Package: spanishoddata (via r-universe)

October 31, 2024

Title Get Spanish Origin-Destination Data

Version 0.0.1

Description Enables access to origin-destination (OD) provided by the Spanish Minstry of Transport, hosted at <https://www.transportes.gob.es/ministerio/proyectos-singulares/estudios-de-movilidad-con-big-data/opendata-movilidad>. It contains functions for downloading zone boundaries and associated origin-destination data. The OD datasets are large. The package eases working with them by using the database interface package 'duckdb', using an optional environment variable 'SPANISH_OD_DATA_DIR' to avoid repeated downloads, and by providing documentation demonstrating how to collect subsets of the resulting databases into memory.

License MIT + file LICENSE

URL https://rOpenSpain.github.io/spanishoddata/, https://github.com/rOpenSpain/spanishoddata

BugReports https://github.com/rOpenSpain/spanishoddata/issues

Depends R (>= 3.5.0)

Imports curl (>= 5.0.0), DBI, dplyr, duckdb (>= 0.5.0), fs, glue, here, lubridate, memuse, parallelly, purrr, readr, rlang, sf, stats, stringr, tibble, xml2

Suggests flowmapblue, flowmapper (>= 0.1.2), furrr, future, hexSticker, mapSpain, quarto, remotes, scales, testthat (>= 3.0.0), tidyverse

VignetteBuilder quarto

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/Needs/website rmarkdown

Repository https://ropenspain.r-universe.dev

RemoteUrl https://github.com/rOpenSpain/spanishoddata

RemoteRef HEAD

RemoteSha b83b6dd570b046e68c6f063ba56baacaec99183d

Contents

`spod_available_data` *Get available data list*

Description

Get a table with links to available data files for the specified data version. Optionally check (see arguments) if certain files have already been downloaded into the cache directory specified with SPANISH_OD_DATA_DIR environment variable or a custom path specified with `data_dir` argument.

Usage

```
spod_available_data(
  ver = 2,
  check_local_files = FALSE,
  quiet = FALSE,
  data_dir = spod_get_data_dir()
)
```

Arguments

`ver`	Integer. Can be 1 or 2. The version of the data to use. v1 spans 2020-2021, v2 covers 2022 and onwards.
`check_local_files`	Whether to check if the local files exist. Defaults to `FALSE`.
`quiet`	A `logical` value indicating whether to suppress messages. Default is `FALSE`.
`data_dir`	The directory where the data is stored. Defaults to the value returned by `spod_get_data_dir()`.

Value

A tibble with links, release dates of files in the data, dates of data coverage, local paths to files, and the download status.

target_url `character`. The URL link to the data file.

pub_ts `POSIXct`. The timestamp of when the file was published.

file_extension `character`. The file extension of the data file (e.g., 'tar', 'gz').

data_ym `Date`. The year and month of the data coverage, if available.

data_ymd `Date`. The specific date of the data coverage, if available.

local_path `character`. The local file path where the data is stored.

downloaded `logical`. Indicator of whether the data file has been downloaded locally.

---

spod_codebook *View codebooks for v1 and v2 open mobility data*

Description

Opens relevant vignette.

Usage

```
spod_codebook(ver = 1)
```

Arguments

`ver`	An `integer` or `numeric` value. The version of the data. Defaults to 1. Can be 1 for v1 (2020-2021) data and 2 for v2 (2022 onwards) data.

Value

Nothing, calls relevant vignette.

spod_connect *Connect to data converted to DuckDB*

Description

This function allows the user to quickly connect to the data converted to DuckDB with the `spod_convert_to_duckdb()` function. This function is a simplificaiton of the connection process. It uses

Usage

```
spod_connect(
  data_path,
  target_table_name = NULL,
  quiet = FALSE,
  max_mem_gb = max(4, spod_available_ram() - 4),
  max_n_cpu = parallelly::availableCores() - 1,
  temp_path = spod_get_temp_dir()
)
```

