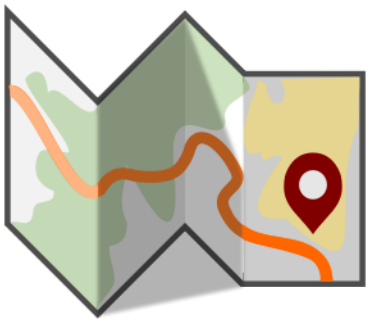
Performing Table Joins (PyQGIS)

QGIS Tutorials and Tips



Author

Ujaval Gandhi

http://google.com/+UjavalGandhi

Translations by

Dick Groskamp

Samenvoegen van tabellen uitvoeren (PyQGIS)

Deze handleiding laat zien hoe scripten in Python te gebruiken in QGIS (PyQGIS) om tabellen samen te voegen en een graduele stijl toe te passen op de resulterende laag. Deze handleiding repliceert de stappen van de handleiding Samenvoegen van tabellen uitvoeren waarbij alleen scripten in Python wordt gebruikt.

Overzicht van de taak

Bekijk de handleiding Samenvoegen van tabellen uitvoeren voor het overzicht.

Andere vaardigheden die u zult leren

- Gezipte lagen laden in QGIS via Python.
- QgsGraduatedSymbolRendererV2 gebruiken om een graduele stijl toe te passen op een vectorlaag.

De gegevens ophalen

Download de volgende bestanden naar uw computer.

tl_2013_06_tract.zip

ca_tracts_pop.csv

ca_tracts_pop.csvt

Gegevensbron [TIGER] [USCENSUS]

