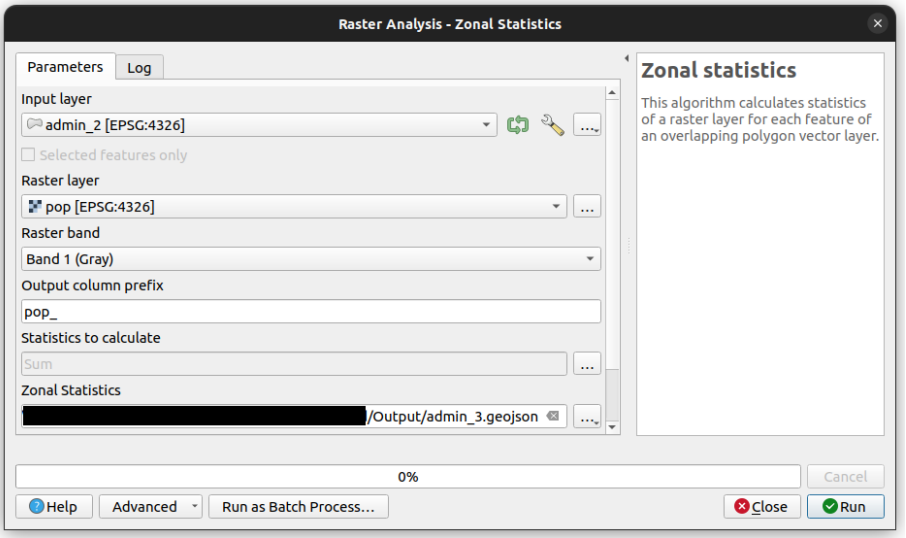
In the `Zonal Statistics` tool, set `Input layer=admin_2`, `Raster layer=pop`, and `Output column prefix=pop_`.
Click the `[...]` button next to `Statistics to calculate`



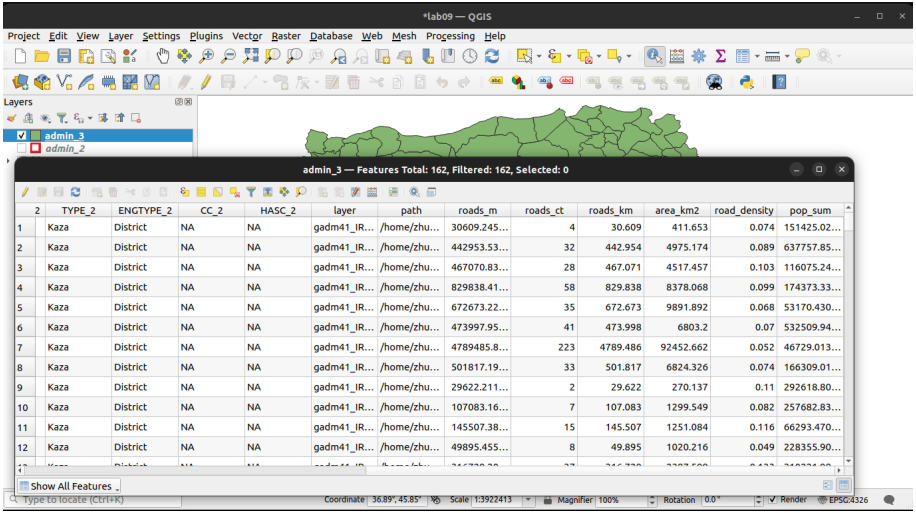
Check the box next to `Sum`. Leave all others un-checked. Click `OK`



Save the zonal statistics output as `admin_3.geojson`. Click `Run`

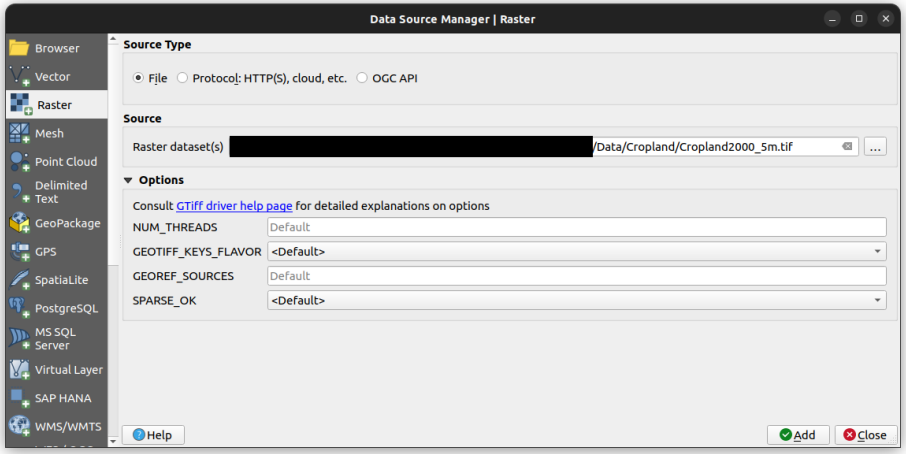


This should have generated a new layer, `admin_3`, with a new field, `pop_sum` (check the layer's attribute table)

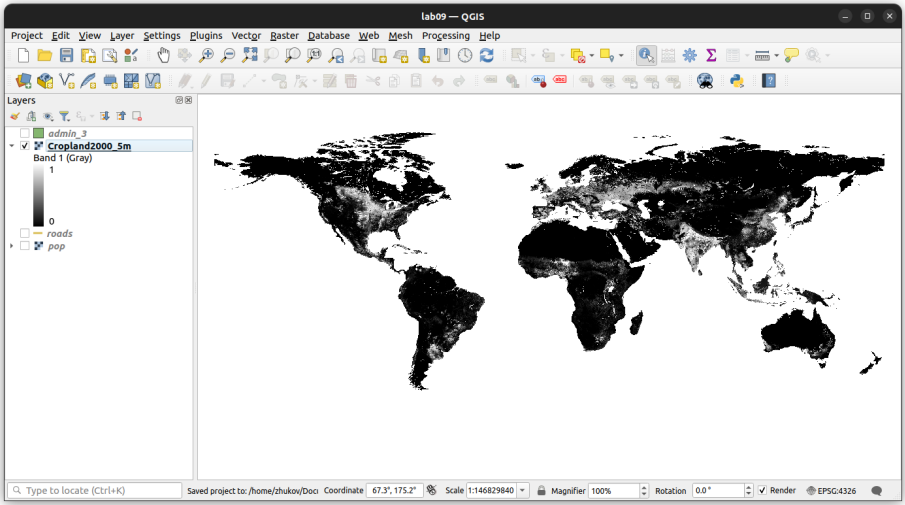


	2	TYPE_2	ENGTYPE_2	CC_2	HASC_2	layer	path	roads_m	roads_ct	roads_km	area_km2	road_density	pop_sum
1		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	30609.245...	4	30.609	411.653	0.074	151425.02...
2		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	442953.53...	32	442.954	4975.174	0.089	637757.85...
3		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	467070.83...	28	467.071	4517.457	0.103	116075.24...
4		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	829838.41...	58	829.838	8378.068	0.099	174373.33...
5		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	672673.22...	35	672.673	9891.892	0.068	53170.430...
6		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	473997.95...	41	473.998	6803.2	0.07	532509.94...
7		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	4789485.8...	223	4789.486	92452.662	0.052	46729.013...
8		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	501817.19...	33	501.817	6824.326	0.074	166309.01...
9		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	29622.211...	2	29.622	270.137	0.11	292618.80...
10		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	107083.16...	7	107.083	1299.549	0.082	257682.83...
11		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	145507.38...	15	145.507	1251.084	0.116	66293.470...
12		Kaza	District	NA	NA	gadm41_IR...	/home/zhu...	49895.455...	8	49.895	1020.216	0.049	228355.90...

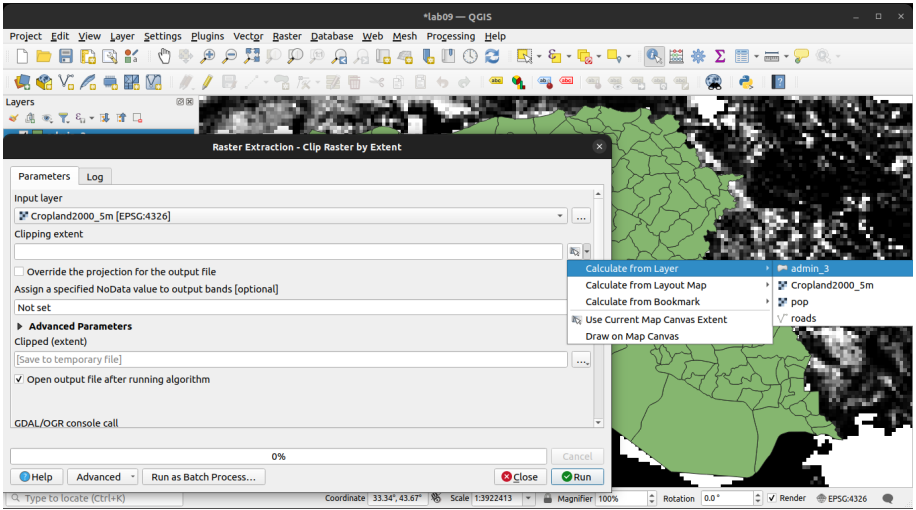
Load *cropland* raster (`Layer` → `Add Layer` → `Add Raster Layer`).
Open the `cropland.tif` file in `Data/Cropland` folder.



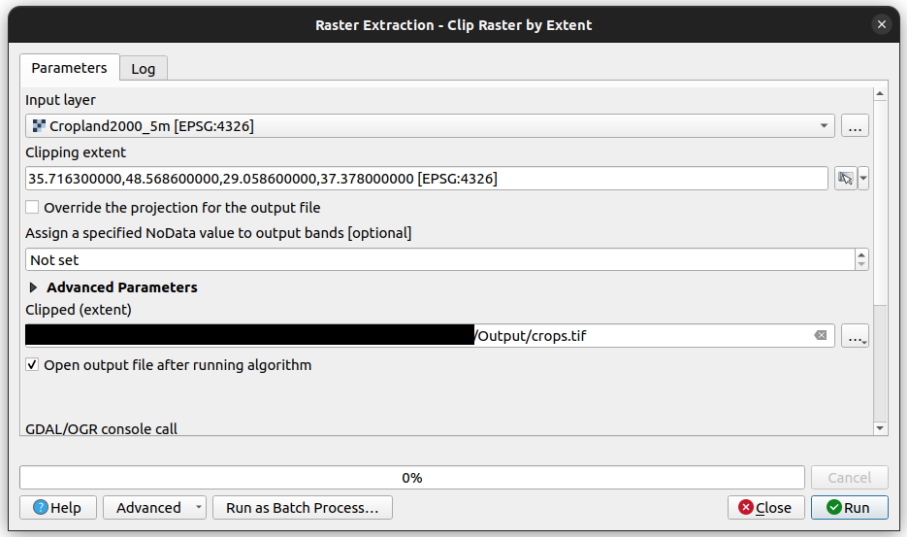
The (global) crops layer should now be visible. Let's **extract a subset** of it, as we just did for population



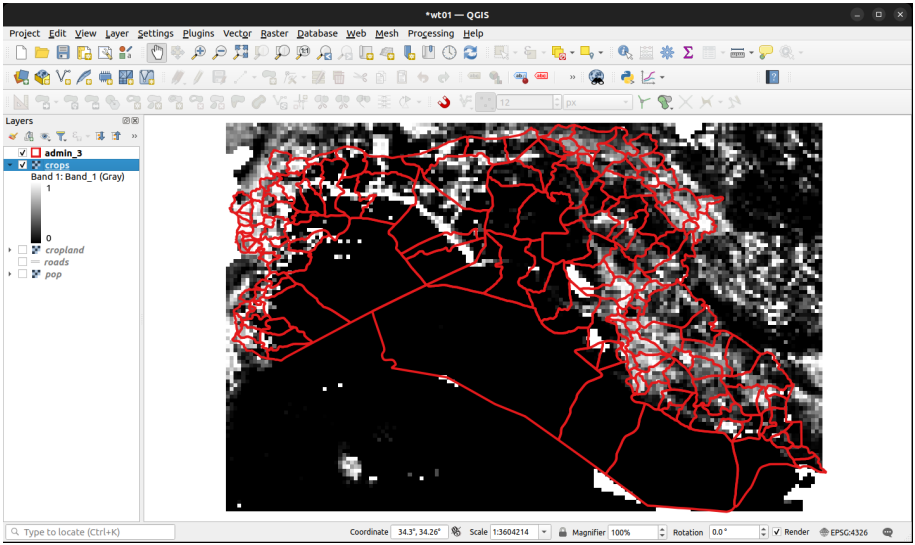
Open the `Clip Raster by Extent` tool (`Raster` → `Extraction` → `Clip Raster by Extent`). Set `Input layer` = `cropland`. Click the `[...]` button next to `Clipping extent` → `Calculate from Layer` → `admin_3`



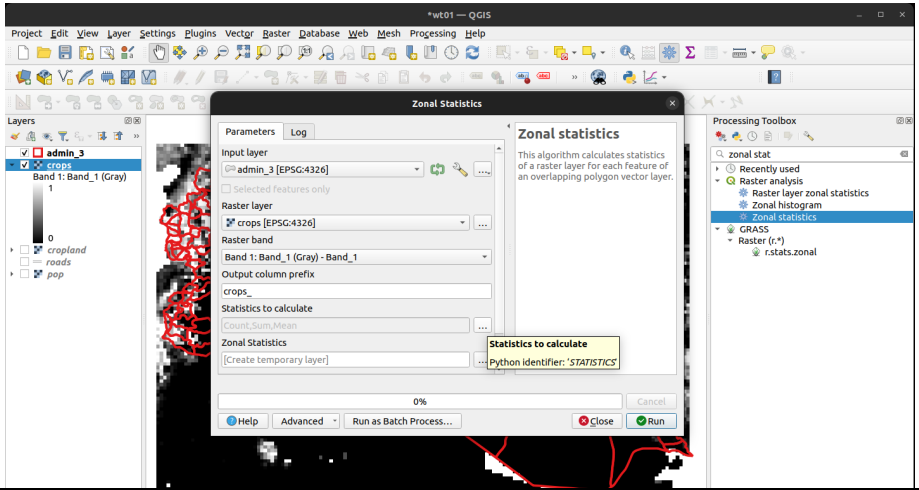
Save the output to a file called `crops.tif`



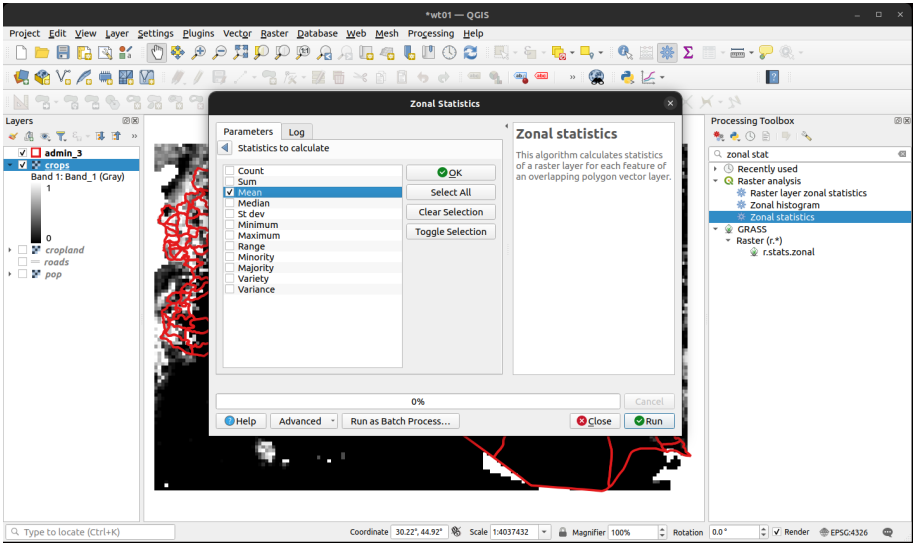
Make sure everything looks right. Remove the original, global `cropland` layer