Arguments

`data_path`	a path to the `DuckDB` database file with '.duckdb' extension, or a path to the folder with `parquet` files. Eigher one should have been created with the `spod_convert()` function.
`target_table_name`	Default is `NULL`. When connecting to a folder of `parquet` files, this argument is ignored. When connecting to a `DuckDB` database, a `character` vector of length 1 with the table name to open from the database file. If not specified, it will be guessed from the `data_path` argument and from table names that are available in the database. If you have not manually interfered with the database, this should be guessed automatically and you do not need to specify it.
`quiet`	A `logical` value indicating whether to suppress messages. Default is `FALSE`.
`max_mem_gb`	The maximum memory to use in GB. A conservative default is 3 GB, which should be enough for resaving the data to DuckDB form a folder of CSV.gz files while being small enough to fit in memory of most even old computers. For data analysis using the already converted data (in DuckDB or Parquet format) or with the raw CSV.gz data, it is recommended to increase it according to available resources.
`max_n_cpu`	The maximum number of threads to use. Defaults to the number of available cores minus 1.
`temp_path`	The path to the temp folder for DuckDB for intermediate spilling in case the set memory limit and/or physical memory of the computer is too low to perform the query. By default this is set to the `temp` directory in the data folder defined by SPANISH_OD_DATA_DIR environment variable. Otherwise, for queries on folders of CSV files or parquet files, the temporary path would be set to the current R working directory, which probably is undesirable, as the current

working directory can be on a slow storage, or storage that may have limited space, compared to the data folder.

Value

a DuckDB table connection object.

`spod_convert` *Convert data from plain text to duckdb or parquet format*

Description

Converts data for faster analysis into either `DuckDB` file or into `parquet` files in a hive-style directory structure. Running analysis on these files is sometimes 100x times faster than working with raw CSV files, espetially when these are in gzip archives. To connect to converted data, please use `mydata <- spod_connect()` passing the path to where the data was saved. The connected `mydata` can be analysed using `dplyr` functions such as `select()`, `filter()`, `mutate()`, `group_by()`, `summarise()`, etc. In the end of any sequence of commands you will need to add `collect()` to execute the whole chain of data manipulations and load the results into memory in an R `data.frame/tibble`. For more in-depth usage of such data, please refer to DuckDB documentation and examples at https://duckdb.org/docs/api/r#dbplyr . Some more useful examples can be found here https://arrow-user2022.netlify.app/data-wrangling#combining-arrow-with-duckdb . You may also use `arrow` package to work with parquet files https://arrow.apache.org/docs/r/.

Usage

```
spod_convert(
  type = c("od", "origin-destination", "os", "overnight_stays", "nt", "number_of_trips"),
  zones = c("districts", "dist", "distr", "distritos", "municipalities", "muni",
    "municip", "municipios"),
  dates = NULL,
  save_format = "duckdb",
  save_path = NULL,
  overwrite = FALSE,
  data_dir = spod_get_data_dir(),
  quiet = FALSE,
  max_mem_gb = max(4, spod_available_ram() - 4),
  max_n_cpu = parallelly::availableCores() - 1,
  max_download_size_gb = 1
)
```

Arguments

`type`	The type of data to download. Can be `"origin-destination"` (or ust `"od"`), or `"number_of_trips"` (or just `"nt"`) for v1 data. For v2 data `"overnight_stays"` (or just `"os"`) is also available. More data types to be supported in the future.

See codebooks for v1 and v2 data in vignettes with `spod_codebook(1)` and `spod_codebook(2)` (spod_codebook).

`zones` The zones for which to download the data. Can be `"districts"` (or `"dist"`, `"distr"`, or the original Spanish `"distritos"`) or `"municipalities"` (or `"muni"`, `"municip"`, or the original Spanish `"municipios"`) for both data versions. Additionaly, these can be `"large_urban_areas"` (or `"lua"`, or the original Spanish `"grandes_areas_urbanas"`, or `"gau"`) for v2 data (2022 onwards).

`dates` A `character` or `Date` vector of dates to process. Kindly keep in mind that v1 and v2 data follow different data collection methodologies and may not be directly comparable. Therefore, do not try to request data from both versions for the same date range. If you need to compare data from both versions, please refer to the respective codebooks and methodology documents. The v1 data covers the period from 2020-02-14 to 2021-05-09, and the v2 data covers the period from 2022-01-01 to the present until further notice. The true dates range is checked against the available data for each version on every function run.

