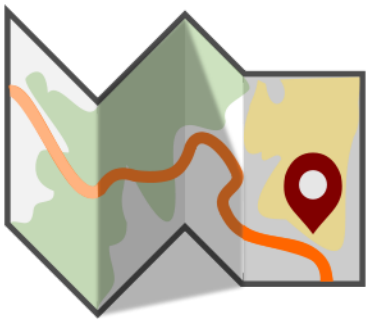
Performing Table Joins (PyQGIS)

QGIS Tutorials and Tips



Author

Ujaval Gandhi

http://google.com/+UjavalGandhi

Translations by

Sorin Călinică

Realizarea Uniunii Tabelelor (PyQGIS)

Acest tutorial vă arată cum să folosiți scripturi Python în QGIS (PyQGIS), pentru a efectua o îmbinare de tabele și pentru a aplica un stil gradat stratului rezultat. Acest tutorial reproduce pașii din tutorialul Unificarea tabelelor folosind doar scriptarea Python.

Privire de ansamblu asupra activității

Parcurgeți tutorialul Unificarea tabelelor pentru o vedere de ansamblu.

Alte competen■e pe care le ve■i dobândi

- Încărcarea în QGIS a straturilor arhivate, prin intermediul Python.
- Folosirea QgsGraduatedSymbolRendererV2 pentru a aplica un stil gradat stratului vectorial.

Obținerea datelor

Descărcați următoarele fișiere în computerul dvs.

tl_2013_06_tract.zip

ca_tracts_pop.csv

ca_tracts_pop.csvt

Surse de date: [TIGER] [USCENSUS]