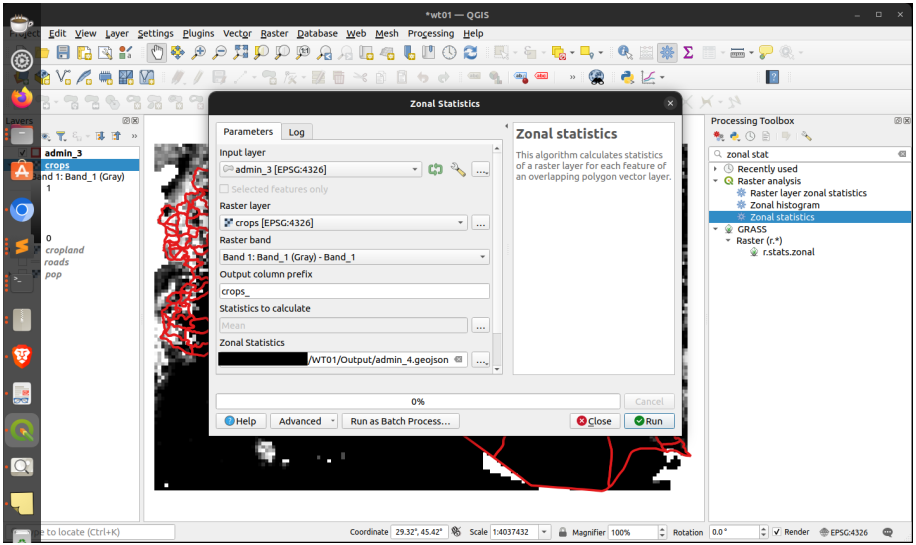
Let's **calculate crop cultivation in each district**. Open the `Zonal statistics` tool in `Processing Toolbox`. Set `Input layer`=`admin_3`, `Raster layer`=`crops`, and `Output column prefix`=`crops_`.
Click the `[...]` button next to `Statistics to calculate`



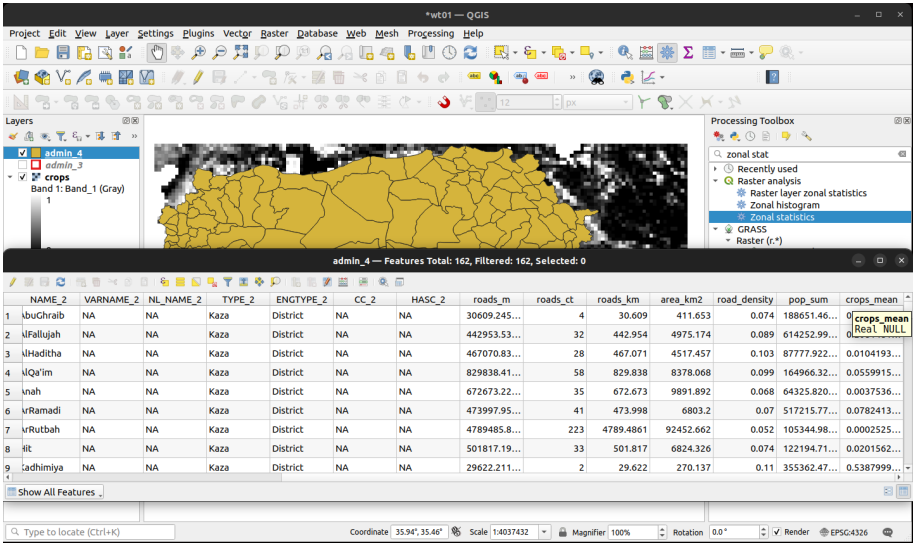
Check the box next to `Mean`. Leave all others un-checked. Click `OK`, then `Run` on the next screen



Save the output to a file called `admin_4.geojson`

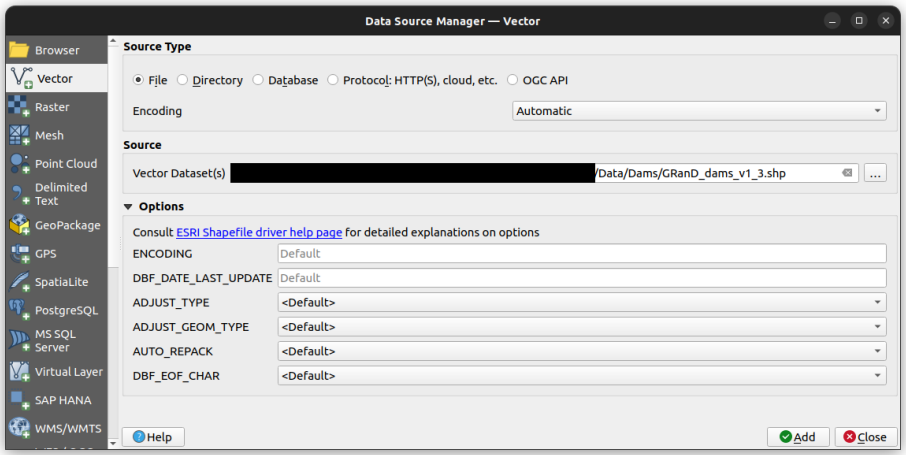


This should have generated a new field, `crops_mean` in the new `admin_4` layer (check the layer's attribute table)

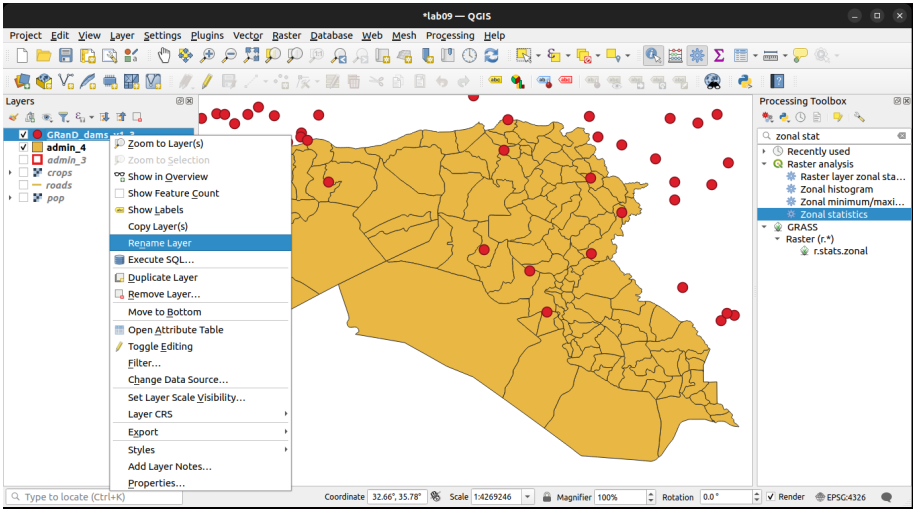


	NAME_2	VARNAME_2	NL_NAME_2	TYPE_2	ENGTYPE_2	CC_2	HASC_2	roads_m	roads_ct	roads_km	area_km2	road_density	pop_sum	crops_mean
1	AbuGhraib	NA	NA	Kaza	District	NA	NA	30609.245...	4	30.609	411.653	0.074	188651.46...	0
2	alFallujah	NA	NA	Kaza	District	NA	NA	442953.53...	32	442.954	4975.174	0.089	614252.99...	0
3	alHaditha	NA	NA	Kaza	District	NA	NA	467070.83...	28	467.071	4517.457	0.103	87777.922...	0.0104193...
4	alQa'im	NA	NA	Kaza	District	NA	NA	829838.41...	58	829.838	8378.068	0.099	164966.32...	0.0559915...
5	Anah	NA	NA	Kaza	District	NA	NA	672673.22...	35	672.673	9891.892	0.068	64325.820...	0.0037536...
6	arRamadi	NA	NA	Kaza	District	NA	NA	473997.95...	41	473.998	6803.2	0.07	517215.77...	0.0782413...
7	arRutbah	NA	NA	Kaza	District	NA	NA	4789485.8...	223	4789.4861	92452.662	0.052	105344.98...	0.0002525...
8	Hit	NA	NA	Kaza	District	NA	NA	501817.19...	33	501.817	6824.326	0.074	122194.71...	0.0201562...
9	Kadhimiya	NA	NA	Kaza	District	NA	NA	29622.211...	2	29.622	270.137	0.11	355362.47...	0.5387999...

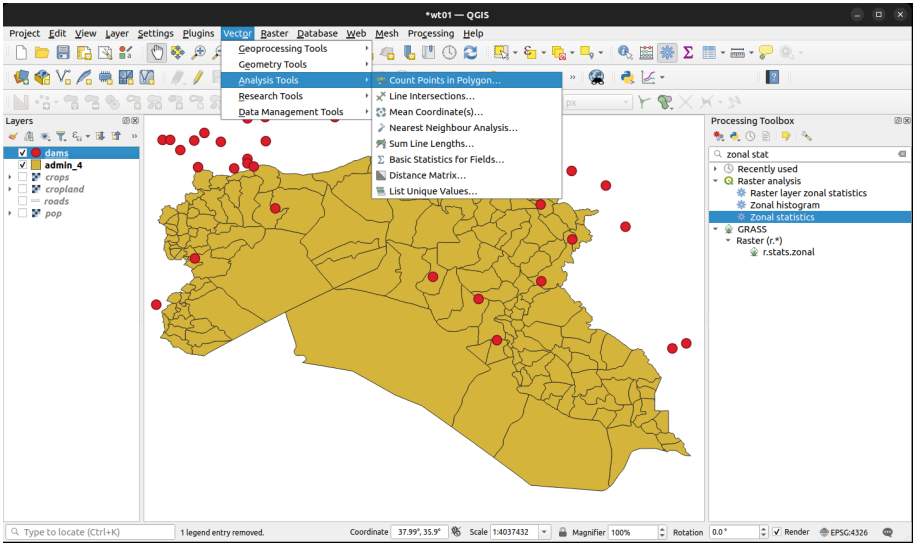
Load *hydroelectric dam* locations (`Layer` → `Add Layer` → `Add Vector Layer`). Open the `GRanD_dams_v1_3.shp` file in `Data/Dams` folder.



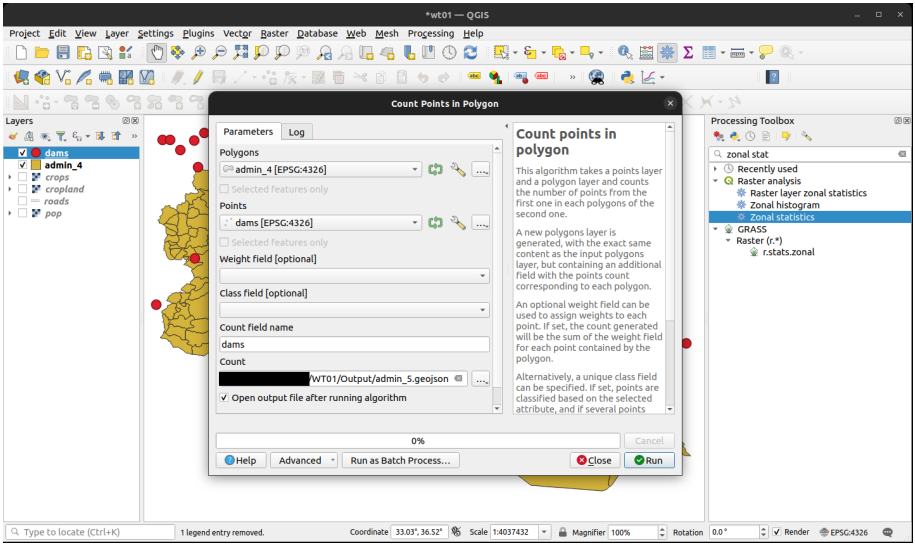
The dam locations should appear on the project window. Rename it from `GRanD_dams_v1_3` to `dams`



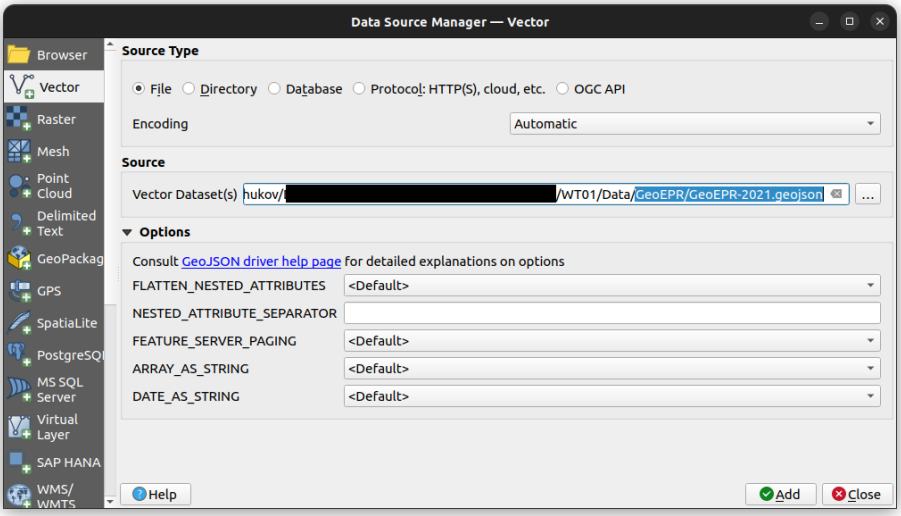
Let's calculate the **number of dams per district**. Open the `Count Points in Polygon` tool (`Vector` → `Analysis Tools` → `Count Points in Polygon`)



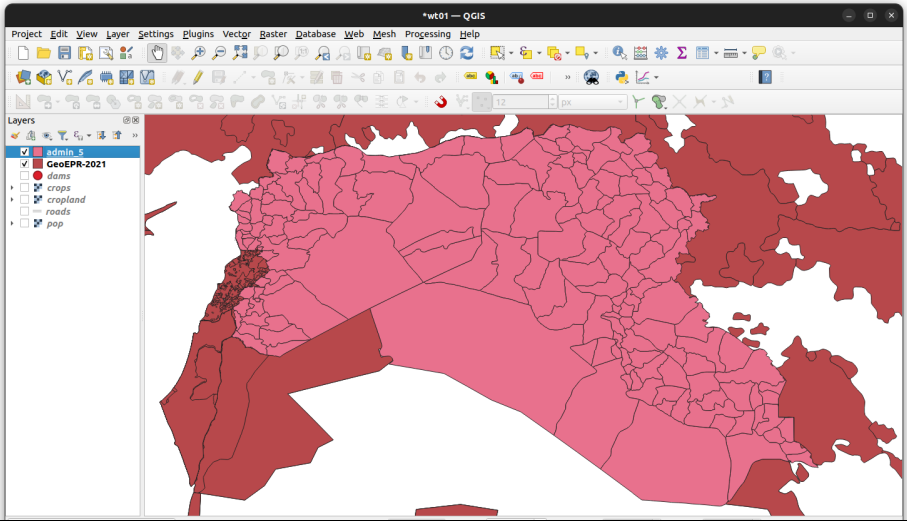
Select `Polygons` = `admin_4`, `Points` = `dams`. Name the count field `dams`, and save the output file as `admin_5.geojson`. Click `Run`



