EV Power – Lab 4 Project Report

Example Solution 1

Part 0: libraries

```{r} pkgs <- c("tidyverse", "janitor", "maps", "scales") to_install <- setdiff(pkgs, row-names(installed.packages())) if (length(to_install)) install.packages(to_install, repos = "https:// cloud.r-project.org") invisible(lapply(pkgs, require, character.only = TRUE))

# helpers

norm_state <- function(x) x |> stringr::str_to_lower() |> stringr::str_squish()

abbr2name <- setNames(tolower(state.name), tolower(state.abb)) abbr2name <- c(abbr2name, "dc" = "district of columbia")

# return a tibble(state, value, year) from either wide (2-letter columns) or long files

read_state_value <- function(path, value_pattern, year) { df <- readr::read_csv(path, show_col_types = FALSE) |> janitor::clean_names()

# Case A: long with a state column already if ("state" %in% names(df)) { val_col <- names(df)[tidyselect::eval_select(rlang::expr(matches(value_pattern)), df) |> names()][1] stopifnot(!is.na(val_col)) out <- df |> mutate(state = norm_state(state), value = readr::parse_number(as.character(.data[[val_col]])), year = year) |> select(state, value, year) return(out) }

# Case B: wide with two-letter state abbreviations abbr_cols <- grep("[1]{2}$", names(df), value = TRUE) if (!length(abbr_cols)) stop("No state columns found in:", path)

out <- df |> tidyr::pivot_longer(all_of(abbr_cols), names_to = "abbr", values_to = "value") |> mutate(state = abbr2name[tolower(abbr)], value = readr::parse_number(as.character(value)), year = year) |> filter(!is.na(state)) |> select(state, value, year) out }

# read EV registrations – handles either (state, ev_regs) or (x2, electric_vehicle_registrations...)

read_ev_2023 <- function(path) { ev <- readr::read_csv(path, show_col_types = FALSE) |> janitor::clean_names() if ("state" %in% names(ev)) { # common tidy shape val_col <- names(ev) [tidyselect::eval_select(rlang::expr(matches("reg")), ev) |> names()][1] out <- ev |> transmute(state = norm_state(state), ev_regs = readr::parse_number(as.character(.data[[val_col]]))) } else if ("x2"

[1] a-z
```

%in% names(ev)) { # DoE export commonly has x2 = state names reg_col <- names(ev)[tidyselect::eval_select(rlang::expr(matches("electric.*registration")), ev) |> names()][1] out <- ev |> transmute(state = norm_state(x2), ev_regs = readr::parse_number(as.character(.data[[reg_col]]))) } else { stop("Could not locate state & registration columns in EV CSV.") } out |> filter(!is.na(state), !is.na(ev_regs)) |> filter(state != "united states") }