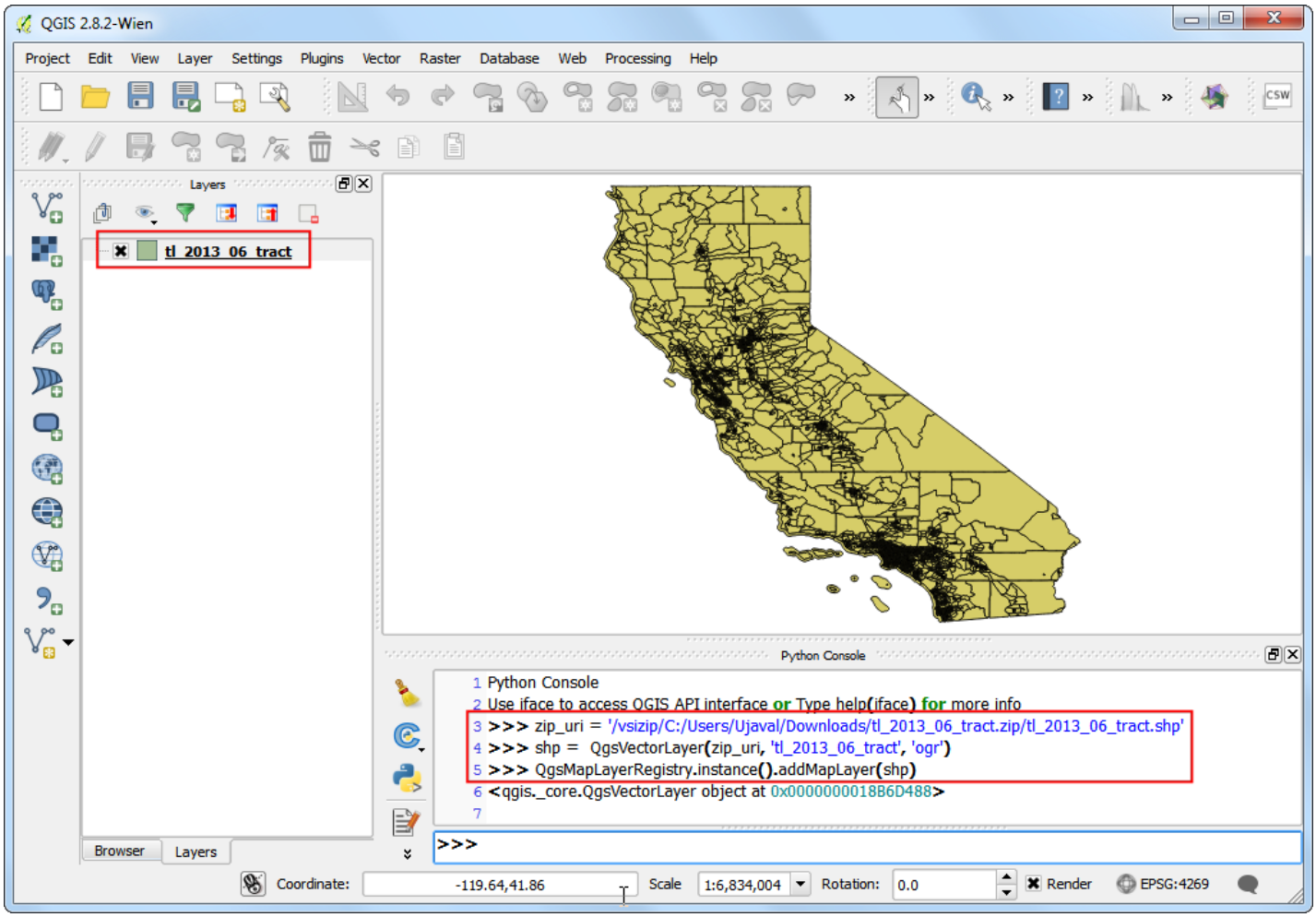
Procedure

U kunt de volgende opdrachten typen in de Python Console of de ingebouwde Editor in QGIS.

1. Laad het shapefile. Het Census Tracts-bestand is een zip-bestand dat het shapefile bevat. Waar we het kunnen uitpakken en het shapefile laden, heeft de OGR- provider de mogelijkheid om het zip-bestand direct te laden via een Virtual Filesystem. Door ***/vsizip/*** toe te voegen aan het pad, kunnen we toegang verkrijgen tot het shapefile dat is opgenomen in het zip-archief.