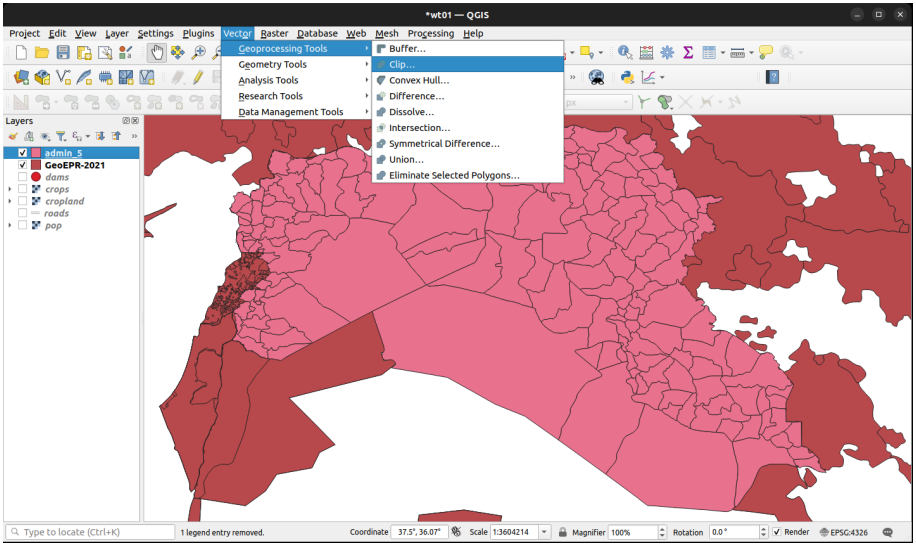
Add the *sectarian divisions* data to the project, using `Add Vector Layer...`\
Load the `GeoEPR-2021.geojson` file in `Data/GeoEPR` folder



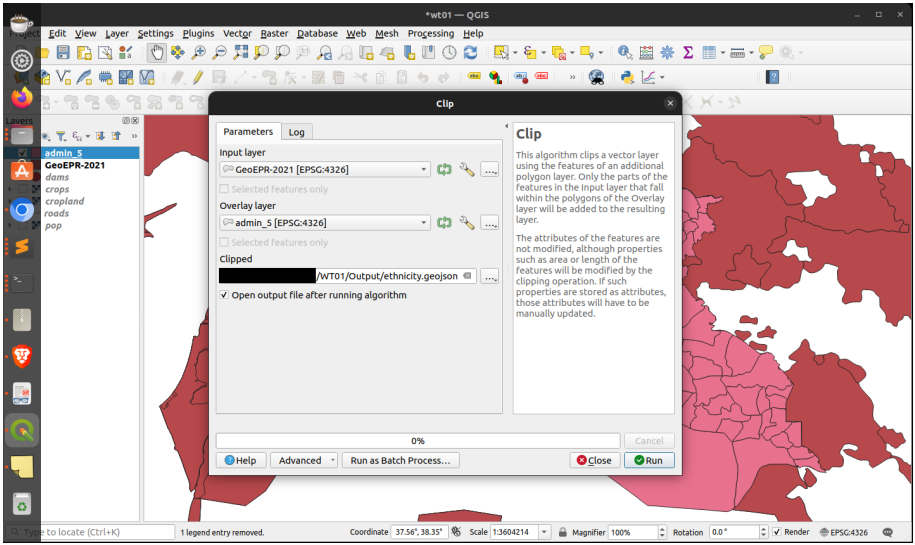
The (global) ethnicity layer should appear in the project window. Let's **calculate the proportion of each district populated by Sunni Arabs**. Let's first extract the subset that overlaps with our study area.



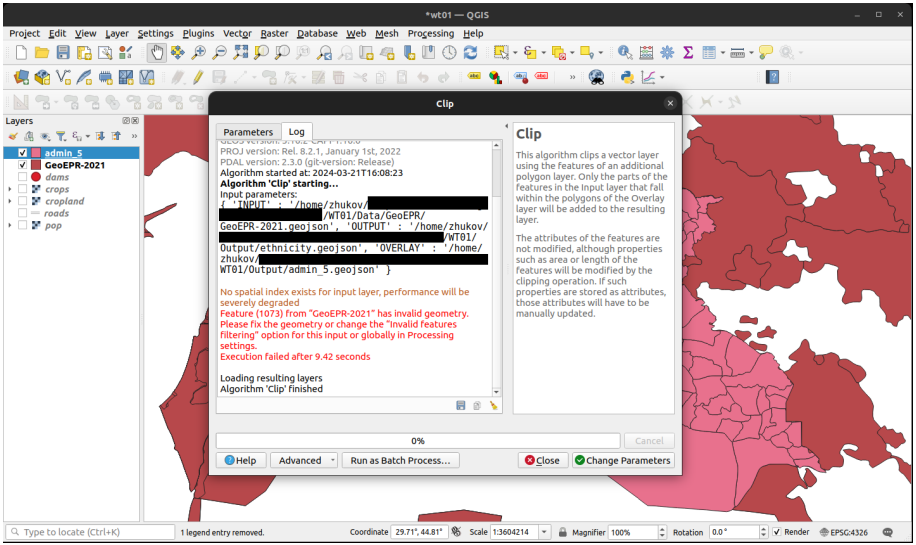
Open the `Clip` (vector) tool (`Vector` → `Geoprocessing Tools` → `Clip`)



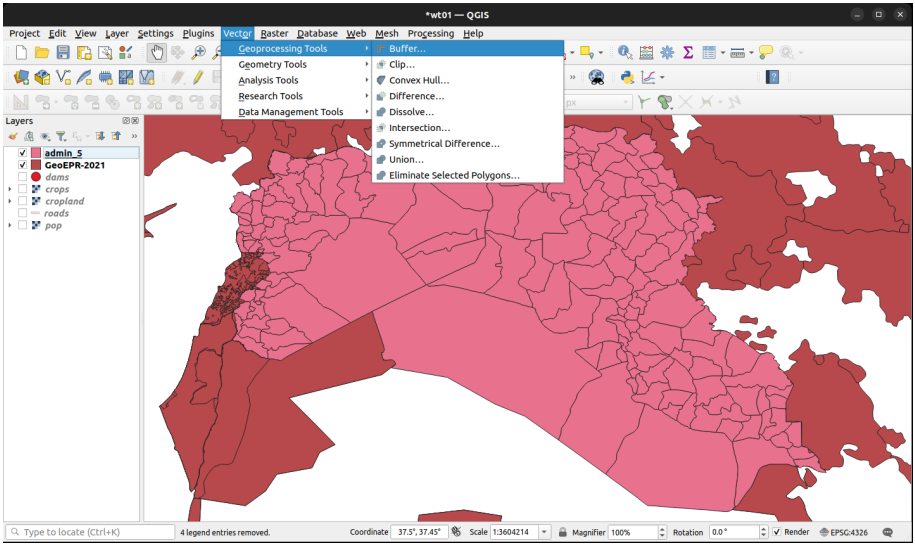
Set `Input layer` = `GeoEPR`, `Overlay layer` = `admin_5`. Save the file as `ethnicity.geojson`. Click `Run`



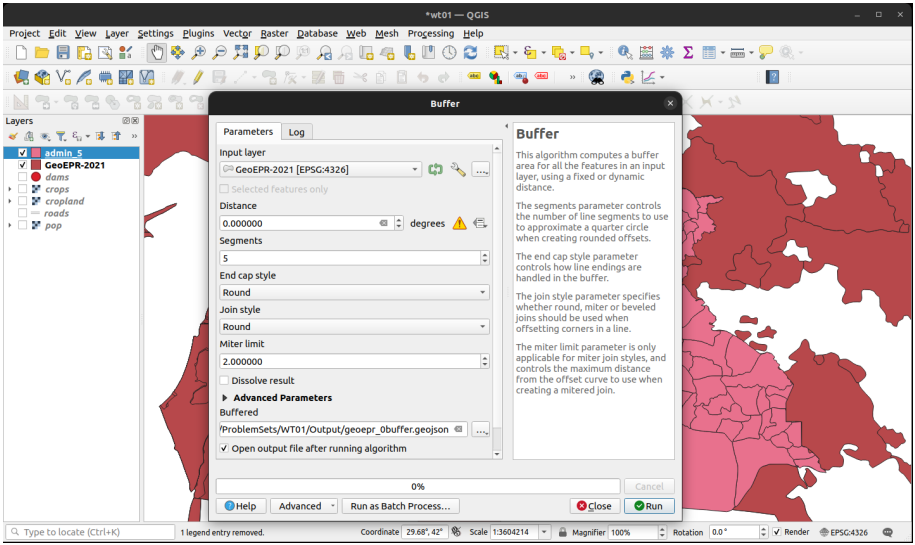
Uh oh! **Execution failed!** The error message says that `GeoEPR-2021` contains "invalid geometries". How do we fix this?



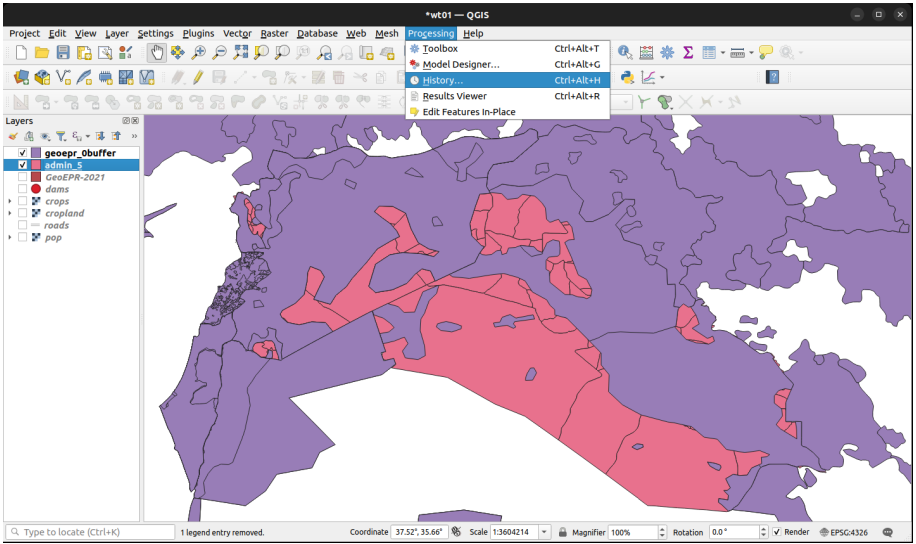
Many "invalid geometry" problems can be solved with the "zero buffer trick". Go to `Vector` → `Geoprocessing Tools` → `Buffer...`



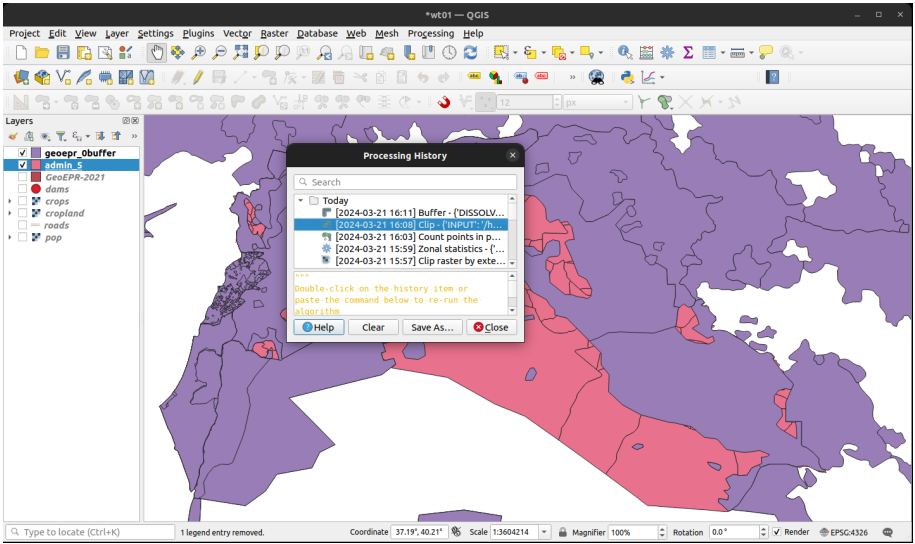
Set `Input layer` = `GeoEPR-2021` and `Distance` = `0`. Keep the defaults for the other parameters and save the file as `geoepr_0buffer.geojson`. Click `Run`



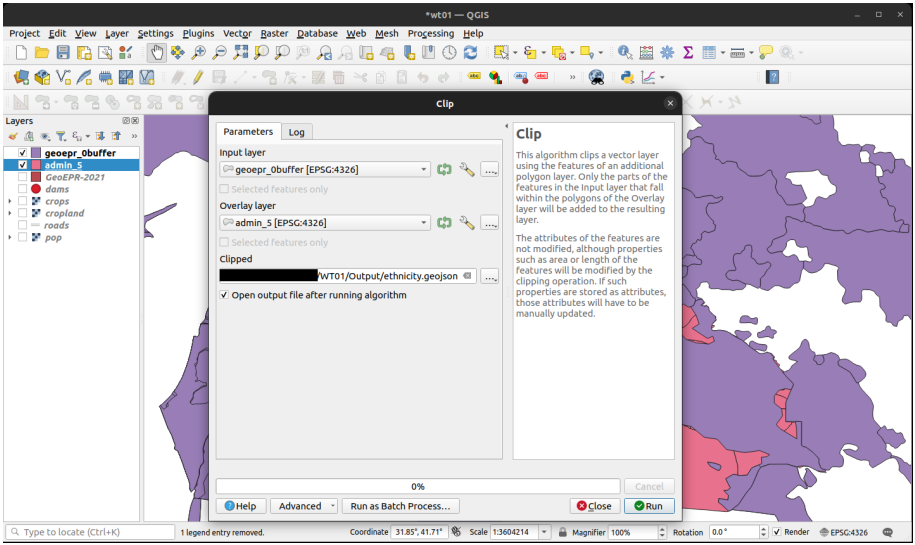
Now let's try the clipping operation again. To save time, go to `Processing` → `History...`



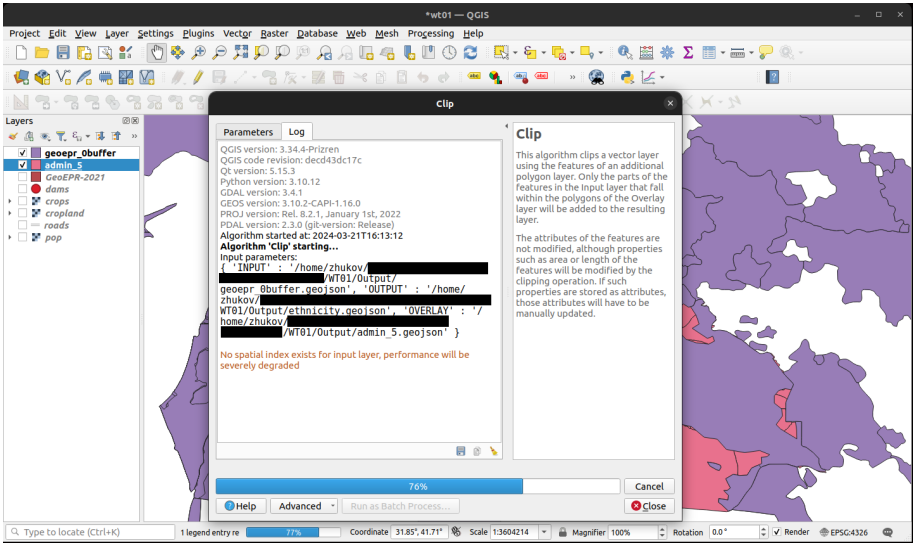
In the `Processing History` window, double-click on `Clip` (it should be the second-most recent item)