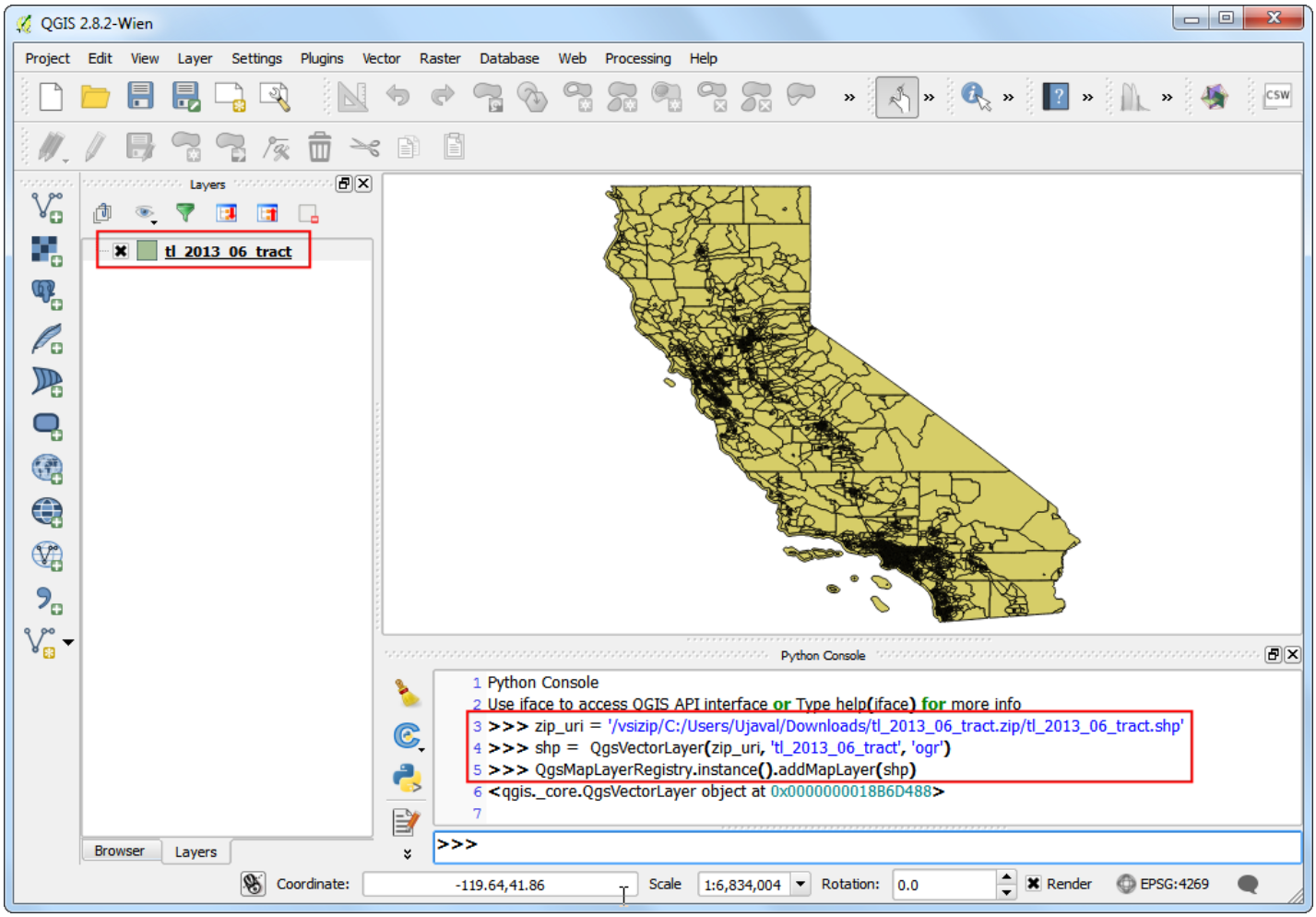
Procedura

Introduceți următoarele comenzi în Consola Python sau în Editorul încorporat în QGIS.

1. Încărcați fișierul shape. Fișierul Census Tracts este fișierul zip care conține fisierul shape. În timp ce dezarhivăm și încărcăm fișierul shape, furnizorul OGR are capacitatea de a încărca fișierul zip direct prin unui Sistem de Fișiere Virtual. Prin adăugarea în calea ***/vsizip/***, putem accesa fișierul shape conținut în arhiva zip.

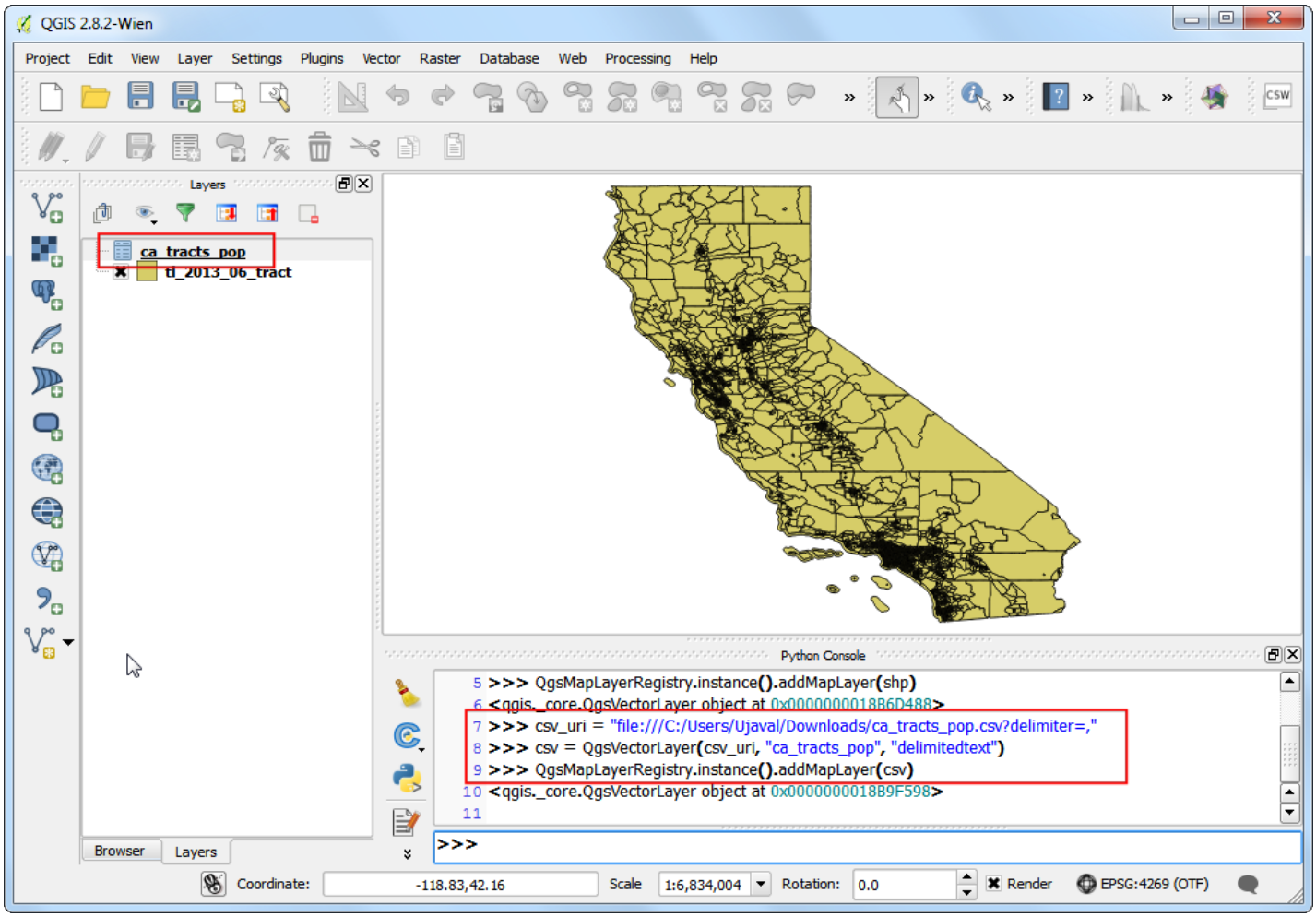
> ***Note***
>
> URI-ul zip va începe cu ***/vsizip//*** pe sistemele Linux și Mac. (Remarcați caracterul / suplimentar)

```
zip_uri = '/vsizip/C:/Users/Ujaval/Downloads/tl_2013_06_tract.zip'
shp =  QgsVectorLayer(zip_uri, 'tl_2013_06_tract', 'ogr')
QgsMapLayerRegistry.instance().addMapLayer(shp)
```