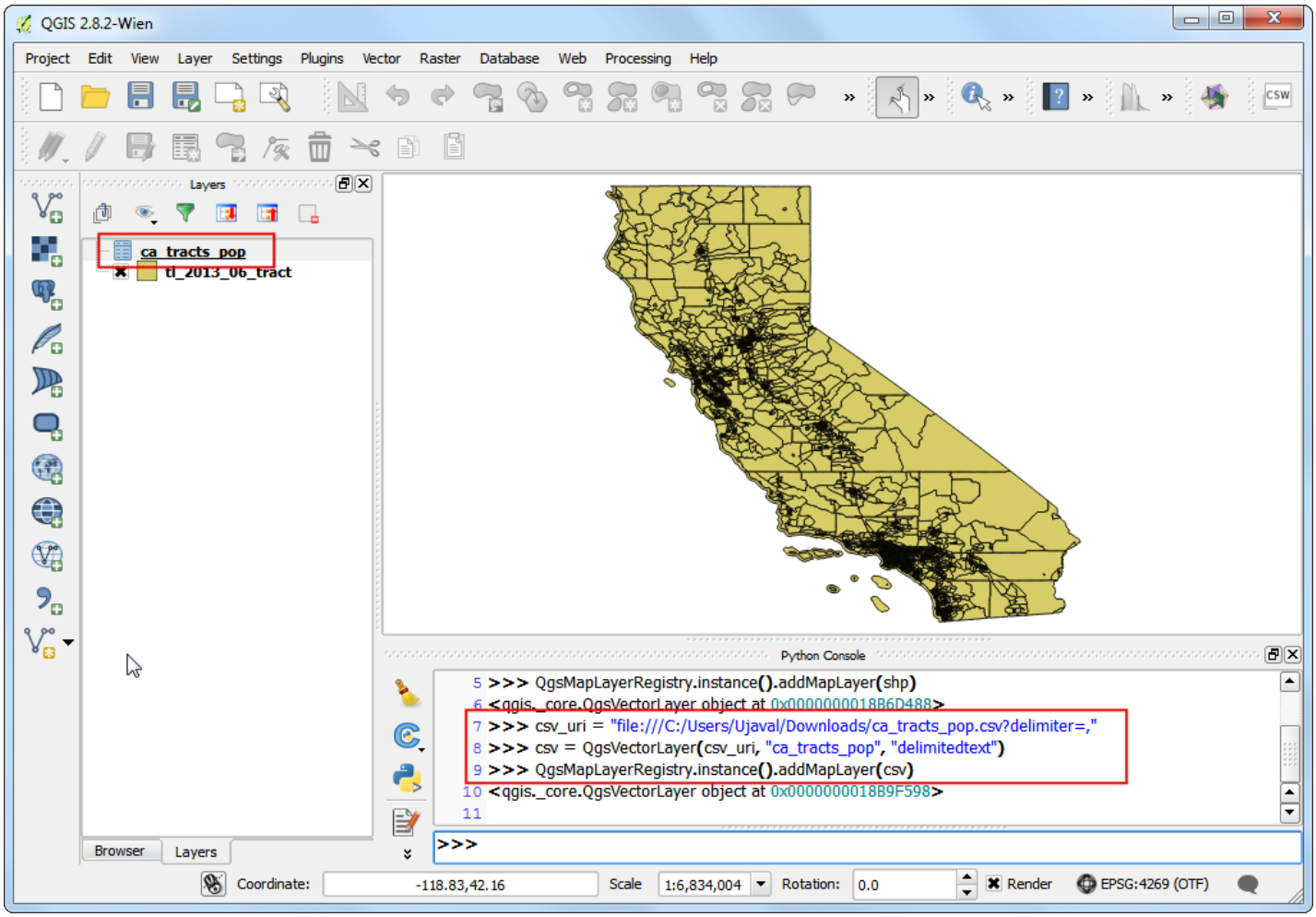
> ***Note***
>
> De zip_uri zou moeten beginnen met ***/vsizip//*** op systemen van Linux en Mac. (Let op de extra /)

```
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. Laad het CSV-bestand. Omdat het CSV-bestand geen ruimtelijke gegevens bevat, laden we het als een tabel met behulp van de provider ***tekstgescheiden***.

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

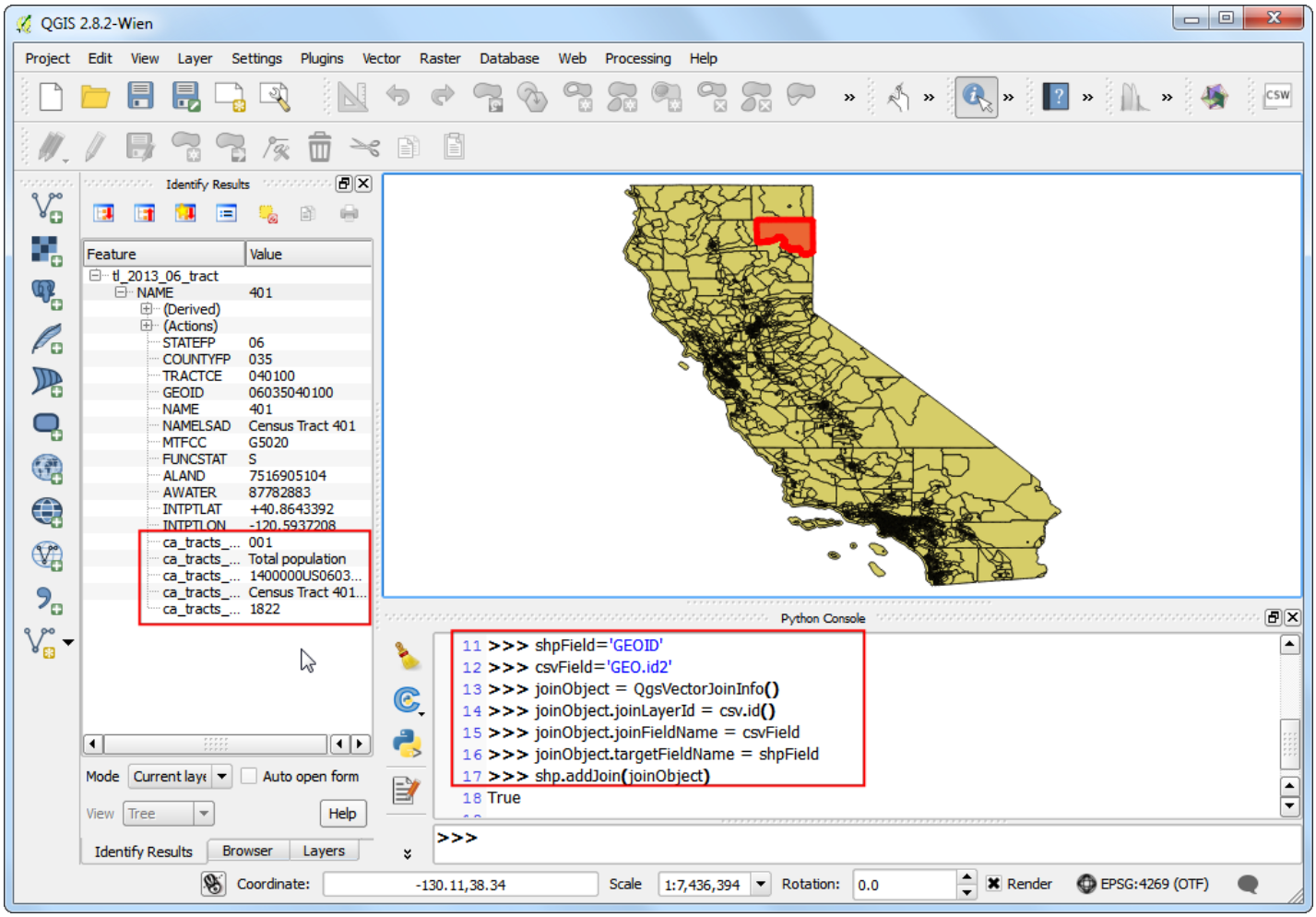


3. De samenvoeging van de tabellen maken. Samenvoegingen van tabellen in QGIS worden uitgevoerd met behulp van het object ***QgsVectorJoinInfo***. We dienen het veld ***GEO.id2*** van de CSV-laag te specificeren als het Join Field en het veld ***GEOID*** uit de laag van het shapefile als het Target Field. Als u eenmaal de volgende heeft uitgevoerd, zal de laag van het shapefile aanvullende attributen hebben die zijn samengevoegd vanuit de CSV-laag.

Note

Een veel voorkomende valkuil bij het gebruiken van QgsVectorJoinInfo is dat beide lagen moeten zijn geladen in de QgsMapLayerRegistry - anders zal de samenvoeging niet werken.

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

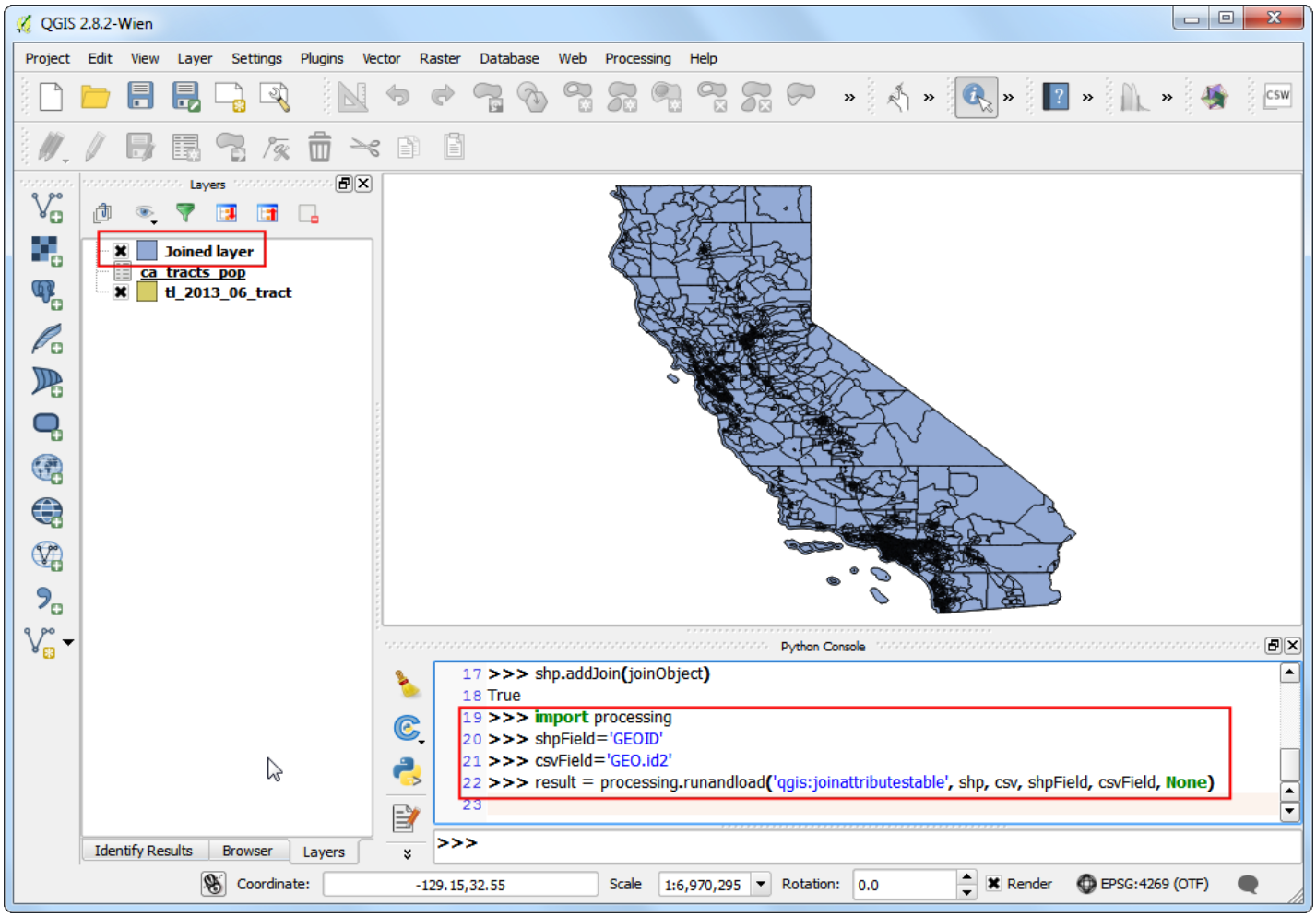