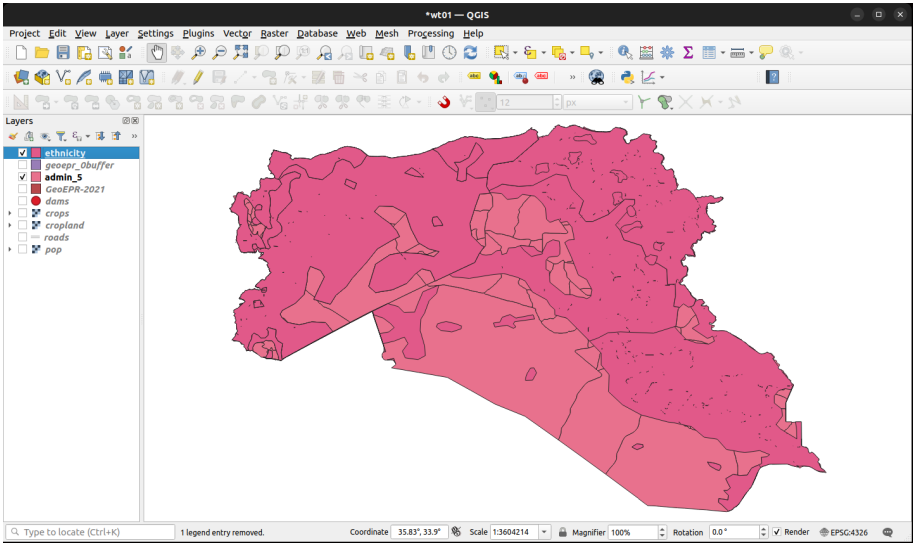
Set the `Input layer` = `geoepr_0buffer`. Click `Run`



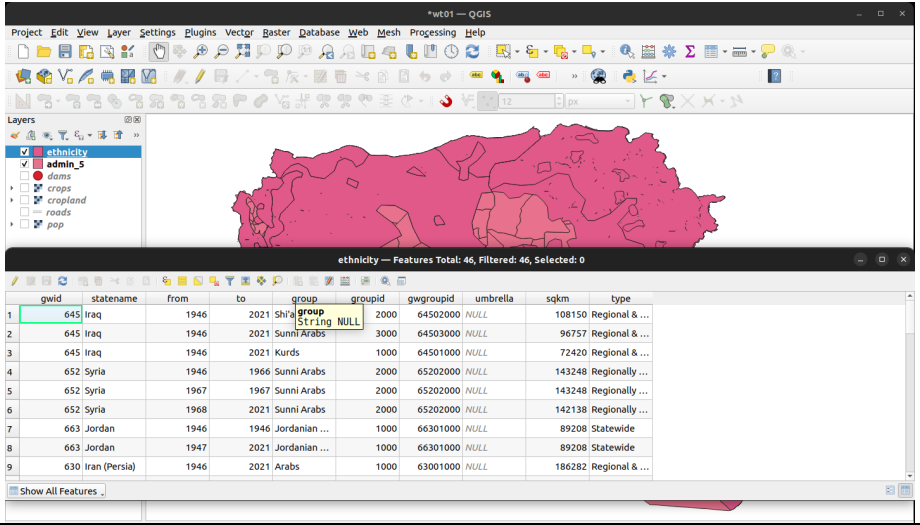
This may take a few minutes, but should finish without errors. . .



The clipped layer should appear in the project window. You can remove the full `GeoEPR` layer to save memory

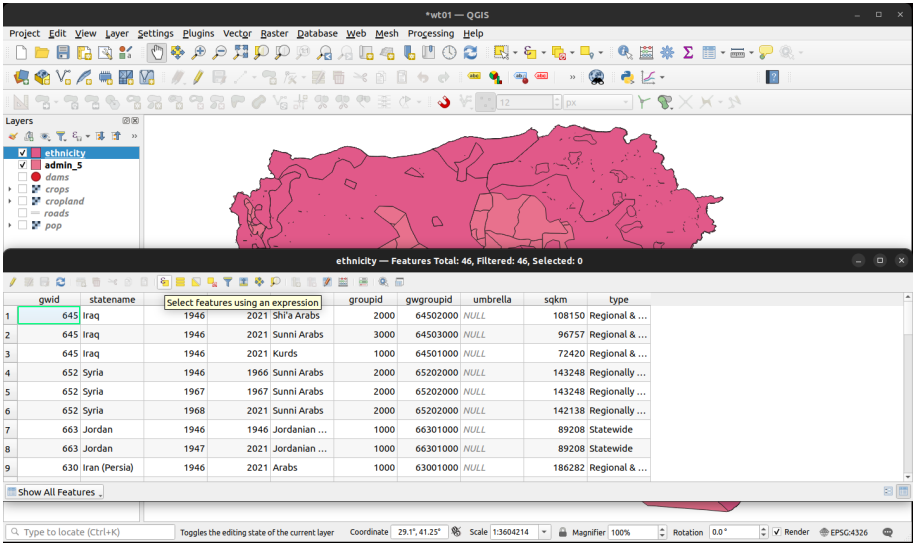


Look at the `group` field in the attribute table for `ethnicity`. There are multiple features for several groups (Kurds, Alawites, etc.). Let's **extract Sunni Arab polygons** only

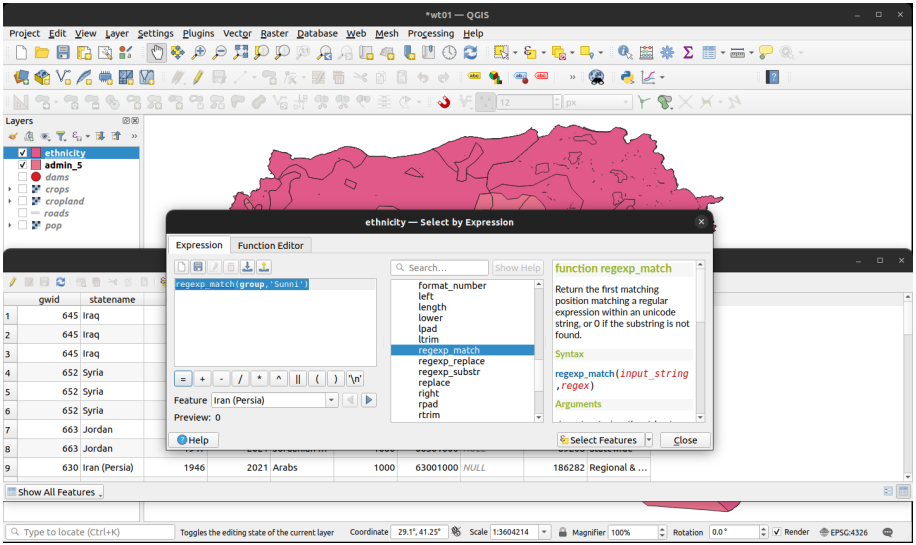


	gwid	statename	from	to	group	groupid	gwgroupid	umbrella	sqkm	type
1	645	Iraq	1946	2021	Shi'a	2000	64502000	NULL	108150	Regional & ...
2	645	Iraq	1946	2021	Sunni Arabs	3000	64503000	NULL	96757	Regional & ...
3	645	Iraq	1946	2021	Kurds	1000	64501000	NULL	72420	Regional & ...
4	652	Syria	1946	1966	Sunni Arabs	2000	65202000	NULL	143248	Regionally ...
5	652	Syria	1967	1967	Sunni Arabs	2000	65202000	NULL	143248	Regionally ...
6	652	Syria	1968	2021	Sunni Arabs	2000	65202000	NULL	142138	Regionally ...
7	663	Jordan	1946	1946	Jordanian ...	1000	66301000	NULL	89208	Statewide
8	663	Jordan	1947	2021	Jordanian ...	1000	66301000	NULL	89208	Statewide
9	630	Iran (Persia)	1946	2021	Arabs	1000	63001000	NULL	186282	Regional & ...

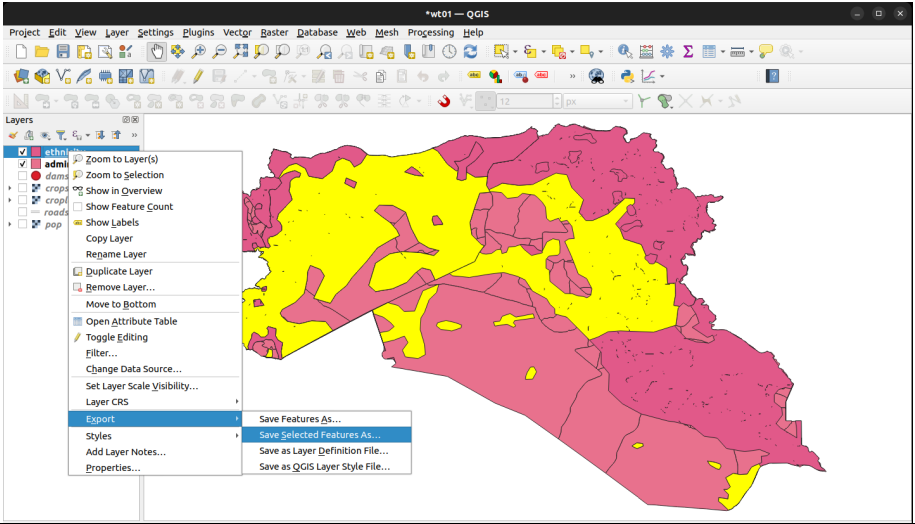
Open the `Select by Expression` tool (`Edit` menu → `Select` → `Select Features by Expression`)



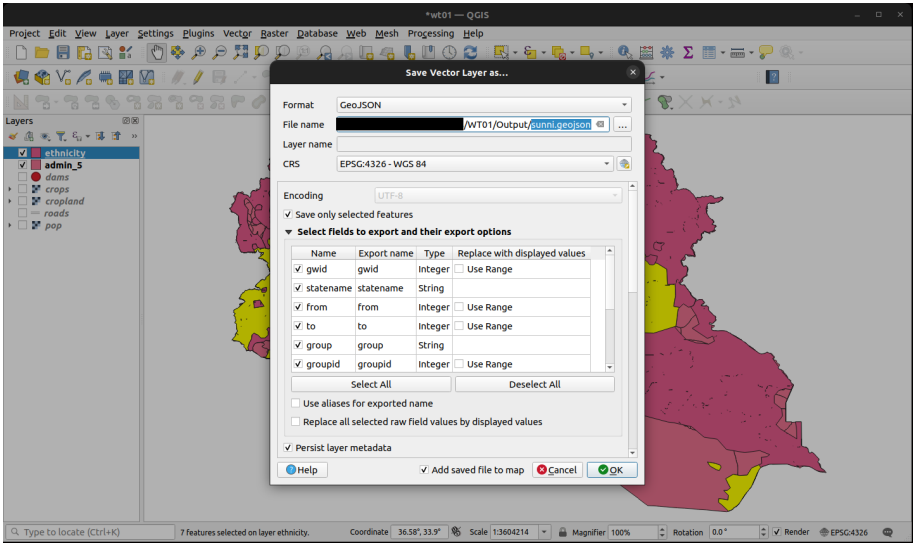
Let's use **regular expressions** to extract the polygons we need. Set `Expression = regexp_match(group,'Sunni')`. Click `Select Features`



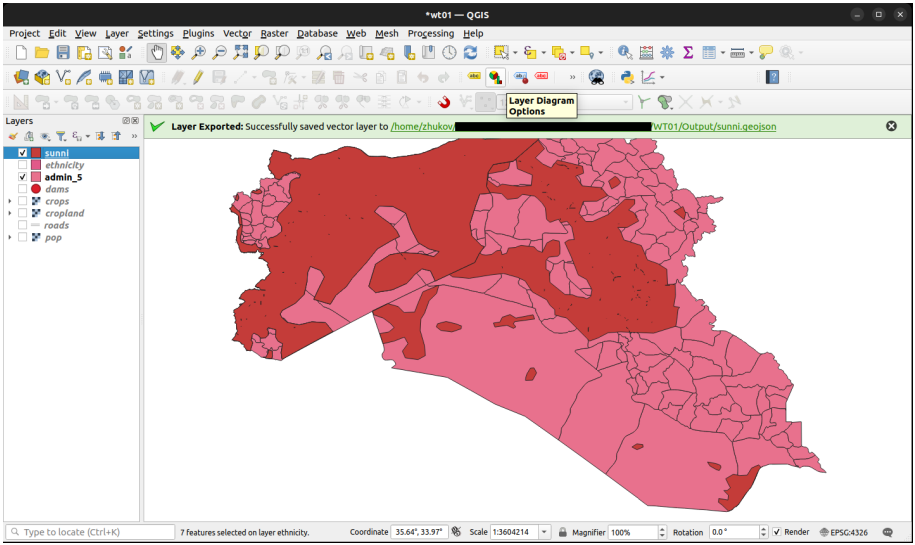
The selected polygons should appear yellow on the map. Let's **extract the selection into a new layer**. Right-click `ethnicity` in the Layer menu, go to `Export`→`Save Selected Features As`



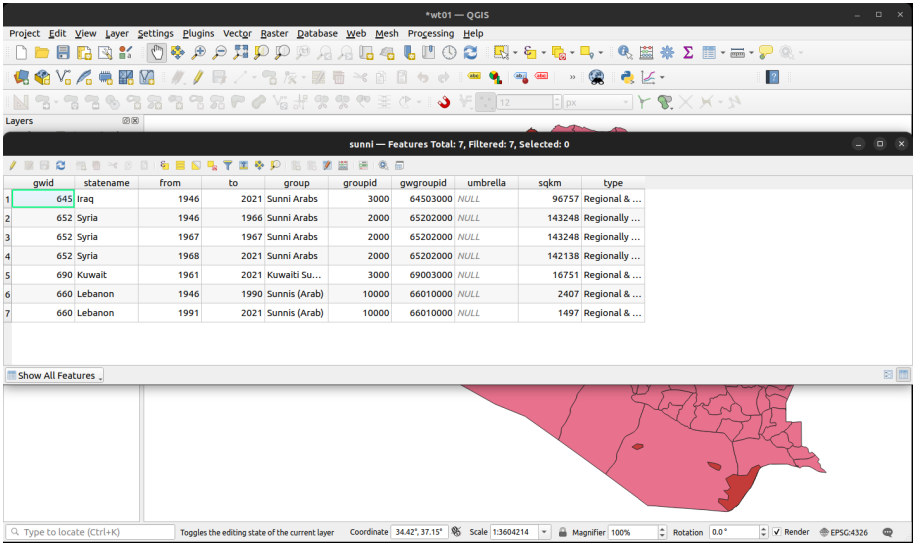
Save the layer as `sunni.geojson`. Make sure the box is checked next to `Save only selected features`. Click `OK`