2. Încărcați fișierul CSV. Atât timp cât fișierul CSV nu conține date spațiale, îl vom încărca sub forma unui tabel folosind furnizorul ***`delimitedtext`***.

```
csv_uri = 'file:///C:/Users/Ujaval/Downloads/ca_tracts_pop.csv?delimiter=,'
csv = QgsVectorLayer(csv_uri, 'ca_tracts_pop', 'delimitedtext')
QgsMapLayerRegistry.instance().addMapLayer(csv)
```

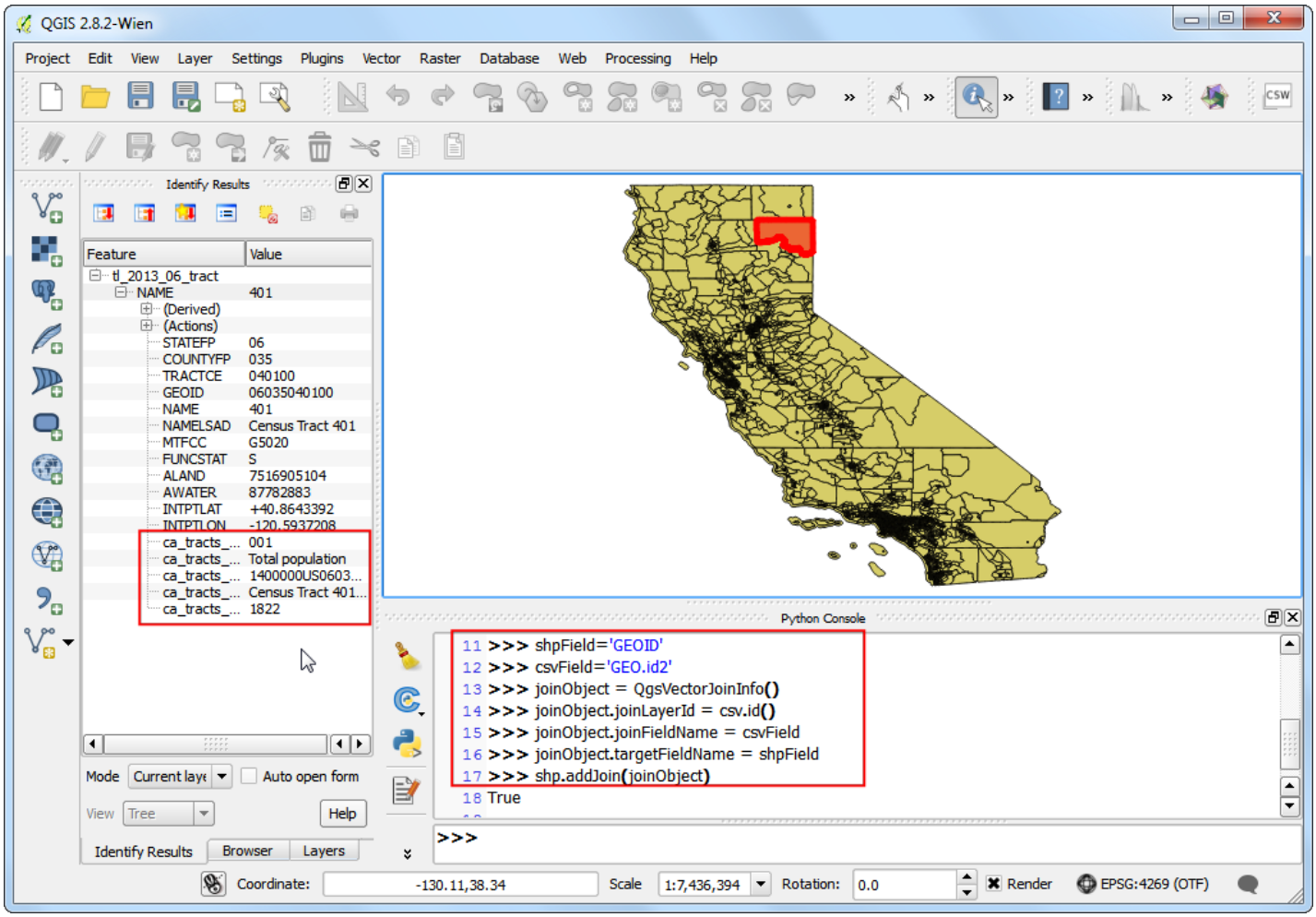


3. Realizați uniunea tabelelor. Îmbinările tabelelor sunt efectuate folosindu-se obiectul ***QgsVectorJoinInfo***. Avem nevoie de câmpul ***GEO.id2*** din stratul CSV ca și câmp de îmbinare și de câmpul ***GEOID*** din stratul fișierului shape ca și Câmp Destinație. O dată ce executați următorul cod, stratul fișierului shape va avea atributele suplimentare aduse din stratul csv.

Note

O capcană comună atunci când se utilizează QgsVectorJoinInfo, este faptul că ambele straturi trebuie să fie încărcate în QgsMapLayerRegistry - în caz contrar, îmbinarea nu ar funcționa.

```
shpField='GEOID'
csvField='GEO.id2'
joinObject = QgsVectorJoinInfo()
joinObject.joinLayerId = csv.id()
joinObject.joinFieldName = csvField
joinObject.targetFieldName = shpField
joinObject.memoryCache = True
shp.addJoin(joinObject)
```