4. Een gemakkelijker - en aanbevolen manier van het bereiken van hetzelfde is via het Framework Processing. U kunt het algoritme ***qgis:joinattributestable*** aanroepen en een samengevoegde laag maken.

Note

We gebruiken de methode ***processing.runandload()*** om het algoritme uit te voeren in plaats van het meer algemene ***processing.runalg()***. Omdat we de resulterende, samengevoegde laag willen laden in QGIS, is ***processing.runandload()*** een betere kueze.

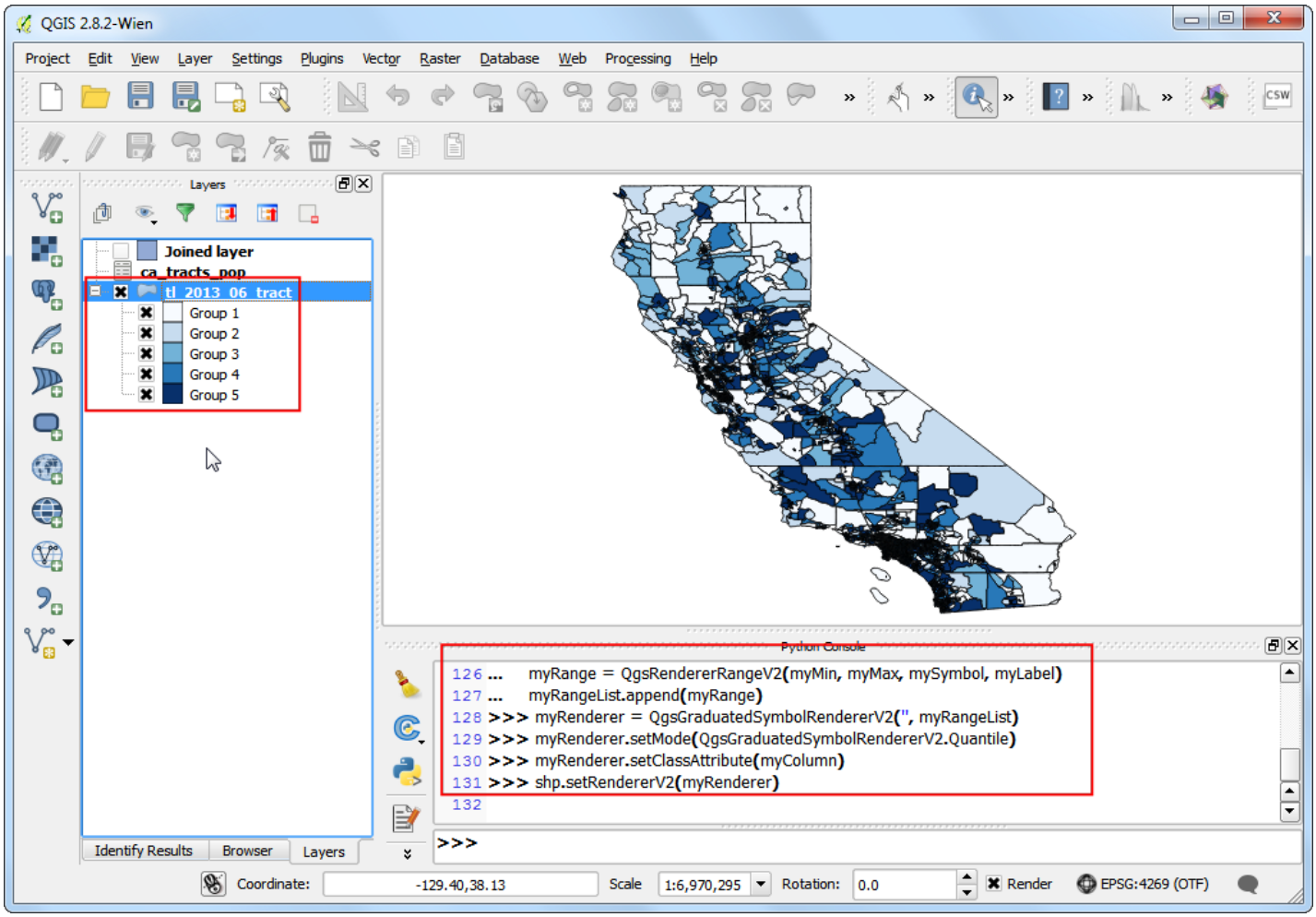
```
import processing
shpField='GEOID'
csvField='GEO.id2'