The extracted selection should appear. Hide the previous layer `ethnicity`

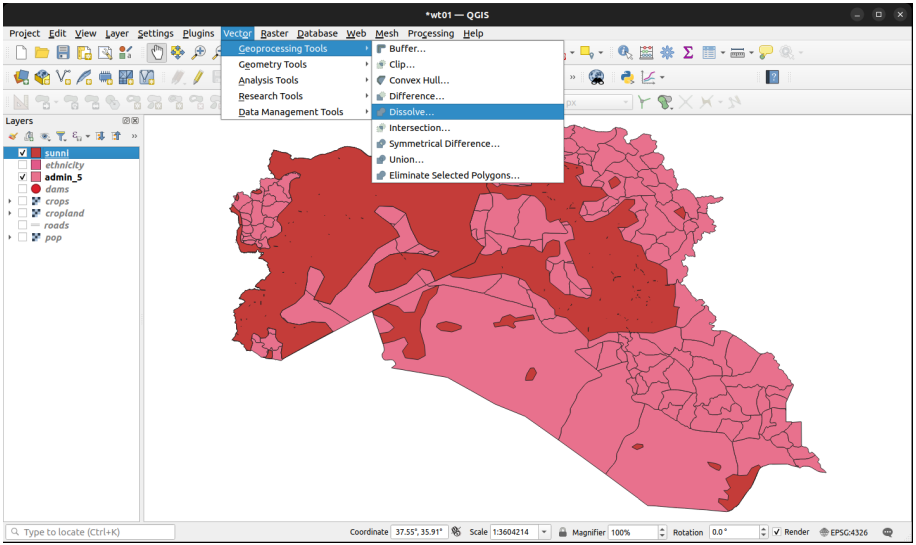


The attribute table for `sunni` should now contain only features where the `group` field contains the word `Sunni`. Let's **consolidate these features into one polygon**

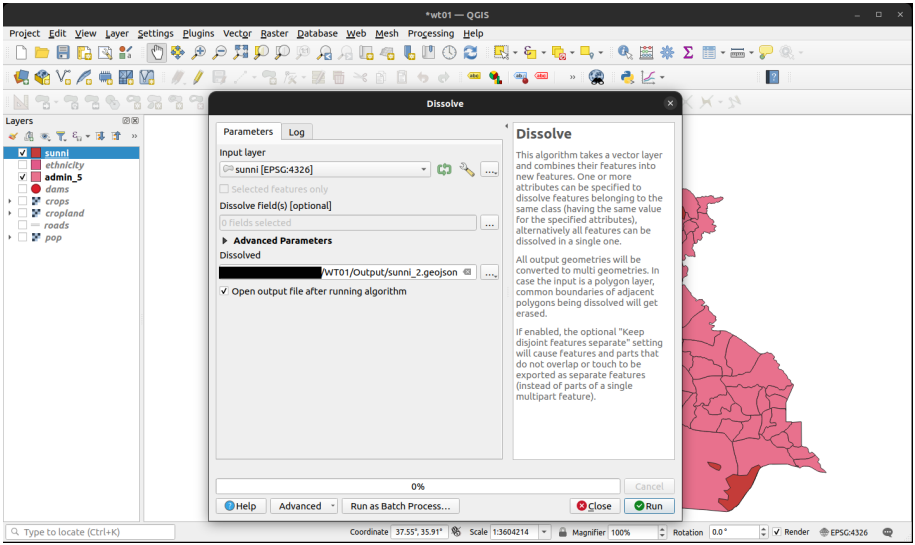


	gwid	statename	from	to	group	groupid	gwgroupid	umbrella	sqkm	type
1	645	Iraq	1946	2021	Sunni Arabs	3000	64503000	NULL	96757	Regionally ...
2	652	Syria	1946	1966	Sunni Arabs	2000	65202000	NULL	143248	Regionally ...
3	652	Syria	1967	1967	Sunni Arabs	2000	65202000	NULL	143248	Regionally ...
4	652	Syria	1968	2021	Sunni Arabs	2000	65202000	NULL	142138	Regionally ...
5	690	Kuwait	1961	2021	Kuwaiti Su...	3000	69003000	NULL	16751	Regional & ...
6	660	Lebanon	1946	1990	Sunnis (Arab)	10000	66010000	NULL	2407	Regional & ...
7	660	Lebanon	1991	2021	Sunnis (Arab)	10000	66010000	NULL	1497	Regional & ...

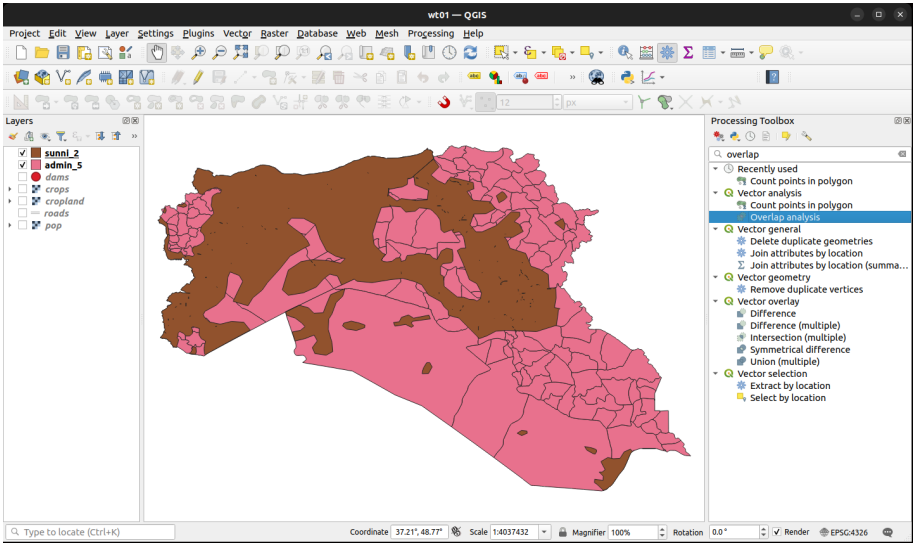
Open the `Dissolve` tool (`Vector` → `Geoprocessing Tools` → `Dissolve`).



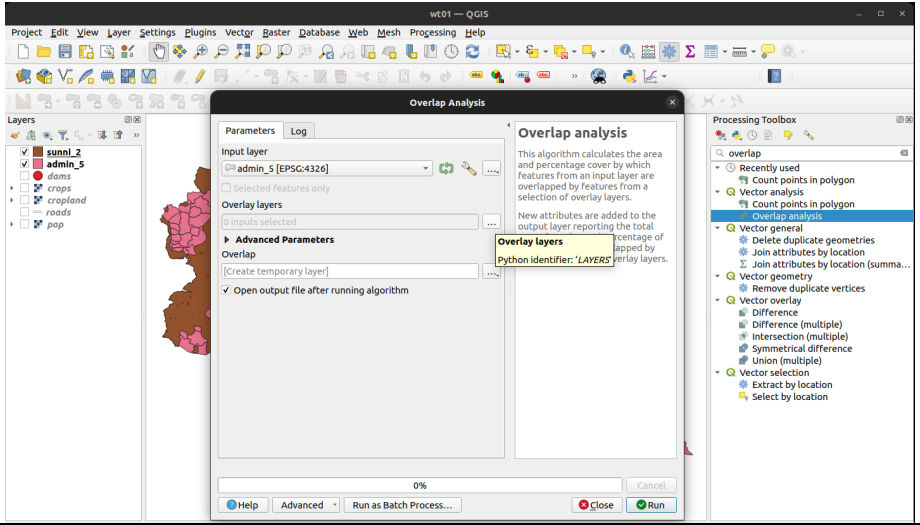
Set `Input layer=sunni`. Save the output as `sunni_2.geojson`



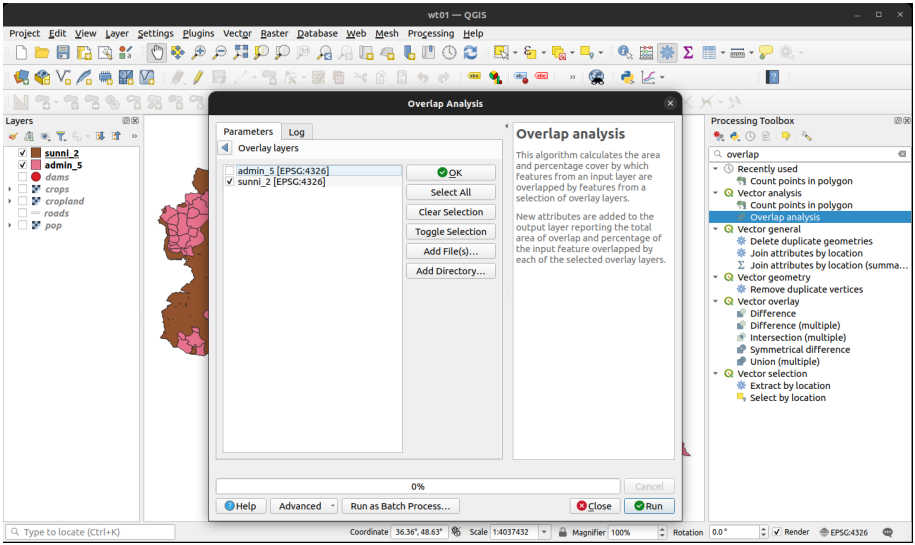
The dissolved polygon should appear in the project window. Now let's **calculate the proportion of each district populated by Sunnis**



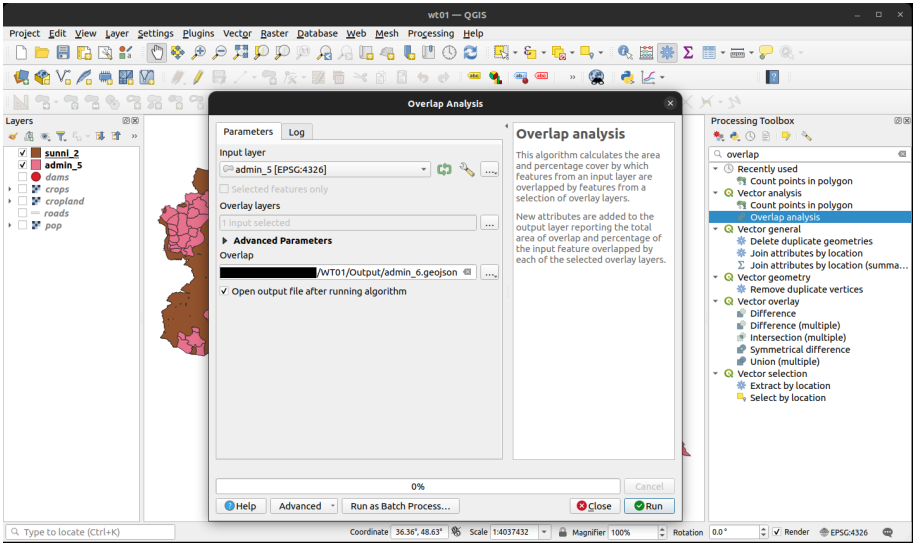
Open the `Overlap Analysis` tool (in `Processing Toolbox` → `Vector Analysis`). Set `Input layer` = `admin_5` and click the `[...]` button next to `Overlay layers`



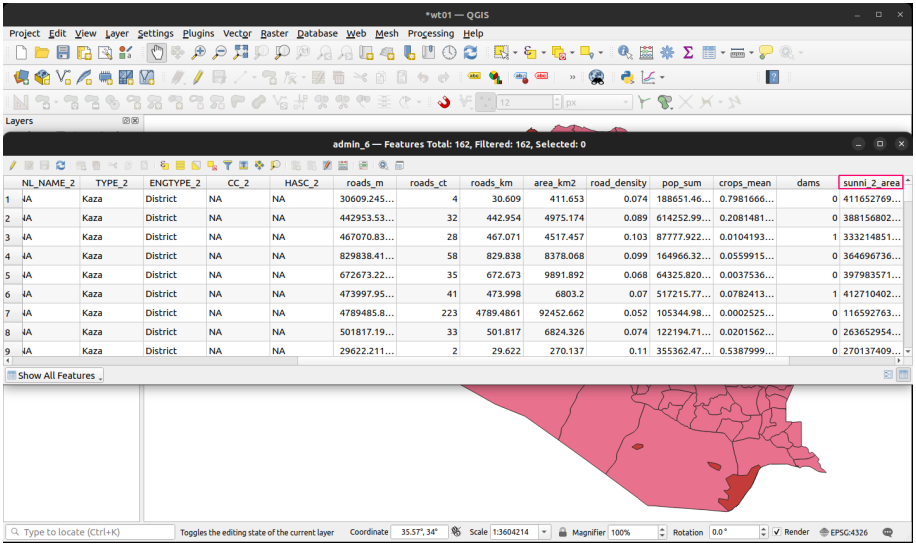
Check the box next to ✓`sunni_2`. Click `OK`



Save the output as `admin_6.geojson` and click `Run`

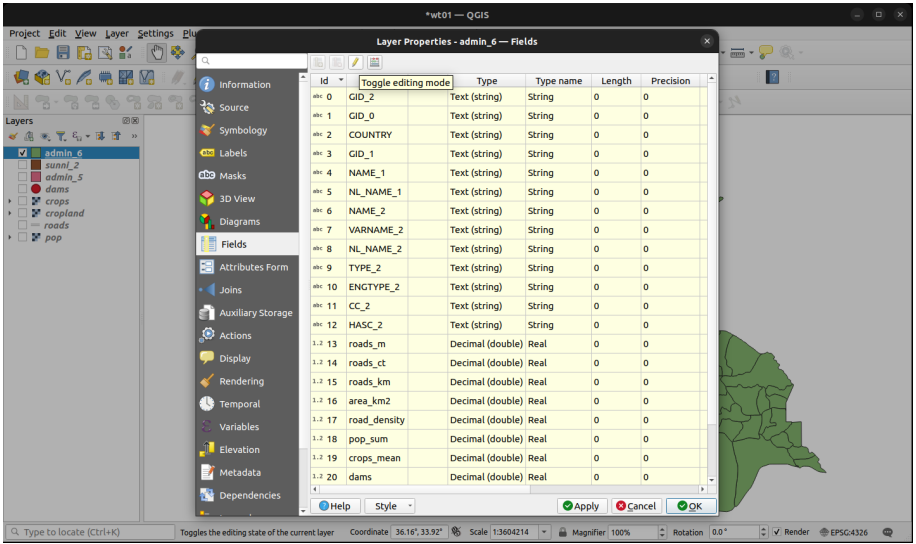


By default, the overlap fields in `admin_6` will be named `sunni_2_area` and `sunni_2_pc`. Let's **change the name of these fields**

