



# CountyStat PDF Template

Tue June 15, 2021 02:16 PM

## Contents

<b>Getting Started</b>	<b>2</b>
<b>Common Standards</b>	<b>2</b>
Loading Data . . . . .	2
Database Connections . . . . .	2
Reading Flat Files . . . . .	2
Spatial Data . . . . .	2
Census Data . . . . .	3
Charts and Graphs . . . . .	4
Colors . . . . .	4
Tables . . . . .	5
<b>Mapping</b>	<b>6</b>
ggplot2 map . . . . .	6
ggmap . . . . .	7



## Getting Started

For many of the basics in [RMarkdown](#) the RStudio introduction is very worth while. Almost every R visualization output is compatible with it.

For this report standard if other images need to be replaced with others that is fine.

## Common Standards

### Loading Data

#### Database Connections

```
require(DBI)

wh_host <- Sys.getenv('WH_HOST')
wh_db <- Sys.getenv('WH_DB')
wh_user <- Sys.getenv('WH_USER')
wh_pass <- Sys.getenv('WH_PASS')

# Connect to the CountyStat DataWarehouse
wh_con <- dbConnect(odbc::odbc(),
                    driver = "{ODBC Driver 17 for SQL Server}",
                    server = wh_host,
                    database = wh_db,
                    UID = wh_user,
                    pwd = wh_pass,
                    Trusted_Connection= "yes")
```

Always keep your credentials for the DataWarehouse in an `.env` file. An example is in this repo, but in normal circumstances this file should be in the `.gitignore` file.

To use [DBI](#) check out their documentation, but for the most part you will be using either the `dbGetQuery()` or `dbReadTable()` function.

#### Reading Flat Files

For flat files, `readr` will do well for most CSV's and other basic file formats, `readxl` is best for excel documents, `jsonlite` is ideal for JSON documents, and `xml2` will do if you have the unfortunate circumstance of dealing with that format.

```
require(readr)
require(readxl)
require(jsonlite)
require(xml2)
```

#### Spatial Data

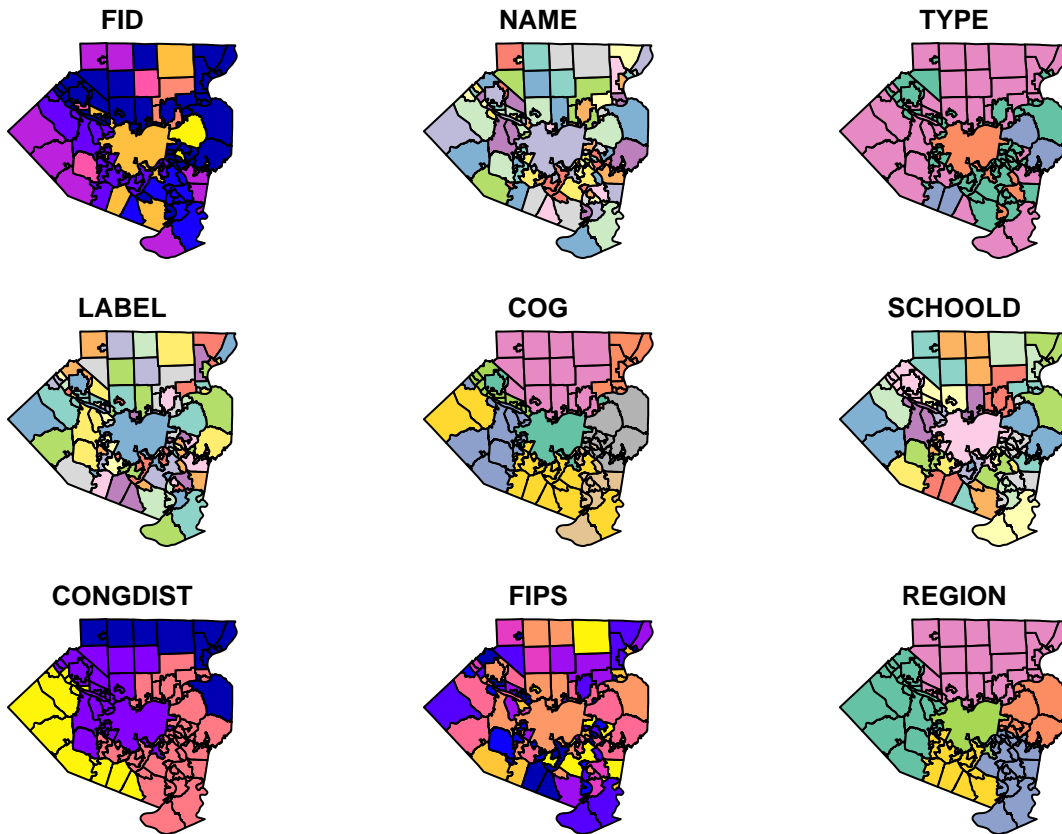
The easiest method for reading data and mapping in either `ggplot2` or `ggmap` is the [sf](#) package. When looking for a spatial dataset, the first option is always what is available from the County ESRI team on their [GIS Open Data site](#). If you can't find what you are looking for, and it is something you think the County should have access to email their team at [GISHelp@AlleghenyCounty.US](mailto:GISHelp@AlleghenyCounty.US).



```
require(sf)

munis <- read_sf('https://opendata.arcgis.com/datasets/9de0e9c07af04e638dbc9cb9070962c2_0.geojson')

plot(munis)
```