The possible values can be any of the following:

- For the `spod_get()` and `spod_convert()` functions, the `dates` can be set to "cached_v1" or "cached_v2" to request data from cached (already previously downloaded) v1 (2020-2021) or v2 (2022 onwards) data. In this case, the function will identify and use all data files that have been downloaded and cached locally, (e.g. using an explicit run of `spod_download()`, or any data requests made using the `spod_get()` or `spod_convert()` functions).
- A single date in ISO (YYYY-MM-DD) or YYYYMMDD format. `character` or `Date` object.
- A vector of dates in ISO (YYYY-MM-DD) or YYYYMMDD format. `character` or `Date` object. Can be any non-consecutive sequence of dates.
- A date range
  - eigher a `character` or `Date` object of length 2 with clearly named elements `start` and `end` in ISO (YYYY-MM-DD) or YYYYMMDD format. E.g. `c(start = "2020-02-15", end = "2020-02-17")`;
  - or a `character` object of the form `YYYY-MM-DD_YYYY-MM-DD` or `YYYYMMDD_YYYYMMDD`. For example, `2020-02-15_2020-02-17` or `20200215_20200217`.
- A regular expression to match dates in the format `YYYYMMDD`. `character` object. For example, `^202002` will match all dates in February 2020.

`save_format` A `character` vector of length 1 with values "duckdb" or "parquet". Defaults to "duckdb". If `NULL` automatically inferred from the `save_path` argument. If only `save_format` is provided, `save_path` will be set to the default location set in `SPANISH_OD_DATA_DIR` environment variable using `Sys.setenv(SPANISH_OD_DATA_DIR = 'path/to/your/cache/dir')`). So for v1 data that path would be `<data_dir>/clean_data/v1/tabu` or `<data_dir>/clean_data/v1/tabular/parquet/`.

You can also set `save_path`. If it ends with ".duckdb", will save to `DuckDB` database format, if `save_path` does not end with ".duckdb", will save to `parquet` format and will treat the `save_path` as a path to a folder, not a file, will create necessary hive-style subdirectories in that folder. Hive style looks like `year=2020/month=2/day=14` and inside each such directory there will be a `data_0.parquet` file that contains the data for that day.

save_path A character vector of length 1. The full (not relative) path to a DuckDB database file or parquet folder.

- If save_path ends with .duckdb, it will be saved as a DuckDB database file. The format argument will be automatically set to save_format='duckdb'.
- If save_path ends with a folder name (e.g. /data_dir/clean_data/v1/tabular/parquet/od_dis for origin-destination data for district level), the data will be saved as a collection of parquet files in a hive-style directory structure. So the subfolders of od_distr will be year=2020/month=2/day=14 and inside each of these folders a single parquet file will be placed containing the data for that day.
- If NULL, uses the default location in data_dir (set by the SPANISH_OD_DATA_DIR environment variable using Sys.setenv(SPANISH_OD_DATA_DIR = 'path/to/your/cache/dir')). Therefore, the default relative path for DuckDB is <data_dir>/clean_data/v1/tabular/duckdb/<t and for parquet files is <data_dir>/clean_data/v1/tabular/parquet/<type>_<zones>/, where type is the type of data (e.g. 'od', 'os', 'nt', that correspoind to 'origin-destination', 'overnight-stays', 'number-of-trips', etc.) and zones is the name of the geographic zones (e.g. 'distr', 'muni', etc.). See the details below in the function arguments description.

overwrite A logical or a character vector of length 1. If TRUE, overwrites existing DuckDBorparquetfil For parquet files can also be set to 'update', so that only parquet files are only created for the dates that have not yet been converted.

data_dir The directory where the data is stored. Defaults to the value returned by spod_get_data_dir() which returns the value of the environment variable SPANISH_OD_DATA_DIR or a temporary directory if the variable is not set.

quiet A logical value indicating whether to suppress messages. Default is FALSE.

max_mem_gb The maximum memory to use in GB. A conservative default is 3 GB, which should be enough for resaving the data to DuckDB form a folder of CSV.gz files while being small enough to fit in memory of most even old computers. For data analysis using the already converted data (in DuckDB or Parquet format) or with the raw CSV.gz data, it is recommended to increase it according to available resources.

max_n_cpu The maximum number of threads to use. Defaults to the number of available cores minus 1.

max_download_size_gb
The maximum download size in gigabytes. Defaults to 1.

Value

Path to saved DuckDB file.

---

spod_disconnect *Safely disconnect from data and free memory*

Description

This function is to ensure that DuckDB connections to CSV.gz files (created via spod_get()), as well as to DuckDB files or folders of parquet files (created via spod_convert()) are closed properly to prevent conflicting connections. Essentially this is just a wrapper around DBI::dbDisconnect() that reaches out into the .srccon object of the tbl_duckdb_connection connection object that is returned to the user via spod_get() and spod_connect(). After disonnecting the database, it also frees up memory by running gc().