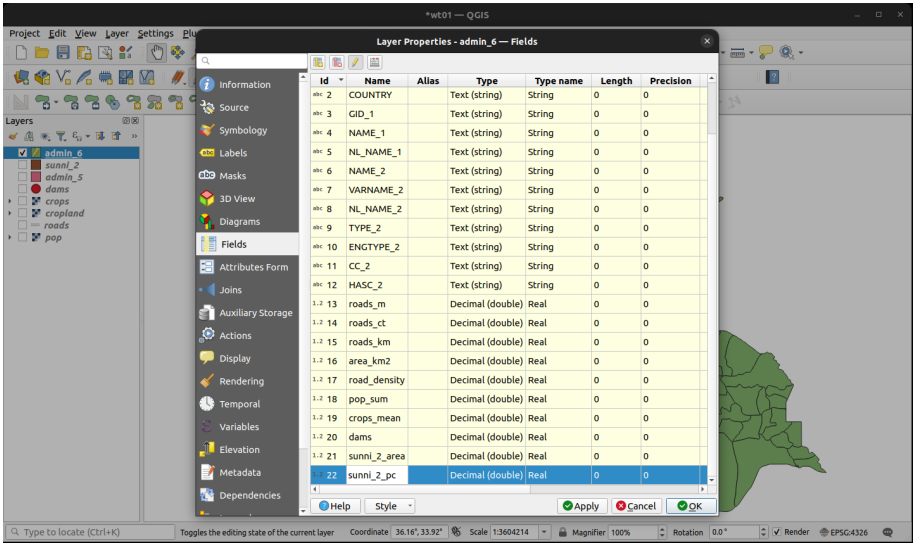


	NL_NAME_2	TYPE_2	ENGTYPE_2	CC_2	HASC_2	roads_m	roads_ct	roads_km	area_km2	road_density	pop_sum	crops_mean	dams	sunni_2_area
1	NA	Kaza	District	NA	NA	30609.245...	4	30.609	411.653	0.074	188651.46...	0.7981666...	0	411652769...
2	NA	Kaza	District	NA	NA	442953.53...	32	442.954	4975.174	0.089	614252.99...	0.2081481...	0	388156802...
3	NA	Kaza	District	NA	NA	467070.83...	28	467.071	4517.457	0.103	87777.922...	0.0104193...	1	333214851...
4	NA	Kaza	District	NA	NA	829838.41...	58	829.838	8378.068	0.099	164966.32...	0.0559915...	0	364696736...
5	NA	Kaza	District	NA	NA	672673.22...	35	672.673	9891.892	0.068	64325.820...	0.0037536...	0	397983571...
6	NA	Kaza	District	NA	NA	473997.95...	41	473.998	6803.2	0.07	517215.77...	0.0782413...	1	412710402...
7	NA	Kaza	District	NA	NA	4789485.8...	223	4789.4861	92452.662	0.052	105344.98...	0.0002525...	0	116592763...
8	NA	Kaza	District	NA	NA	501817.19...	33	501.817	6824.326	0.074	122194.71...	0.0201562...	0	263652954...
9	NA	Kaza	District	NA	NA	29622.211...	2	29.622	270.137	0.11	355362.47...	0.5387999...	0	270137409...

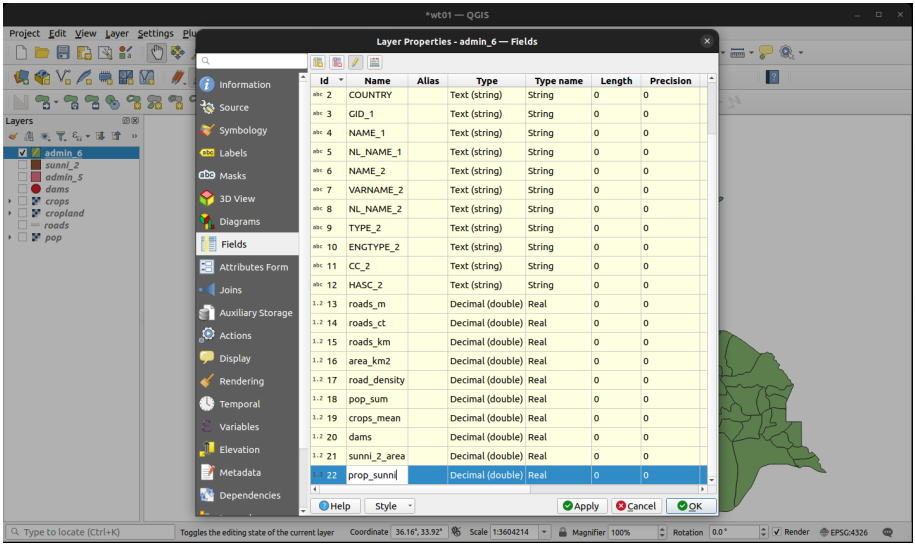
Open the layer `Properties` for `admin_6`, go to the `Fields` tab and click on the "pencil" button (`Toggle editing mode`)



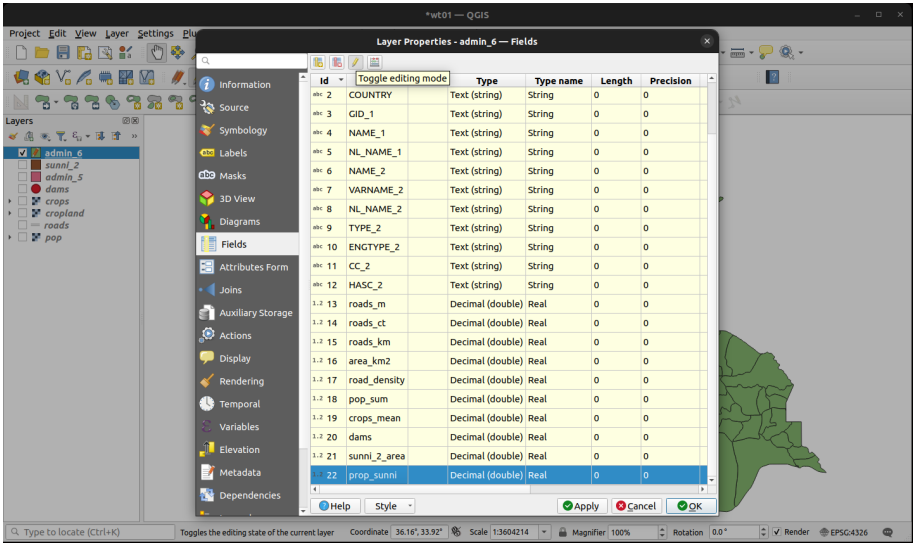
Scroll down to `sunni_2_pc` and double-click on its name



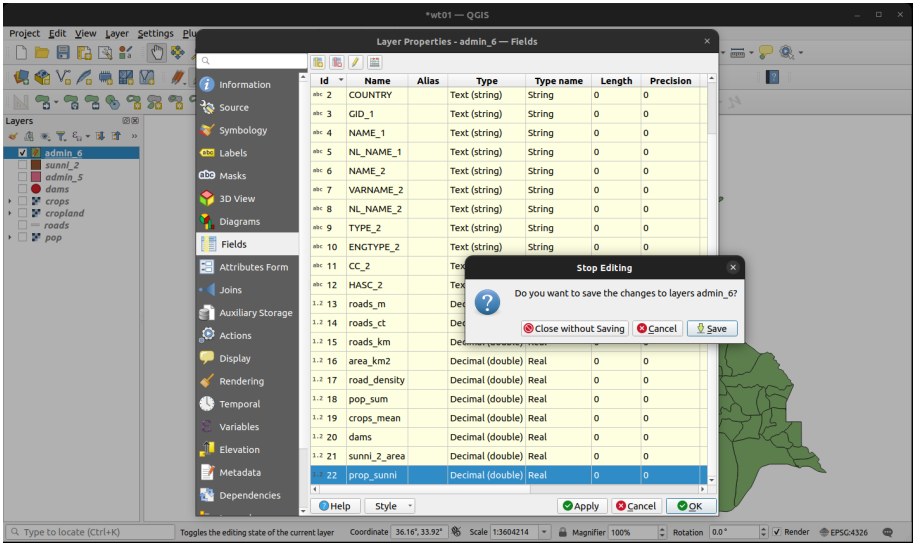
Rename the field `prop_sunni`



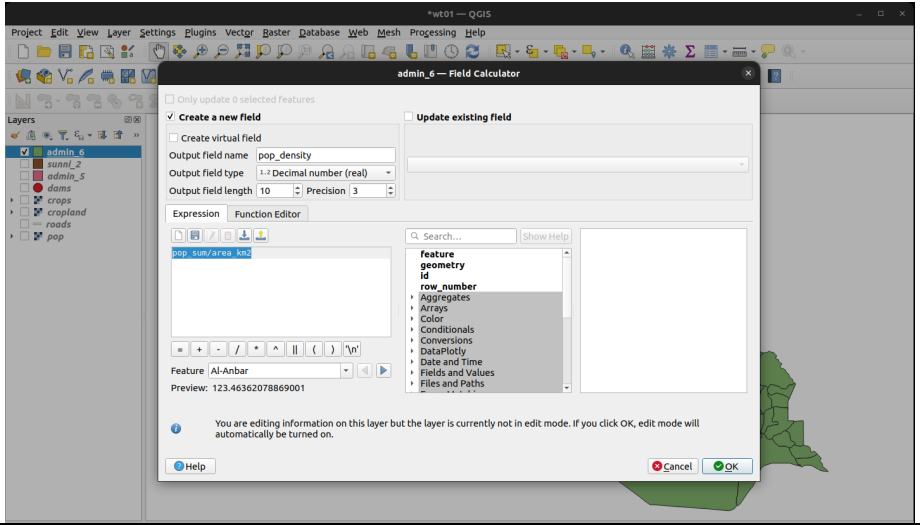
Hit Enter key (or Return key) to commit the name change. Click on the "pencil" again to leave editing mode



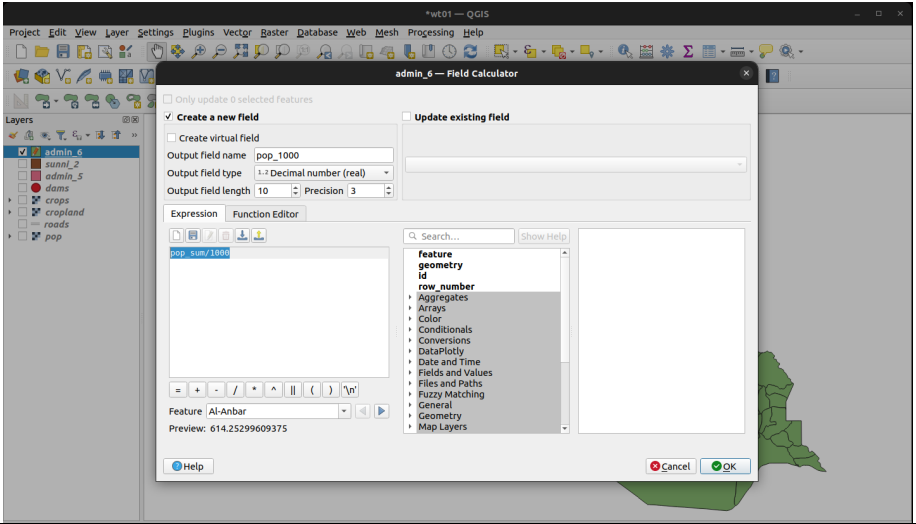
When prompted, save the changes you just made to `admin_6`



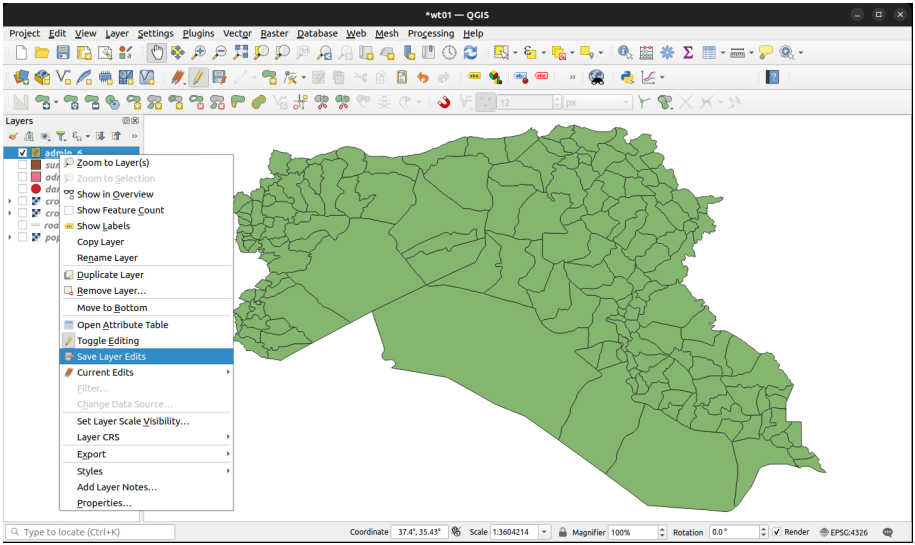
While we're at it, let's create a couple additional fields that could come in handy in the analysis. Create a **population density** field, named `pop_density`, of type `Decimal number (real)`, with `Expression` set to `pop_sum/area_km2`



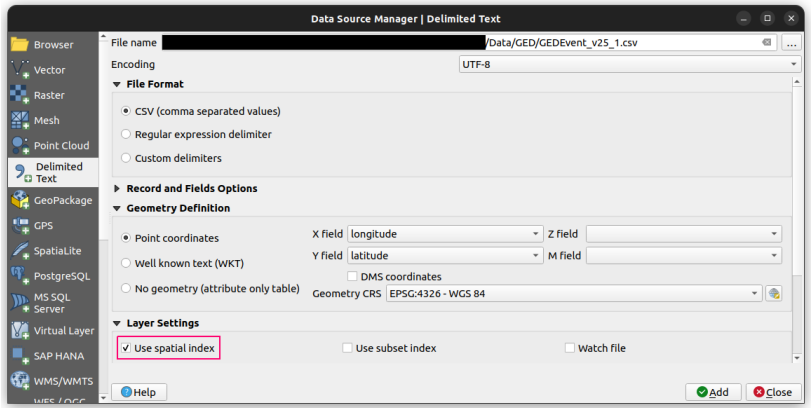
Let's also create a **rescaled population** field (1000's of residents). Name it `pop_1000`, with type `Decimal number (real)`. Set `Expression` to `pop_sum/1000`



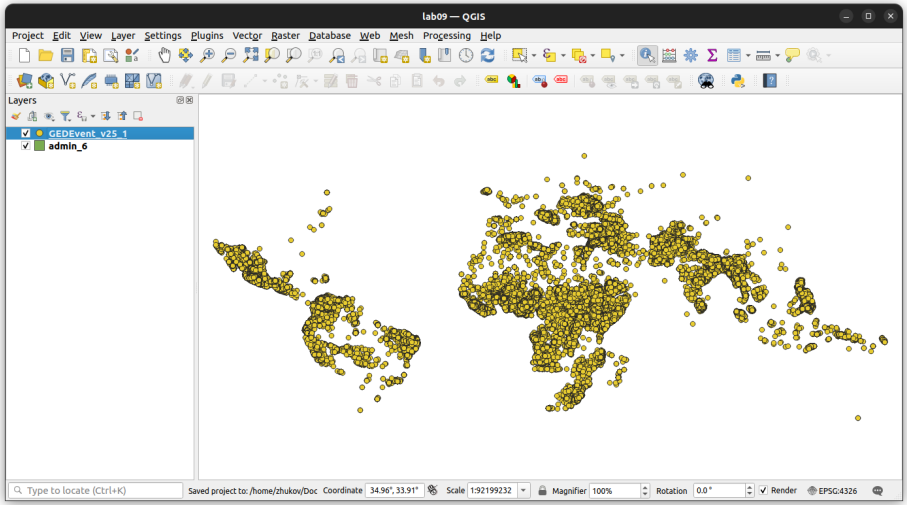
Save the layer edits to `admin_6`!