## Census Data

The easiest way to use the Census API in R is with the [tidycensus](#) package. First you have to [sign-up](#) for an api key, but its a free and easy process.

```
require(tidycensus)

census_api_key(Sys.getenv("census"))

v19 <- load_variables(2019, "acs5", cache = TRUE)

alco_muni_pop <- get_acs("county subdivision",
  state = 'PA',
  county = 'Allegheny',
  year = 2019,
  variables = 'B01003_001',
  geometry = T) # Includes geometry column for mapping
```



## Charts and Graphs

The easiest package for quickly creating charts and graphs in R is [ggplot2](#). One of the best resources for ggplot2 visualizations is the [Topic 50 ggplot2 Visualizations](#) from Selva Prabhakaran. This includes full code and examples using you can access directly from R. This site also has a good introduction to ggplot2 if you find the one from RStudio insufficient.

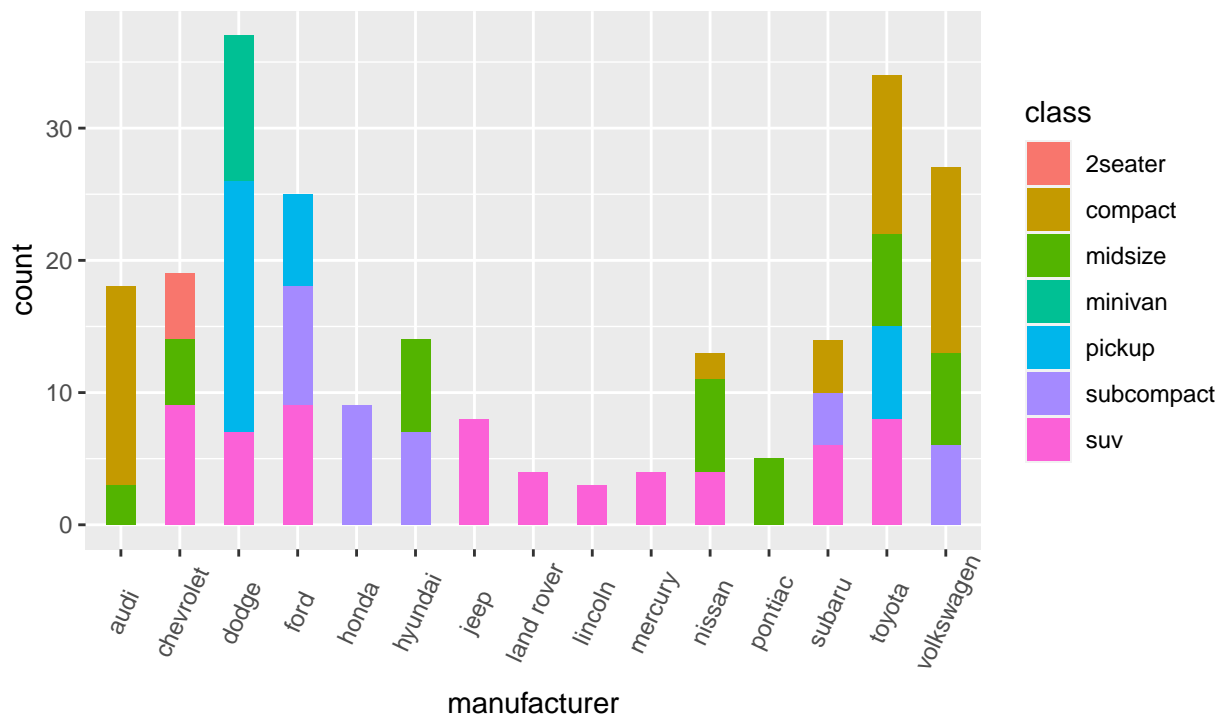
```
require(ggplot2)

g <- ggplot(mpg, aes(manufacturer)) + geom_bar(aes(fill=class), width = 0.5) +
  theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(title="Histogram on Categorical Variable",
       subtitle="Manufacturer across Vehicle Classes",
       caption = "EPA Fuel Economy 1999–2008")
```

g

### Histogram on Categorical Variable

#### Manufacturer across Vehicle Classes



EPA Fuel Economy 1999–2008

## Colors

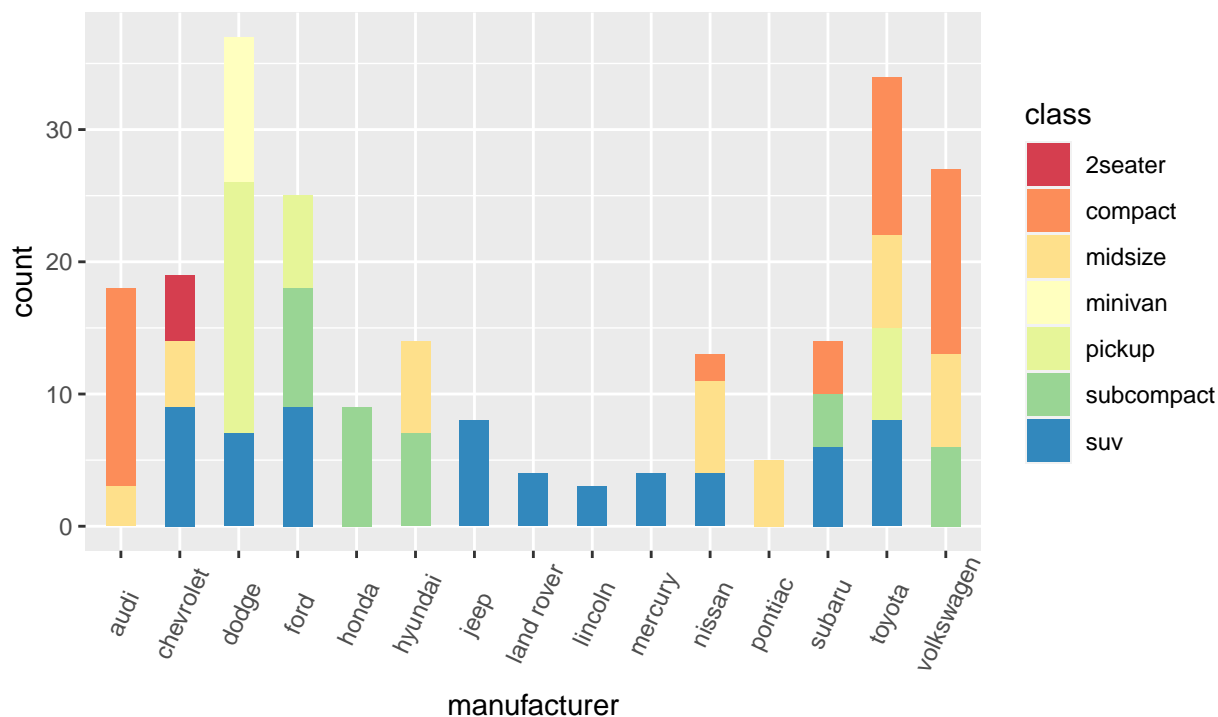
A great resource for color palettes is [colorbrewer](#), which is easy to implement in [ggplot2](#).

```
g + scale_fill_brewer(palette = "Spectral")
```