Usage

```
spod_disconnect(tbl_con, free_mem = TRUE)
```

Arguments

tbl_con	A tbl_duckdb_connection connection object that you get from either spod_get() or spod_connect().
free_mem	A logical. Whether to free up memory by running gc(). Defaults to TRUE.

Examples

```
## Not run:
od_distr <- spod_get("od", zones = "distr", dates <- c("2020-01-01", "2020-01-02"))
spod_disconnect(od_distr)

## End(Not run)
```

---

spod_download *Download the data files of specified type, zones, and dates*

Description

This function downloads the data files of the specified type, zones, dates and data version.

Usage

```
spod_download(
  type = c("od", "origin-destination", "os", "overnight_stays", "nt", "number_of_trips"),
  zones = c("districts", "dist", "distr", "distritos", "municipalities", "muni",
   "municip", "municipios", "lua", "large_urban_areas", "gau", "grandes_areas_urbanas"),
  dates = NULL,
  max_download_size_gb = 1,
  data_dir = spod_get_data_dir(),
  quiet = FALSE,
  return_local_file_paths = FALSE
)
```

Arguments

type	The type of data to download. Can be "origin-destination" (or ust "od"), or "number_of_trips" (or just "nt") for v1 data. For v2 data "overnight_stays" (or just "os") is also available. More data types to be supported in the future. See codebooks for v1 and v2 data in vignettes with spod_codebook(1) and spod_codebook(2) (spod_codebook).
zones	The zones for which to download the data. Can be "districts" (or "dist", "distr", or the original Spanish "distritos") or "municipalities" (or "muni", "municip", or the original Spanish "municipios") for both data versions. Additionaly, these can be "large_urban_areas" (or "lua", or the original Spanish "grandes_areas_urbanas", or "gau") for v2 data (2022 onwards).
dates	A character or Date vector of dates to process. Kindly keep in mind that v1 and v2 data follow different data collection methodologies and may not be directly comparable. Therefore, do not try to request data from both versions for the same date range. If you need to compare data from both versions, please refer to the respective codebooks and methodology documents. The v1 data covers the period from 2020-02-14 to 2021-05-09, and the v2 data covers the period from 2022-01-01 to the present until further notice. The true dates range is checked against the available data for each version on every function run. The possible values can be any of the following: • For the spod_get() and spod_convert() functions, the dates can be set to "cached_v1" or "cached_v2" to request data from cached (already previously downloaded) v1 (2020-2021) or v2 (2022 onwards) data. In this case, the function will identify and use all data files that have been downloaded and cached locally, (e.g. using an explicit run of spod_download(), or any data requests made using the spod_get() or spod_convert() functions). • A single date in ISO (YYYY-MM-DD) or YYYYMMDD format. character or Date object. • A vector of dates in ISO (YYYY-MM-DD) or YYYYMMDD format. character or Date object. Can be any non-consecutive sequence of dates. • A date range – eigher a character or Date object of length 2 with clearly named elements start and end in ISO (YYYY-MM-DD) or YYYYMMDD format. E.g. c(start = "2020-02-15", end = "2020-02-17"); – or a character object of the form YYYY-MM-DD_YYYY-MM-DD or YYYYMMDD_YYYYMMDD. For example, 2020-02-15_2020-02-17 or 20200215_20200217. • A regular expression to match dates in the format YYYYMMDD. character object. For example, ^202002 will match all dates in February 2020.
max_download_size_gb	The maximum download size in gigabytes. Defaults to 1.
data_dir	The directory where the data is stored. Defaults to the value returned by spod_get_data_dir() which returns the value of the environment variable SPANISH_OD_DATA_DIR or a temporary directory if the variable is not set.
quiet	A logical value indicating whether to suppress messages. Default is FALSE.

return_local_file_paths
: Logical. If TRUE, the function returns a character vector of the paths to the downloaded files. If FALSE, the function returns NULL.

Value

Nothing. If return_local_file_paths = TRUE, a character vector of the paths to the downloaded files.