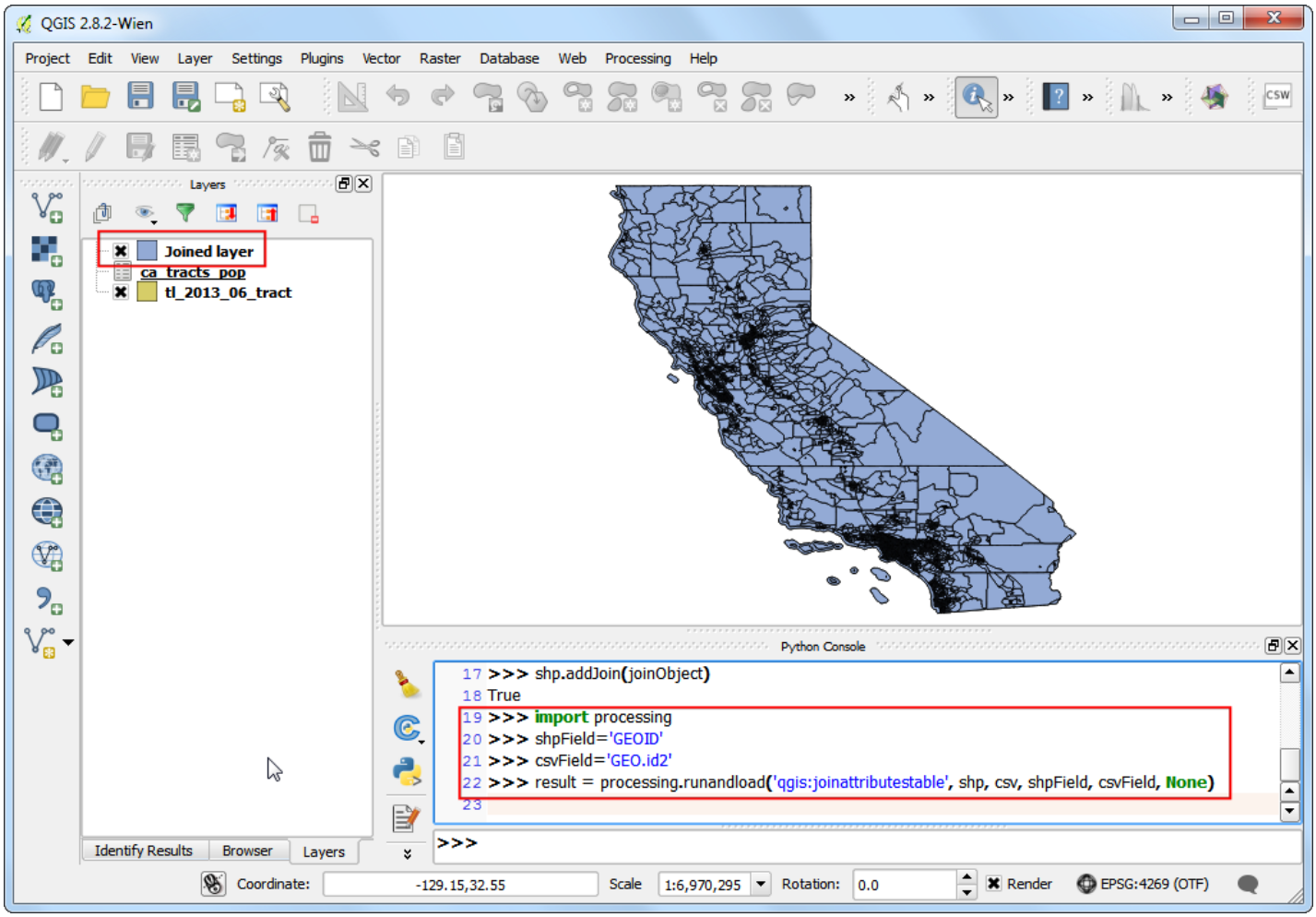


4. O modalitate mai ușoară - și preferată, de realizare a acestui lucru este prin intermediul Cadrului de Procesare. Puteți apela algoritmul ***`qgis:joinattributestable`***, apoi să creați un strat îmbinat.

Note

Folosim metoda ***`processing.runandload()`*** pentru a executa algoritmul în locul mult mai obișnuitei ***`processing.runalg()`***. Din moment ce ne dorim să încărcăm stratul îmbinat în QGIS, ***`processing.runandload()`*** reprezintă o alegere mai bună.