## Histogram on Categorical Variable

Manufacturer across Vehicle Classes



EPA Fuel Economy 1999–2008

## Tables

The best function for PDF table `kable()`. CountyStat has a standard table format to start with. The [kableExtra](#) package greatly expands the options you have to format your `kable()` tables. Below is the standard CountyStat format, but Analysts should feel free to use the wide variety of features the `kableExtra` library provides to improve readability and convey the message you are trying to deliver to your audience.

```
require(kableExtra)

iris %>%
  sample_n(15) %>% # Select 15 random rows
  kbl(caption = "Fisher's or Anderson's") %>%
  kable_styling(font_size = 12, latex_options = "HOLD_position") %>%
  row_spec(0, bold = T, background = "#008080", color = "white")
```



Table 1: Fisher's or Anderson's

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
6.4	3.1	5.5	1.8	virginica
5.4	3.4	1.5	0.4	setosa
5.3	3.7	1.5	0.2	setosa
4.4	3.0	1.3	0.2	setosa
6.2	2.2	4.5	1.5	versicolor
7.7	2.6	6.9	2.3	virginica
5.7	2.9	4.2	1.3	versicolor
6.0	3.4	4.5	1.6	versicolor
6.8	2.8	4.8	1.4	versicolor
5.2	4.1	1.5	0.1	setosa
6.3	2.3	4.4	1.3	versicolor
4.8	3.0	1.4	0.3	setosa
6.3	3.3	6.0	2.5	virginica
5.5	2.3	4.0	1.3	versicolor
5.0	3.6	1.4	0.2	setosa

## Mapping

### ggplot2 map

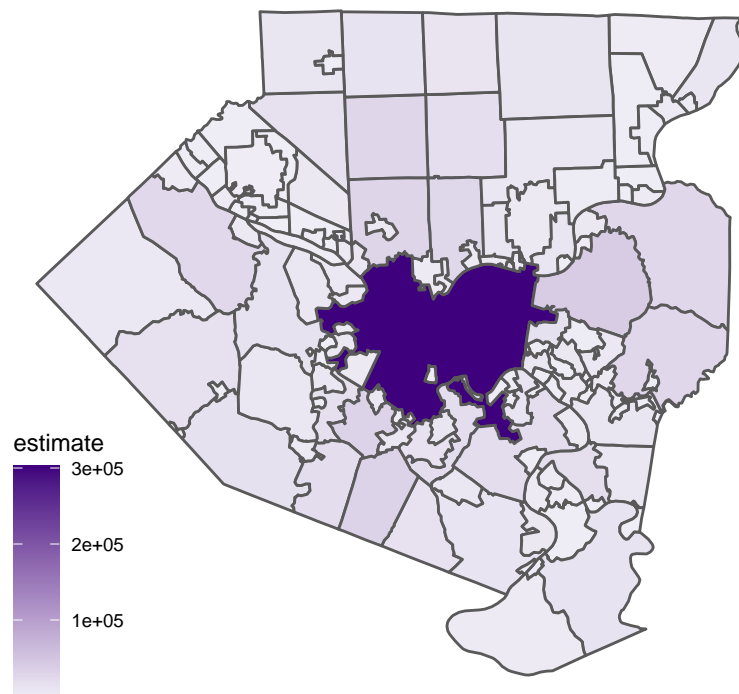
If you don't need a base map because the shapefile contains enough information for your audience about location you can just use ggplot2.

```
require(ggthemes)

ggplot(data = alco_muni_pop, aes(fill = estimate)) +
  geom_sf() +
  theme_map() +
  labs(title = "Population by Municipality",
       subtitle = "Allegheny County",
       caption = "ACS 2019 Estimate") +
  scale_fill_gradient(low = '#efedf5', high = '#3f007d')
```