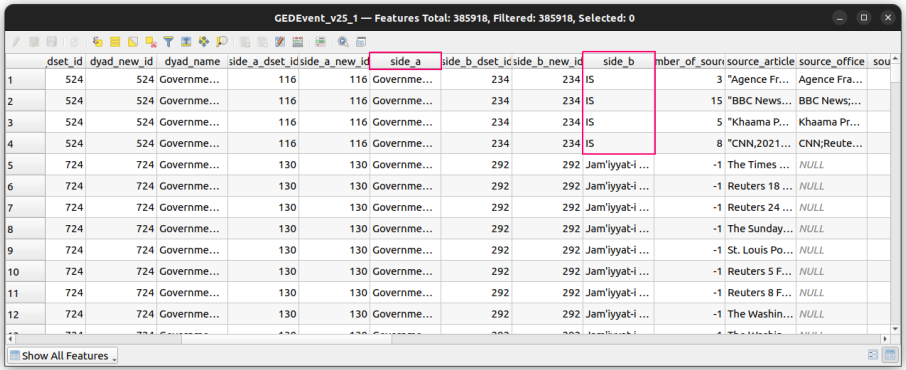
Add the *Islamic State violence data* to the project, using `Add Delimited Text Layer...`. Load the `GEDEvent_v25_1.csv` file in `Data/GED` folder. Set `X field = longitude` and `Y field = latitude`. Check the box next to ✓`Use spatial index`



The (global) GED violent events layer should appear. There are several hundred thousand points here. We need to **extract events in Syria and Iraq involving the Islamic State**

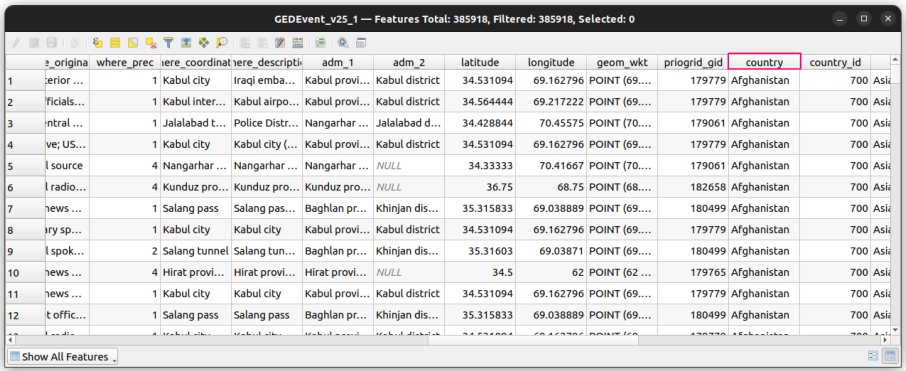


To figure out how to extract this data subset (by actor and location), let's explore the attribute table. We see fields for `side_a` and `side_b`, which list actors



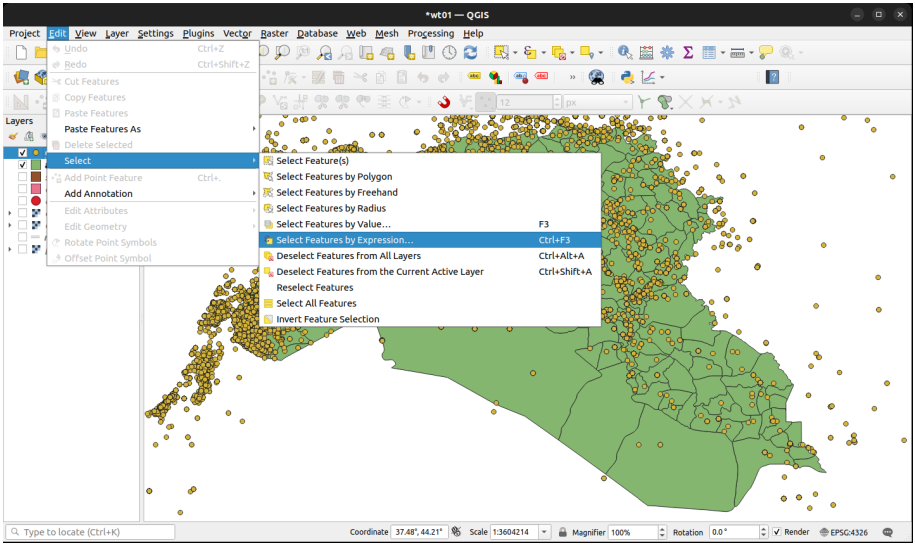
	dset_id	dyad_new_id	dyad_name	side_a_dset_id	side_a_new_id	side_a	side_b_dset_id	side_b_new_id	side_b	mber_of_sour	source_article	source_office	sou
1	524	524	Governme…	116	116	Governme…	234	234	IS	3	"Agence Fr…	Agence Fra…	
2	524	524	Governme…	116	116	Governme…	234	234	IS	15	"BBC News…	BBC News;…	
3	524	524	Governme…	116	116	Governme…	234	234	IS	5	"Khaama P…	Khaama Pr…	
4	524	524	Governme…	116	116	Governme…	234	234	IS	8	"CNN,2021…	CNN;Reute…	
5	724	724	Governme…	130	130	Governme…	292	292	Jam'iyyat-i …	-1	The Times …	NULL	
6	724	724	Governme…	130	130	Governme…	292	292	Jam'iyyat-i …	-1	Reuters 18 …	NULL	
7	724	724	Governme…	130	130	Governme…	292	292	Jam'iyyat-i …	-1	Reuters 24 …	NULL	
8	724	724	Governme…	130	130	Governme…	292	292	Jam'iyyat-i …	-1	The Sunday…	NULL	
9	724	724	Governme…	130	130	Governme…	292	292	Jam'iyyat-i …	-1	St. Louis Po…	NULL	
10	724	724	Governme…	130	130	Governme…	292	292	Jam'iyyat-i …	-1	Reuters 5 F…	NULL	
11	724	724	Governme…	130	130	Governme…	292	292	Jam'iyyat-i …	-1	Reuters 8 F…	NULL	
12	724	724	Governme…	130	130	Governme…	292	292	Jam'iyyat-i …	-1	The Washin…	NULL	

There is also a `country` field. So, we need to select points where `country` is Iraq or Syria and the Islamic State is `side_a` (or `side_b`)



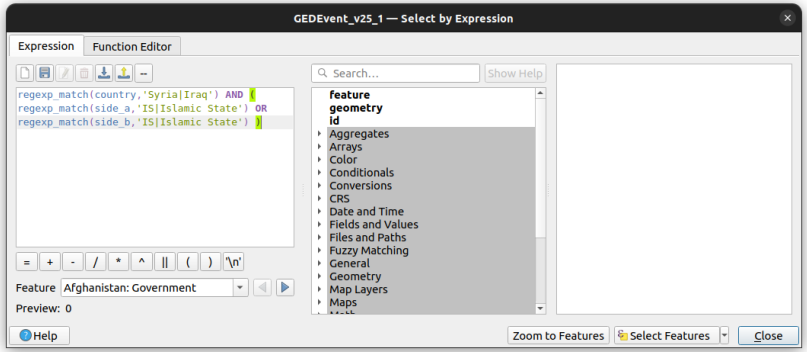
	e_origina	where_prec	ere_coordinat	here_descripti	adm_1	adm_2	latitude	longitude	geom_wkt	priogrid_gid	country	country_id	
1	terior …	1	Kabul city	Iraqi emba…	Kabul provi…	Kabul district	34.531094	69.162796	POINT (69.…	179779	Afghanistan	700	Asia
2	fficials…	1	Kabul inter…	Kabul airpo…	Kabul provi…	Kabul district	34.564444	69.217222	POINT (69.…	179779	Afghanistan	700	Asia
3	ntral …	1	Jalalabad t…	Police Distr…	Nangarhar …	Jalalabad d…	34.428844	70.45575	POINT (70.…	179061	Afghanistan	700	Asia
4	ve; US…	1	Kabul city	Kabul city (…	Kabul provi…	Kabul district	34.531094	69.162796	POINT (69.…	179779	Afghanistan	700	Asia
5	l source	4	Nangarhar …	Nangarhar …	Nangarhar …	*NULL*	34.33333	70.41667	POINT (70.…	179061	Afghanistan	700	Asia
6	l radio…	4	Kunduz pro…	Kunduz pro…	Kunduz pro…	*NULL*	36.75	68.75	POINT (68.…	182658	Afghanistan	700	Asia
7	news …	1	Salang pass	Salang pas…	Baghlan pr…	Khinjan dis…	35.315833	69.038889	POINT (69.…	180499	Afghanistan	700	Asia
8	ary sp…	1	Kabul city	Kabul city	Kabul provi…	Kabul district	34.531094	69.162796	POINT (69.…	179779	Afghanistan	700	Asia
9	l spok…	2	Salang tunnel	Salang tun…	Baghlan pr…	Khinjan dis…	35.31603	69.03871	POINT (69.…	180499	Afghanistan	700	Asia
10	news …	4	Hirat provi…	Hirat provi…	Hirat provi…	*NULL*	34.5	62	POINT (62 …	179765	Afghanistan	700	Asia
11	news …	1	Kabul city	Kabul city	Kabul provi…	Kabul district	34.531094	69.162796	POINT (69.…	179779	Afghanistan	700	Asia
12	t offic…	1	Salang pass	Salang pass	Baghlan pr…	Khinjan dis…	35.315833	69.038889	POINT (69.…	180499	Afghanistan	700	Asia

Highlight the GED layer and go to `Edit` → `Select` → `Select by Expression...`



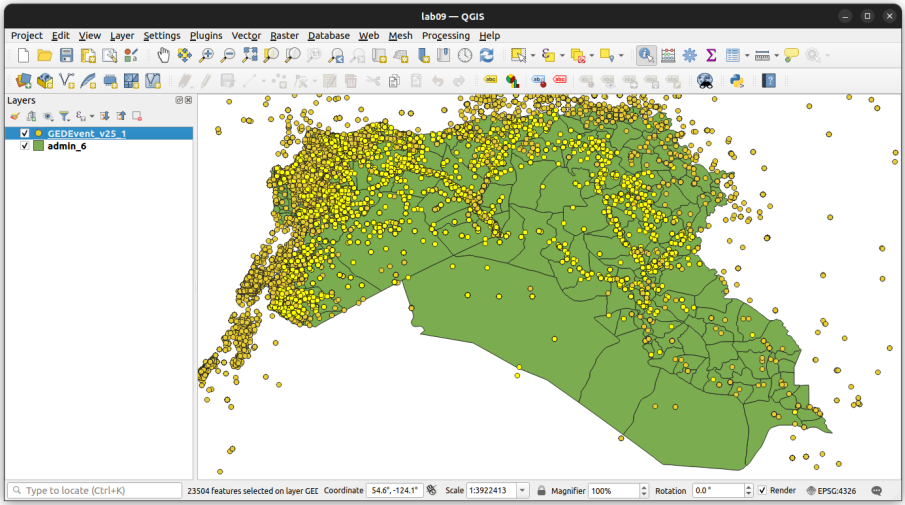
Let's combine regular expressions with logical operators. Set `Expression` to

```
regexp_match(country,'Syria|Iraq') AND (
regexp_match(side_a,'IS|Islamic State') OR
regexp_match(side_b,'IS|Islamic State') )
```

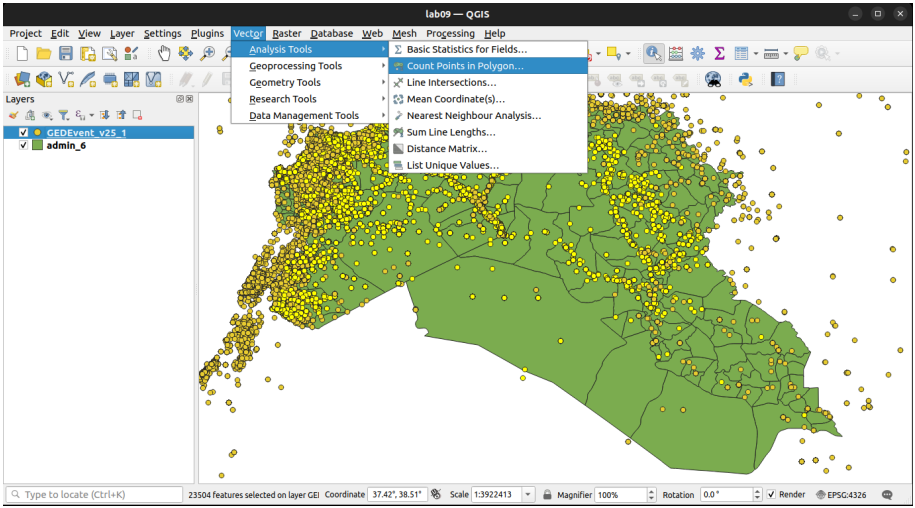


The vertical slash | is a regular expression for "OR". So, in English this expression means 'features where the field `country` contains "Iraq or Syria" and either `side_a` or `side_b` contains "IS or Islamic State" '

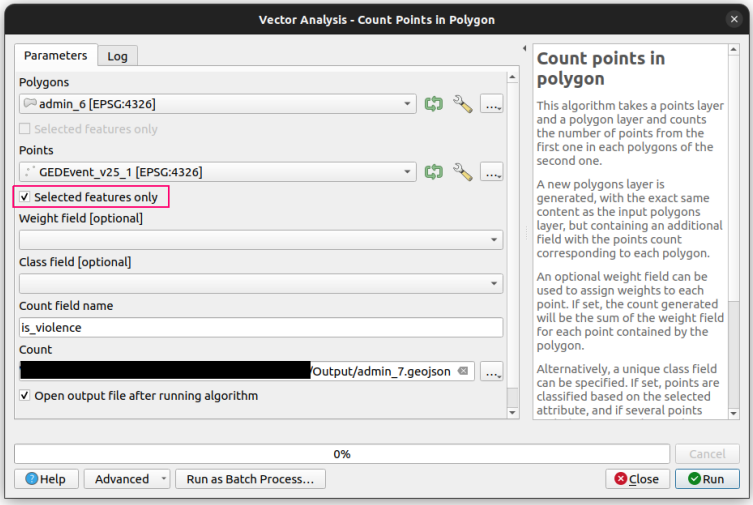
This should select a little over 23,000 events. Now let's calculate the **number of ISIS attacks per district**