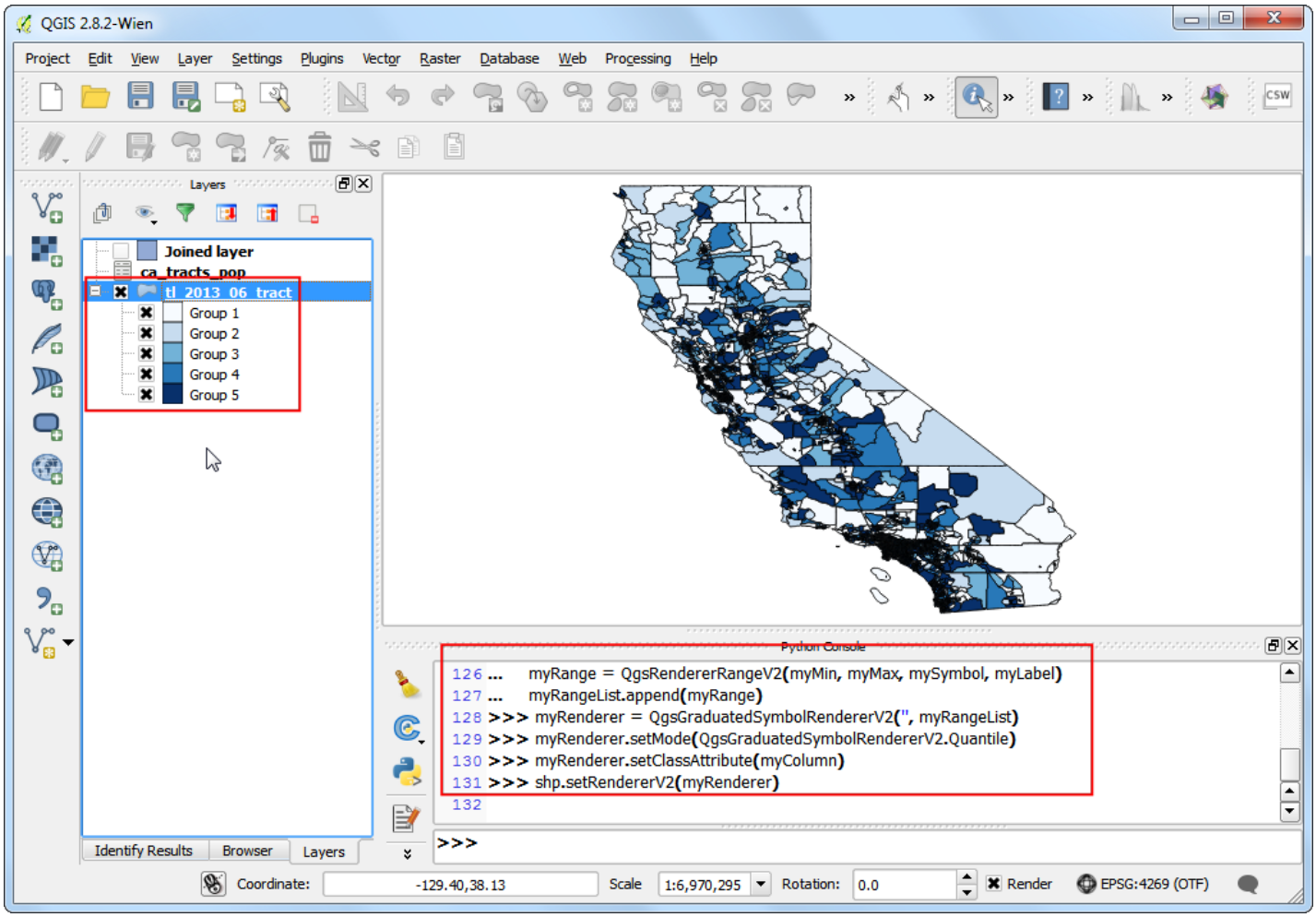
```
import processing
shpField='GEOID'
csvField='GEO.id2'
result = processing.runandload('qgis:joinattributestable', shp, csv, shpField, csvField, None)
```



