Life expectancy

First, we'll read in a TSV file containing the most recent CIA World Factbook data using the meta-csv library.

```
(def cia-factbook
  (csv/read-csv "./datasets/cia-factbook.tsv"))
```

▸(▸{"Airp/cap" "0.28693821" "Airports" "389000000" "Birthrate" "12.17" "Cell phones" "

Expanding the results in the data viewer tells us that there are some `nil` values in columns of interest, and that our TSV importer was thrown off by this fact and so didn't convert the numerical columns to number types.

We're going to post process this table a bit with ordinary Clojure sequence functions to filter out rows that have `nil`s for our columns of interest, select those rows, convert strings to numbers, and — because we're Clojurists — convert keys to keywords.

```
(def life-expectancy
  (->> cia-factbook
       (remove #(some nil? (map (partial get %) ["Country" "GDP/cap" "Lif
       (map #(sorted-map :country (str/trim (get % "Country"))
                         :gdp (read-string (get % "GDP/cap"))
                         :life-expectancy (read-string (get % "Life expec
```

▸(▸{:country "China" :gdp 9800 :life-expectancy 75.15} ▸{:country "India" :gdp 4000 :l

Things look pretty good in the data structure browser, but it would be easier to get an overview in tabular form. Luckily, Clerk's built in table viewer is able to infer how to handle all of the most common configurations of rows and columns automatically.

```
(clerk/table life-expectancy)
```

:country	:gdp	:life-expectancy
China	9800	75.15
India	4000	67.8
European Union	34500	80.02