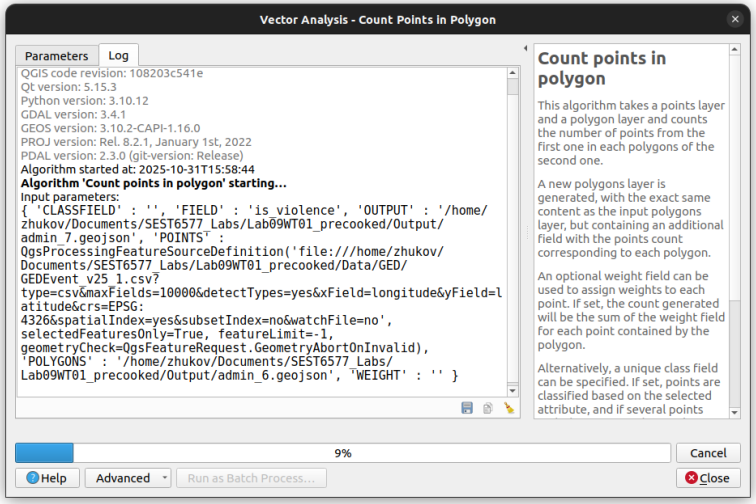
Open the `Count Points in Polygon` tool



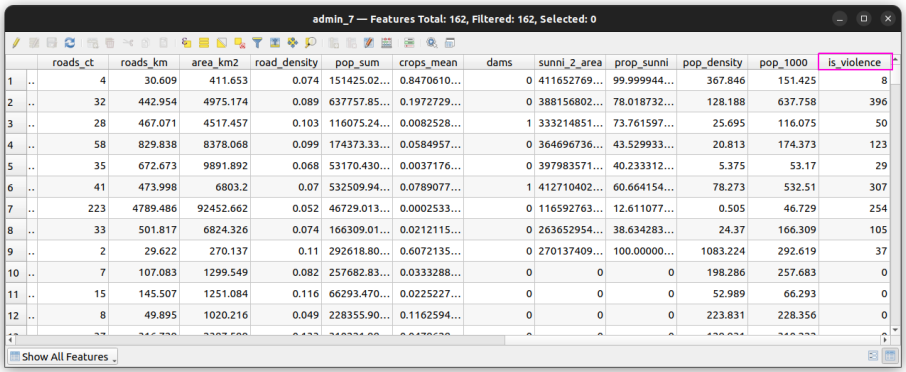
Select `Polygons` = `admin_6`, `Points` = `GEDEvent_v25_1`. Make sure the box is checked next to `Selected Features Only` for the points. Name the count field `is_violence`, and save the output file as `admin_7.geojson`. Click `Run`



This may take a minute or two to run

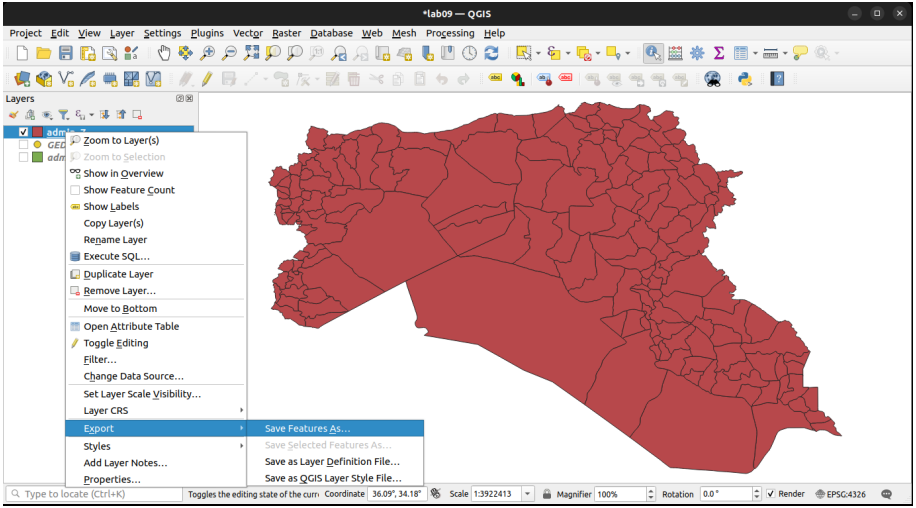


The attribute table for this new layer should include all the new fields we have generated. Let's now **export the table as CSV** for further analysis

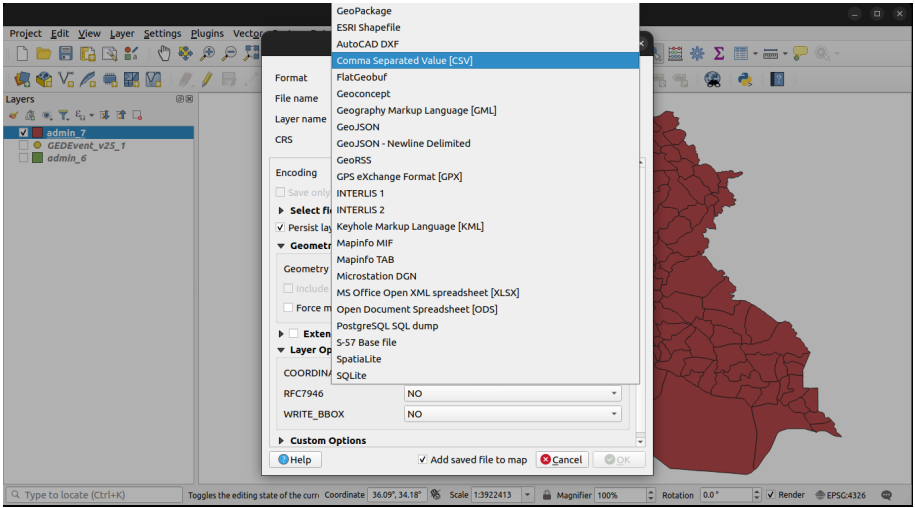


		roads_ct	roads_km	area_km2	road_density	pop_sum	crops_mean	dams	sunni_2_area	prop_sunni	pop_density	pop_1000	is_violence
1	..	4	30.609	411.653	0.074	151425.02...	0.8470610...	0	411652769...	99.999944...	367.846	151.425	8
2	..	32	442.954	4975.174	0.089	637757.85...	0.1972729...	0	388156802...	78.018732...	128.188	637.758	396
3	..	28	467.071	4517.457	0.103	116075.24...	0.0082528...	1	333214851...	73.761597...	25.695	116.075	50
4	..	58	829.838	8378.068	0.099	174373.33...	0.0584957...	0	364696736...	43.529933...	20.813	174.373	123
5	..	35	672.673	9891.892	0.068	53170.430...	0.0037176...	0	397983571...	40.233312...	5.375	53.17	29
6	..	41	473.998	6803.2	0.07	532509.94...	0.0789077...	1	412710402...	60.664154...	78.273	532.51	307
7	..	223	4789.486	92452.662	0.052	46729.013...	0.0002533...	0	116592763...	12.611077...	0.505	46.729	254
8	..	33	501.817	6824.326	0.074	166309.01...	0.0212115...	0	263652954...	38.634283...	24.37	166.309	105
9	..	2	29.622	270.137	0.11	292618.80...	0.6072135...	0	270137409...	100.00000...	1083.224	292.619	37
10	..	7	107.083	1299.549	0.082	257682.83...	0.0333288...	0	0	0	198.286	257.683	0
11	..	15	145.507	1251.084	0.116	66293.470...	0.0225227...	0	0	0	52.989	66.293	0
12	..	8	49.895	1020.216	0.049	228355.90...	0.1162594...	0	0	0	223.831	228.356	0

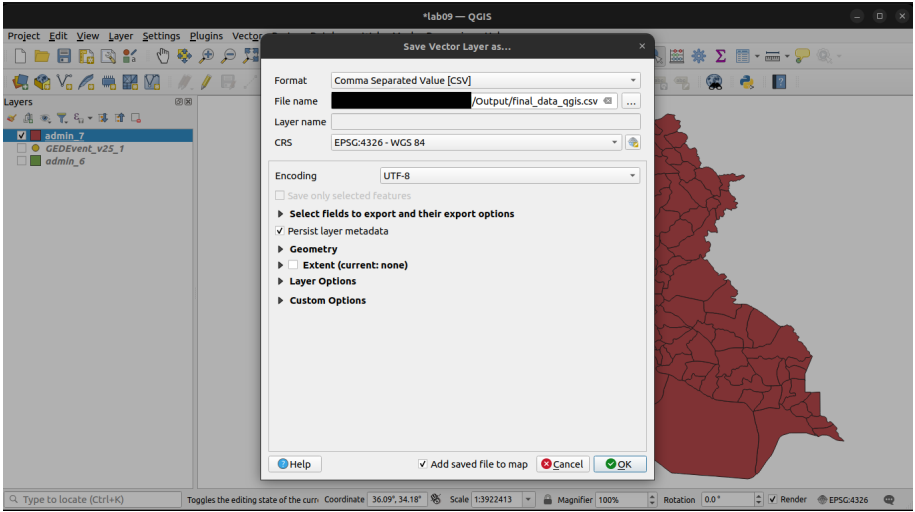
Right-click `admin_7` in the Layer menu, go to `Export`→`Save Features As`.



Save the layer as an `Comma Separated Values` file



Name the file `final_data_qgis.csv`. Click `OK`



Analysis

Loading R packages

To conduct a regression analysis of these data in R, we will be using the `stargazer` package

```
library(stargazer)
```

NOTE: The replication code for all of the preceding steps R is in `wt01_demo.R` on RStudio Cloud, and in `Lab09WT01.zip` (posted on Canvas).

Regression models

Now we're finally able to proceed to the *analysis stage*. For this we will need to **open the CSV file** we just created in R.

This code chunk imports the `final_data_qgis.csv` file into an object called `X`, and then lists the variable names:

```
X = read.csv("Output/final_data_qgis.csv")
names(X)
```

```
##  [1] "GID_2"         "GID_0"         "COUNTRY"       "GID_1"         "NAME_1"
##  [6] "NL_NAME_1"     "NAME_2"        "VARNAME_2"     "NL_NAME_2"     "TYPE_2"
## [11] "ENGTYPE_2"     "CC_2"          "HASC_2"        "layer"         "path"
## [16] "roads_m"       "roads_ct"      "roads_km"      "area_km2"      "road_density"
## [21] "pop_sum"       "crops_mean"    "dams"          "sunni_2_area" "prop_sunni"
## [26] "pop_density"   "pop_1000"      "is_violence"
```

All the variables we created seem to be here.
Let's now *run some regression models!*

Quick refresher: our regression analysis will test 5 hypotheses at once

$$\begin{aligned}\text{violence}_i = &\beta_1 \text{road density}_i + \beta_2 \text{population}_i + \beta_3 \text{cropland}_i \\ &+ \beta_4 \text{dams}_i + \beta_5 \text{Sunni presence}_i + \epsilon_i\end{aligned}$$

where

- violence_i is the observed number of ISIS attacks in district i (`is_violence`)
- road density_i, ..., Sunni presence_i are explanatory variables (`road_density`, `pop_1000`, `crops_mean`, `dams`, `prop_sunni`)
- β are coefficient estimates corresponding to each Hypothesis

Hypothesis	Expectation	Observation
1. Power projection	$\beta_1 < 0$	?
2. Demographics	$\beta_2 > 0$	?
3. Political economy	$\beta_3 < 0$	?
4. Key infrastructure	$\beta_4 > 0$	?
5. Sectarian divisions	$\beta_5 > 0$	?

The first model (`mod1`) is an **Ordinary Least Squares** model that regresses a logarithmically-transformed dependent variable `log(is_violence + 1)` on all of the explanatory variables that correspond to our hypotheses. The log-transform is useful here because the `is_violence` variable is highly skewed

```
mod1 = lm(log(is_violence+1) ~ road_density + pop_1000 + crops_mean +
    dams + prop_sunni, data=X)
summary(mod1)
```

```
##
## Call:
## lm(formula = log(is_violence + 1) ~ road_density + pop_1000 +
##     crops_mean + dams + prop_sunni, data = X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5068 -1.0639 -0.1906  0.9280  3.8392
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.4918950  0.3455419   4.318 2.79e-05 ***
## road_density -1.4859463  3.1746426  -0.468 0.640390
## pop_1000      0.0009083  0.0002421   3.752 0.000247 ***
## crops_mean   -1.4268796  0.4836122  -2.950 0.003662 **
## dams          1.1105982  0.5346231   2.077 0.039408 *
## prop_sunni    0.0363425  0.0029937  12.140  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

The second (mod2) is a **Generalized Linear Model with a quasi-Poisson link**. This parameterization is designed to accommodate dependent variables that are (over-dispersed) event counts

```
mod2 = glm(is_violence ~ road_density + pop_1000 + crops_mean +
    dams + prop_sunni, data=X, family="quasipoisson")
summary(mod2)
```

```
##
## Call:
## glm(formula = is_violence ~ road_density + pop_1000 + crops_mean +
##     dams + prop_sunni, family = "quasipoisson", data = X)
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.3622385  0.5788847   7.536 3.70e-12 ***
## road_density -9.2950240  5.8355866  -1.593  0.11323
## pop_1000      0.0004305  0.0001670   2.578  0.01087 *
## crops_mean   -2.1724732  0.7841517  -2.770  0.00628 **
## dams          0.8329988  0.4094748   2.034  0.04361 *
## prop_sunni    0.0302367  0.0051981   5.817 3.29e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 429.0671)
##
##     Null deviance: 71658  on 161  degrees of freedom
## Residual deviance: 38816  on 156  degrees of freedom
## AIC: NA
##
```

The third and fourth models (`mod3`, `mod4`) add province-level **fixed effects** (`NAME_1`), which allow each province (e.g. Anbar) to have a different baseline level of violence

```
mod3 = lm(log(is_violence+1) ~ road_density + pop_1000 + crops_mean +
    dams + prop_sunni + NAME_1, data=X)
mod4 = glm(is_violence ~ road_density + pop_1000 + crops_mean +
    dams + prop_sunni + NAME_1, data=X, family="quasipoisson")
```

`mod3` is OLS, `mod4` is quasi-Poisson

We can use the `stargazer()` command to export the models' coefficient estimates into a formatted table, which you could add to a paper or report:

```
stargazer::stargazer(mod1,mod2,mod3,mod4,type = "text",
          keep = c("road_density","pop_1000","crops_mean",
            "dams","prop_sunni"),
          add.lines = list(c("Province FE","N","N","Y","Y")))
```

```
##
## ================================================================================================
##                                              Dependent variable:
##                          ------------------------------------------------------------------------
##                           log(is_violence + 1)      is_violence      log(is_violence + 1)      is_violence
##                                   OLS           glm: quasipoisson           OLS           glm: quasipoisson
##                                                    link = log                                 link = log
##                                   (1)                 (2)                 (3)                 (4)
## ------------------------------------------------------------------------------------------------
## road_density                     -1.486              -9.295              -1.845             -12.958***
##                                  (3.175)             (5.836)             (3.525)             (4.943)
##
## pop_1000                        0.001***            0.0004**            0.002***            0.001***
##                                 (0.0002)            (0.0002)            (0.0003)            (0.0002)
##
## crops_mean                      -1.427***           -2.172***           -2.204***           -1.721***
##                                  (0.484)             (0.784)             (0.485)             (0.644)
##
## dams                             1.111**             0.833**             0.647               0.603*
##                                  (0.535)             (0.409)             (0.430)             (0.315)
##
## prop_sunni                      0.036***            0.030***            0.017***            0.016***
##                                  (0.003)             (0.005)             (0.005)             (0.005)
##
## ------------------------------------------------------------------------------------------------
## Province FE                         N                   N                   Y                   Y
## Observations                      162                 162                 162                 162
## R2                               0.569                                   0.819
## Adjusted R2                      0.555                                   0.767
## Residual Std. Error    1.547 (df = 156)                    1.121 (df = 125)
## F Statistic         41.196*** (df = 5; 156)             15.695*** (df = 36; 125)
## ================================================================================================
## Note:                                                              *p<0.1; **p<0.05; ***p<0.01
```

What does this tell us about whether the data support our Hypotheses?

Hypothesis	Expectation	Confirm? (OLS)	Confirm? (QP)
1. Power projection	$\beta_{\texttt{road\_density}} < 0$	✗	✓
2. Demographics	$\beta_{\texttt{pop\_1000}} > 0$	✓	✓
3. Political economy	$\beta_{\texttt{crops\_mean}} < 0$	✓	✓
4. Key infrastructure	$\beta_{\texttt{dams}} > 0$	✗	✓
5. Sectarian divisions	$\beta_{\texttt{prop\_sunni}} > 0$	✓	✓