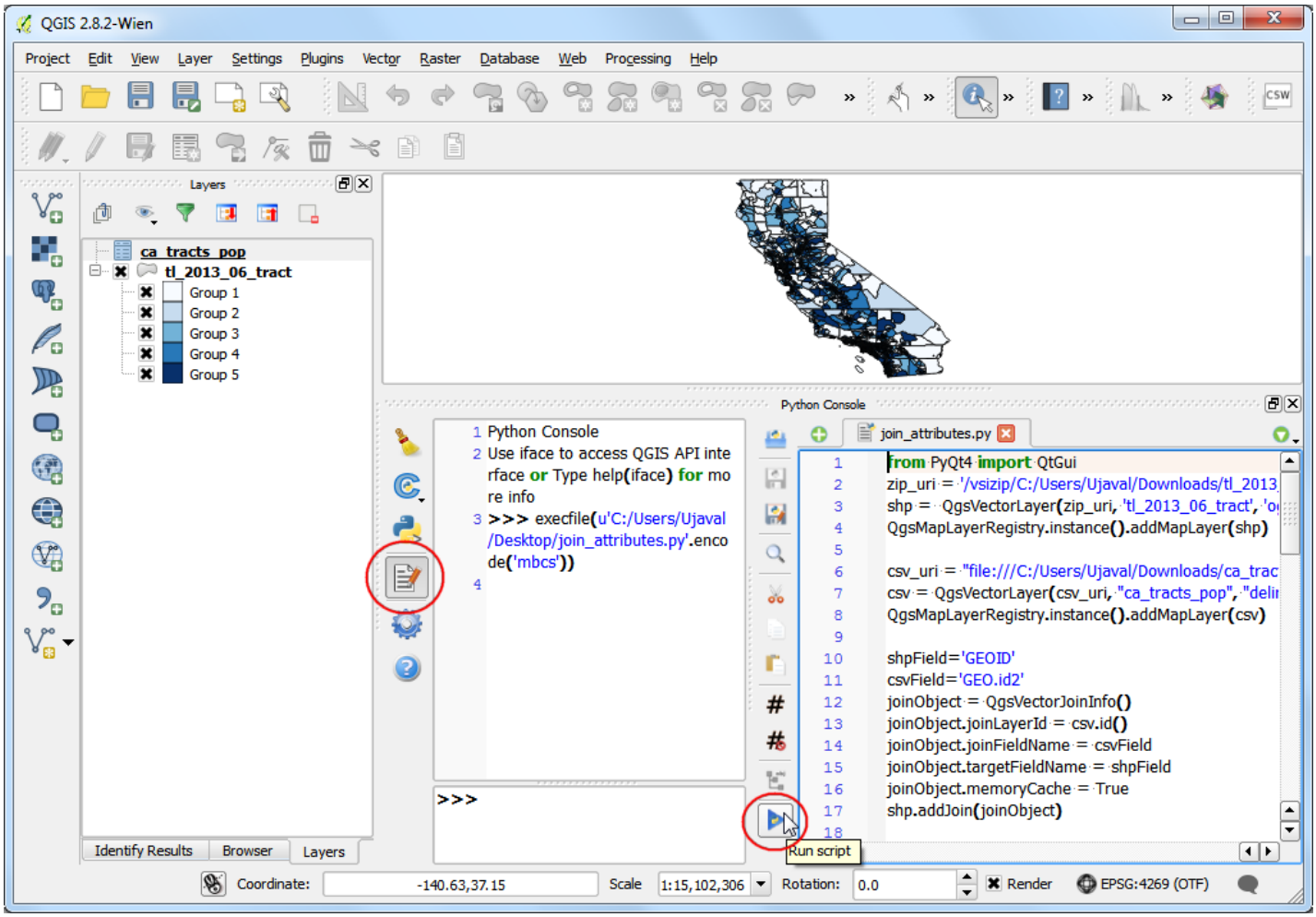
5. Vom rămâne cu îmbinarea originală, folosind ***QgsVectorJoinInfo*** pentru restul tutorialului. Acum este timpul să aplicăm stratului îmbinat un stil gradat. Denumirea câmpului pentru populație, din stratul îmbinat, este ***ca_tracts_pop_D001***. Vom aplica un render gradat folosind clasa ***QgsGraduatedSymbolRendererV2*** în modul ***Quantile***. Parcurgeți Unificarea tabelelor pentru culorile și intervalele de care avem nevoie în utilizare.

```
from PyQt4 import QtGui

myColumn = 'ca_tracts_pop_D001 '
myRangeList = []
myOpacity = 1

ranges = []

myMin1 = 0.0
myMax1 = 3157.2
myLabel1 = 'Group 1'
myColor1 = QtGui.QColor('#f7fbff')
ranges.append((myMin1, myMax1, myLabel1, myColor1))

myMin2 = 3157.2
myMax2 = 4019.0
myLabel2 = 'Group 2'
```

```
myColor2 = QtGui.QColor('#c7dcef')
ranges.append((myMin2, myMax2, myLabel2, myColor2))

myMin3 = 4019.0
myMax3 = 4865.8
myLabel3 = 'Group 3'
myColor3 = QtGui.QColor('#72b2d7')
ranges.append((myMin3, myMax3, myLabel3, myColor3))

myMin4 = 4865.8
myMax4 = 5996.4
myLabel4 = 'Group 4'
myColor4 = QtGui.QColor('#2878b8')
ranges.append((myMin4, myMax4, myLabel4, myColor4))

myMin5 = 5996.4
myMax5 = 37452.0
myLabel5 = 'Group 5'
myColor5 = QtGui.QColor('#08306b')
ranges.append((myMin5, myMax5, myLabel5, myColor5))

for myMin, myMax, myLabel, myColor in ranges:
  mySymbol = QgsSymbolV2.defaultSymbol(shp.geometryType())
  mySymbol.setColor(myColor)
  mySymbol.setAlpha(myOpacity)
  myRange = QgsRendererRangeV2(myMin, myMax, mySymbol, myLabel)
  myRangeList.append(myRange)

myRenderer = QgsGraduatedSymbolRendererV2('', myRangeList)
myRenderer.setMode(QgsGraduatedSymbolRendererV2.Quantile)
myRenderer.setClassAttribute(myColumn)

shp.setRendererV2(myRenderer)
```



