(I might roll this document into Vignettes, but for now it is the easiest starting place for glptools.)

**Purpose**

glptools has two main purposes. First, it houses functions that streamline many common tasks when working with Greater Louisville Project data and graphs. Second, it contains basic reference data, crosswalks, and maps that are clean and ready to merge with data. Hopefully, using the glptools package reduces the amount of time required sourcing functions and reading in reference data. The package is designed to integrate with both the glpdata package (which houses the bulk of common GLP data) and other projects that involve similar tasks.

This document contains an overview of the contents of the glptools package and will be flushed out with information on maintaining and updating glptools.

**Structure**

The **data-raw** folder contains raw reference data and scripts used to clean and process the data. **generate\_all\_data.R** runs these scripts in sequence. They are designed to run without installing the glptools package first. Some of them require the outputs of previous ones to work. For example, MSA\_FIPS.R requires the FIPS and MSA data frames from basic\_info.R.

* **basic\_info.R** creates data frames that “define” our peer cities. The CSV files are created by hand. It generates data frames of FIPS codes, MSA codes, and state abbreviations.
* **MSA\_FIPS.R** creates a crosswalk from MSA codes to FIPS codes.
* **PUMA\_crosswalk.R** creates crosswalks from MSA and FIPS codes to PUMAs.
* **zip\_codes.R** creates crosswalks from MSA and FIPS codes to zip codes.
* **tract\_crosswalk.R** creates a crosswalks between 2000 and 2010 census tracts.
* **inflation\_and\_COLA.R** creates a data frame containing cost-of-living and inflation data.
* **maps.R** creates other crosswalks between geographies and creates SpatialPolygonDataFrames for GLP maps.
* **council\_tract.R** is going to generate a crosswalk from tracts to Metro Council Districts. I need to take another pass at this as it’s not final.
* **population.R** creates data frames of populations for peer cities. This has generally been supplanted by population data from the glpdata package to make sure we have the same population data across everything. It’s no longer called by **generate\_all\_data.R.**

The **R** folder houses functions and documentation for the reference data.

Utilities

* **operators.R** defines some versions of basic operators.
  + **%not\_in%** is the counterpart to %in%
  + **%cols\_in%** returns the column names of a data frame found in a character vector
  + **%cols\_not\_in%** returns the column names of a data frame NOT found in a character vector
  + **%p%** pastes together two strings
* **general\_utils.R** contains some basic utility functions that are useful beyond GLP
  + **assign\_col\_join** and **assign\_row\_join** let you bind data frames in a loop without checking to see if you’re in the first iteration
  + **rollmeanr** calculates the rolling mean of a vector
  + **norm\_z** normalizes a vector, generating z-scores
  + **get\_sysdata** retrieves an object from the sysdata file
  + **update\_sysdata** adds objects to the sysdata file
* **peer\_utils.R** contains functions for working with peer city data
  + **pull\_peers** subsets a data frame to GLP peer city FIPS codes or MSA codes
  + **stl\_merge** combines data for the St. Louis city and county
  + **sum\_FIPS\_to\_MSA** adds MSAs to a data frame with FIPS codes and summarizes the data
  + **COLA** adjusts data for inflation and/or regional price parity
* **map\_utils.R** contains functions for working with tract-level or other neighborhood-level GLP data
  + **process\_map** creates data at the neighborhood level from a data frame containing tract data. **process\_map\_og** does the same thing but doesn’t incorporate the var\_type column.
  + **tract\_to\_council** converts data from the tract level to the Metro Council District level
  + **tract\_00\_to\_10** converts data from 2000 Census tracts to 2010 Census tracts
  + **tract\_10\_to\_00** converts data from 2010 Census tracts to 2000 Census tracts
* **vartype\_utils.R** contains functions useful for dealing with the var\_type column
  + **sum\_by\_var\_type** is useful for collapsing groups of data across demographics (e.g. summarizing estimates, margins of error, and populations across gender)
  + **pivot\_vartype\_longer** converts var\_type from column names to values in a var\_type column, and names from a “variables” column to their own columns
  + **pivot\_vartype\_wider** converts var\_type from values in a var\_type column to column names, and variable names to values in a “variable” column
* **glp\_utils.R** contains other utilities for working with specific situations in GLP data
  + **glp\_load\_packages** loads the packages used often for other functions
  + **df\_type** returns the geography of a data frame (but should probably be removed)
  + **bind\_df** joins any number of data frames by the usual GLP ID variables
  + **reshape\_sex** converts data from a wide format with suffixes like “.female” into a long format with a *sex* column.
  + **total\_demograhpics** aggregates data that is broken down by demographic to produce totals
  + **organize** reorganizes GLP data to the same layout (useful at the end of a script)
  + **unique\_check** checks to see the number of observations by group. Usually useful for confirming that you have exactly the same number of observations for every demographic.
  + **complete\_check** confirms that every combination of demographics is included in the data. (I don’t think this works yet.)
  + **years\_in\_df** returns the years in a data frame for which a variable is available for a certain combination of demographics. Useful if you’ve combined a lot of different variables with different years of availability into one data frame.
  + **add\_population** adds population data to the data frame
  + **per\_capita\_adj** adjusts variables by population
  + **complete\_vector\_arg** is a version of the *complete* function that is designed to operate on GLP data and accept a character vector as an argument.