## Population by Municipality Allegheny County



ACS 2019 Estimate

### ggmap

[ggmap](#) works if you need to include a base map for context. Like leaflet the Stamen.Toner base map is suggested.

```
require(ggmap)
require(ggthemes)

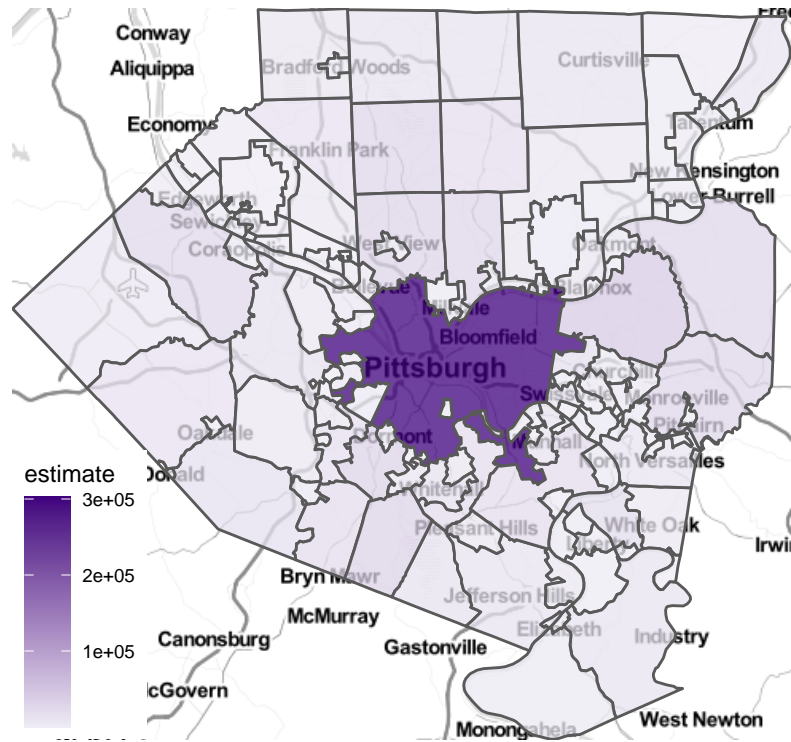
bbox <- st_bbox(munis)
bbox_trans <- c(left = bbox[[1]], bottom = bbox[[2]], right = bbox[[3]], top = bbox[[4]])

get_stamenmap(bbox_trans, maptype = "toner-lite") %>%
  ggmap() +
  theme_map() +
  labs(title = "Population by Municipality",
        subtitle = "Allegheny County",
        caption = "ACS 2019 Estimate") +
  geom_sf(data = alco_muni_pop, aes(fill = estimate), inherit.aes = FALSE, alpha = .75) +
  scale_fill_gradient(low = '#efedf5', high = '#3f007d')
```



## Population by Municipality

### Allegheny County



ACS 2019 Estimate