6. Tastarea codului în Consola Python este utilă pentru mici activități, dar este mult mai ușor de utilizat Editorul încorporat. Puteți copia întregul script în Editor, apoi să faceți clic pe Run. O dată ce script-ul se încheie, vom crea o îmbinare de tabel și vom stiliza stratul rezultat fără pași manuali.



Mai jos este fișierul ***join_attributes.py*** complet, pentru referință.

```
from PyQt4 import QtGui
zip_uri = '/vsizip/C:/Users/Ujaval/Downloads/tl_2013_06_tract.zip/tl_2013_06_tract.shp'
shp =  QgsVectorLayer(zip_uri, 'tl_2013_06_tract', 'ogr')
QgsMapLayerRegistry.instance().addMapLayer(shp)

csv_uri = "file:///C:/Users/Ujaval/Downloads/ca_tracts_pop.csv?delimiter=,"
csv = QgsVectorLayer(csv_uri, "ca_tracts_pop", "delimitedtext")
QgsMapLayerRegistry.instance().addMapLayer(csv)

shpField='GEOID'
csvField='GEO.id2'
joinObject = QgsVectorJoinInfo()
joinObject.joinLayerId = csv.id()
joinObject.joinFieldName = csvField
joinObject.targetFieldName = shpField
joinObject.memoryCache = True
shp.addJoin(joinObject)

myColumn = 'ca_tracts_pop_D001 '
myRangeList = []
myOpacity = 1

ranges = []
```

```
myMin1 = 0.0
myMax1 = 3157.2
myLabel1 = 'Group 1'
myColor1 = QtGui.QColor('#f7fbff')
ranges.append((myMin1, myMax1, myLabel1, myColor1))

myMin2 = 3157.2
myMax2 = 4019.0
myLabel2 = 'Group 2'
myColor2 = QtGui.QColor('#c7dcef')
ranges.append((myMin2, myMax2, myLabel2, myColor2))

myMin3 = 4019.0
myMax3 = 4865.8
myLabel3 = 'Group 3'
myColor3 = QtGui.QColor('#72b2d7')
ranges.append((myMin3, myMax3, myLabel3, myColor3))

myMin4 = 4865.8
myMax4 = 5996.4
myLabel4 = 'Group 4'
myColor4 = QtGui.QColor('#2878b8')
ranges.append((myMin4, myMax4, myLabel4, myColor4))

myMin5 = 5996.4
myMax5 = 37452.0
myLabel5 = 'Group 5'
myColor5 = QtGui.QColor('#08306b')
ranges.append((myMin5, myMax5, myLabel5, myColor5))

for myMin, myMax, myLabel, myColor in ranges:
    mySymbol = QgsSymbolV2.defaultSymbol(shp.geometryType())
    mySymbol.setColor(myColor)
    mySymbol.setAlpha(myOpacity)
    myRange = QgsRendererRangeV2(myMin, myMax, mySymbol, myLabel)
    myRangeList.append(myRange)

myRenderer = QgsGraduatedSymbolRendererV2('', myRangeList)
myRenderer.setMode(QgsGraduatedSymbolRendererV2.Quantile)
myRenderer.setClassAttribute(myColumn)

shp.setRendererV2(myRenderer)
```