Getting and reading data

* **census\_api.R** contains functions to download and clean data from the Census Bureau
  + **build\_census\_var\_df** chopsdowncensus\_api\_vars
  + **get\_census** calls the census API for different combinations of cities and years.
  + **process\_census** is a huge algorithm to process downloaded census data
* **readers.R** contains functions to help read folders of files from the ACS, BRFSS, CBP, SAIHIE, CPS unemployment, or generic .csv files. They generally read each file, subset it to peer cities, add a year column, tidy it, and output a single data frame.
  + **any\_time** is a generic version of the following method that can be used on any folder of tabular data with a custom function
  + **acs\_time** reads American Community Survey data (mostly deprecated due to the Census API)
  + **brfss\_time** reads data from the BRFSS survey.
  + **business\_time** reads data from County Business Patterns.
  + **insurance\_time** reads Small Area Income and Health Insurance data.
  + **unemployment\_time** reads CPS unemployment data.
  + **wonder\_time** reads data from CDC Wonder.
  + **wonder\_time\_age** reads data from CDC Wonder where each file represents an age group

Processing data

* **microdata.R** contains functions to process ACS and CPS microdata, as well as summarize microdata by peer geography and demographic group.
  + **clean\_acs\_micro** cleans an ACS download
  + **clean\_cps\_mico** cleans a CPS download
  + **survey\_by\_demog** processes census microdata and produces a clean data frame broken down by demographic. The svy\_by\_demog function is mostly a wrapper around either **svy\_bootstrap** and **svy\_repwts**, which do the actual calculation using either ACS replicate weights or bootstrap estimates.
* **wonder.R** containsfunctionsspecifically to process CDC wonder data.
  + **clean\_wonder** cleans the raw CDC wonder data
  + **age\_adj\_rate** calculates an age-adjusted rate from the output of **clean\_wonder**
  + **stl\_adj\_wonder** combines St. Louis counties

Graphing data

* **ranking.R** contains a function to produce GLP ranking graphs.
  + **ranking** produces a GLP ranking graph
  + **ranking\_data** outputs the data that would be included in the ranking graph produced by **ranking.** Useful for adding ranking numbers to a data frame without producing an image.
* **trendline.R** contains wrapper functions used to produce GLP-style trendline graphs (or output the data behind them.) Each of these functions call **tl**, described in the next file.
  + **trend** produces a trendline graph comparing Louisville to summary stats (25th and 75th percentile) of peer cities
  + **trend\_maxmin** produces a trendline graph comparing Louisville to the most-improved and least-improved peer city over a timefrrame
  + **trend\_data** outputs the data points that would be graphed by calling **trend**
  + **trend\_data\_maxmin** outputs the data points that would be graphed by calling **trend\_maxmin**
* **trendline\_helpers.R** contains a common trendline function that is called by the trendline wrapper functions. The main trendline function calls upon several intermediate functions designed to make the function more digestible.
  + **tl** calls each of the following functions and handles the inputs and outputs of each.
  + **make\_list** creates a named list of objects so functions can return multiple objects at once.
  + **tl\_filter** subsets a data frame by demograhpics and peer city set
  + **tl\_reshape\_data** and **tl\_reshape\_data\_maxmin** reshape the data frame to long format
  + **tl\_rolling\_mean** applies a rolling mean to the data
  + **tl\_year\_breaks** adds any breaks in the data to reflects years that shouldn’t be compared
  + **tl\_add\_line\_data** and **tl\_add\_line\_data\_maxmin** create labels, levels, factors, etc. to determine the line characteristics in the graph.
  + **tl\_break\_settings** calculates the years that should be marked along the x-axis
  + **tl\_plot** sets up the initial plot object
  + **tl\_limits** adds axis limits and plot labels
  + **tl\_style** adds style elements
  + **tl\_lines** and **tl\_lines\_maxmin** add the lines to the graph
* **map.R** contains a function to produce leaflet graphs of Louisville data.
  + **make\_map** creates a leaflet map object using GLP data
  + **make\_map\_00** creates a leaflet map using GLP data and old (2000) Census tracts, but should be deprecated

Other

* **data.R** contains documentation for reference data.

**Using and updating glptools**

Some of scripts rely on reference data produced in other scripts (such as a list of GLP peer FIPS codes or an MSA to FIPS crosswalk). The file data-raw/generate\_all\_data.R runs each script sequentially so that the global environment is populated with any necessary objects.

After editing a script, run **generate\_all\_data.R** to update all of the reference information.