result = processing.runandload('qgis:joinattributestable', shp, csv, shpField, csvField, None)
```



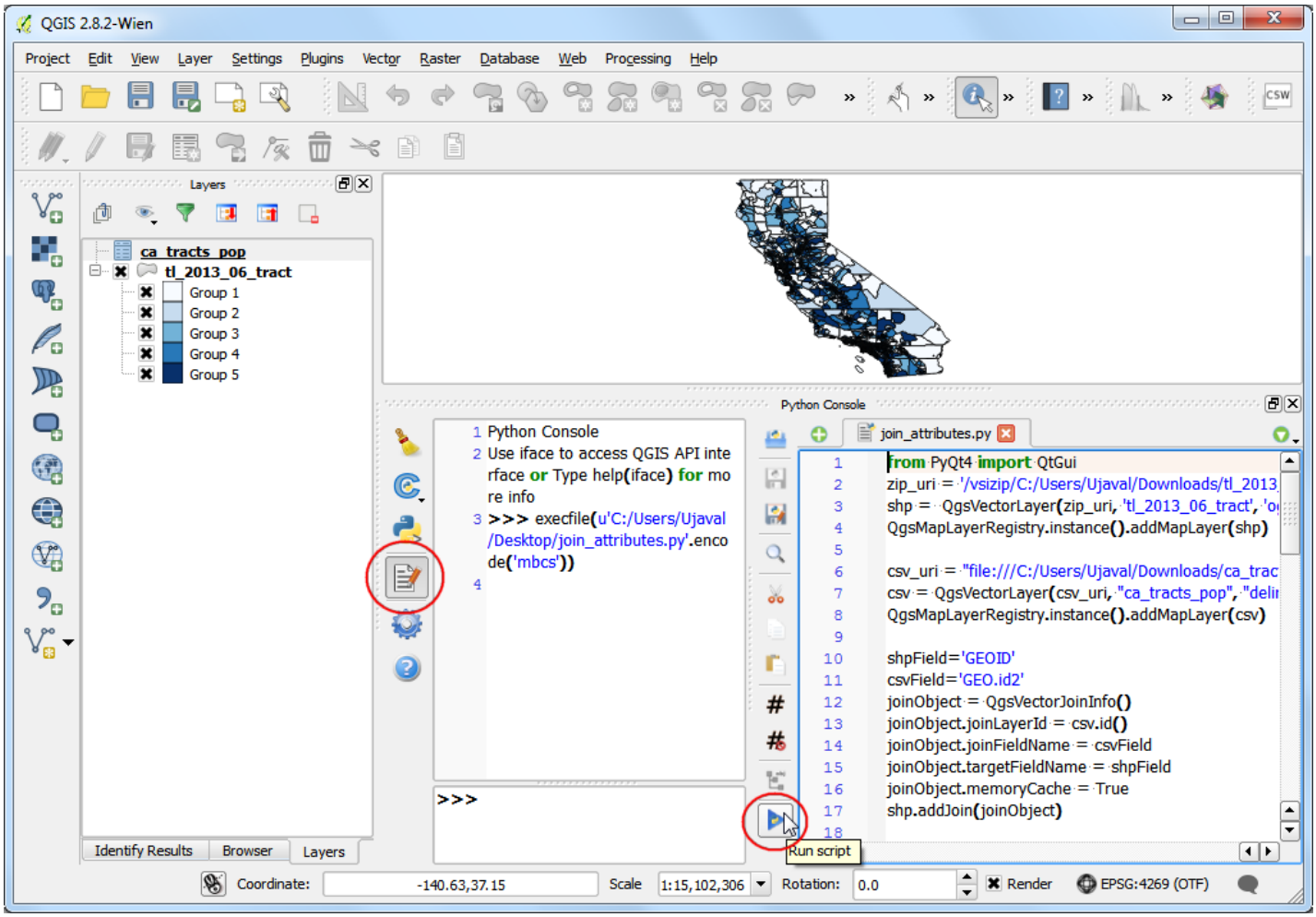
5. We zullen bij de originele samenvoeging met behulp van ***QgsVectorJoinInfo*** blijven voor de rest van de handleiding. Nu is het tijd om een graduele stijl toe te passen op de samengevoegde laag. De veldnaam voor de bevolking in de samengevoegde laag is ***ca_tracts_pop_D001***. We zullen een renderer Gradueel toepassen met behulp van de klasse ***QgsGraduatedSymbolRendererV2*** in de modus ***Kwantiel***. Bekijk Samenvoegen van tabellen uitvoeren voor de kleuren en bereiken die we moeten gebruiken.

```
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. Typen van de code in de Python Console is nuttig voor kleine taken, maar het is eenvoudiger de ingebouwde Editor te gebruiken. U kunt het volledige script kopiëren naar de Editor en klikken op Run. Al het script is voltooid, zou u een samenvoeging van tabellen hebben gemaakt en de resulterende laag hebben opgemaakt, zonder handmatige stappen.



Hieronder staat, als referentie, het volledige bestand ***join_attributes.py***.

```
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)
```