Examples

```
## Not run:
# Download the origin-destination on district level for the a date range in March 2020
spod_download(
  type = "od", zones = "districts",
  dates = c(start = "2020-03-20", end = "2020-03-24")
)

# Download the origin-destination on district level for select dates in 2020 and 2021
spod_download(
  type = "od", zones = "dist",
  dates = c("2020-03-20", "2020-03-24", "2021-03-20", "2021-03-24")
)

# Download the origin-destination on municipality level using regex for a date range in March 2020
# (the regex will capture the dates 2020-03-20 to 2020-03-24)
spod_download(
  type = "od", zones = "municip",
  dates = "2020032[0-4]"
)

## End(Not run)
```

---

spod_get *Get tabular data*

Description

This function creates a DuckDB lazy table connection object from the specified type and zones. It checks for missing data and downloads it if necessary. The connnection is made to the raw CSV files in gzip archives, so analysing the data through this connection may be slow if you select more than a few days. You can manipulate this object using {dplyr} functions such as select, filter, mutate, group_by, summarise, etc. In the end of any sequence of commands you will need to add collect to execute the whole chain of data manipulations and load the results into memory in an R data.frame/tibble. See codebooks for v1 and v2 data in vignettes with spod_codebook(1) and spod_codebook(2) (spod_codebook).

If you want to analyse longer periods of time (especiially several months or even the whole data over several years), consider using the spod_convert and then spod_connect.

Usage

```
spod_get(
  type = c("od", "origin-destination", "os", "overnight_stays", "nt", "number_of_trips"),
  zones = c("districts", "dist", "distr", "distritos", "municipalities", "muni",
   "municip", "municipios", "lua", "large_urban_areas", "gau", "grandes_areas_urbanas"),
  dates = NULL,
  data_dir = spod_get_data_dir(),
  quiet = FALSE,
  max_mem_gb = max(4, spod_available_ram() - 4),
  max_n_cpu = parallelly::availableCores() - 1,
  max_download_size_gb = 1,
  duckdb_target = ":memory:",
  temp_path = spod_get_temp_dir()
)
```

Arguments

type	The type of data to download. Can be "origin-destination" (or ust "od"), or "number_of_trips" (or just "nt") for v1 data. For v2 data "overnight_stays" (or just "os") is also available. More data types to be supported in the future. See codebooks for v1 and v2 data in vignettes with spod_codebook(1) and spod_codebook(2) (spod_codebook).
zones	The zones for which to download the data. Can be "districts" (or "dist", "distr", or the original Spanish "distritos") or "municipalities" (or "muni", "municip", or the original Spanish "municipios") for both data versions. Additionaly, these can be "large_urban_areas" (or "lua", or the original Spanish "grandes_areas_urbanas", or "gau") for v2 data (2022 onwards).
dates	A character or Date vector of dates to process. Kindly keep in mind that v1 and v2 data follow different data collection methodologies and may not be directly comparable. Therefore, do not try to request data from both versions for the same date range. If you need to compare data from both versions, please refer to the respective codebooks and methodology documents. The v1 data covers the period from 2020-02-14 to 2021-05-09, and the v2 data covers the period from 2022-01-01 to the present until further notice. The true dates range is checked against the available data for each version on every function run. The possible values can be any of the following: • For the spod_get() and spod_convert() functions, the dates can be set to "cached_v1" or "cached_v2" to request data from cached (already previously downloaded) v1 (2020-2021) or v2 (2022 onwards) data. In this case, the function will identify and use all data files that have been downloaded and cached locally, (e.g. using an explicit run of spod_download(), or any data requests made using the spod_get() or spod_convert() functions). • A single date in ISO (YYYY-MM-DD) or YYYYMMDD format. character or Date object. • A vector of dates in ISO (YYYY-MM-DD) or YYYYMMDD format. character or Date object. Can be any non-consecutive sequence of dates. • A date range

  - eigher a character or Date object of length 2 with clearly named elements start and end in ISO (YYYY-MM-DD) or YYYYMMDD format. E.g. c(start = "2020-02-15", end = "2020-02-17");
  - or a character object of the form YYYY-MM-DD_YYYY-MM-DD or YYYYMMDD_YYYYMMDD. For example, 2020-02-15_2020-02-17 or 20200215_20200217.
- A regular expression to match dates in the format YYYYMMDD. character object. For example, ^202002 will match all dates in February 2020.

data_dir
: The directory where the data is stored. Defaults to the value returned by spod_get_data_dir() which returns the value of the environment variable SPANISH_OD_DATA_DIR or a temporary directory if the variable is not set.

quiet
: A logical value indicating whether to suppress messages. Default is FALSE.

max_mem_gb
: The maximum memory to use in GB. A conservative default is 3 GB, which should be enough for resaving the data to DuckDB form a folder of CSV.gz files while being small enough to fit in memory of most even old computers. For data analysis using the already converted data (in DuckDB or Parquet format) or with the raw CSV.gz data, it is recommended to increase it according to available resources.

max_n_cpu
: The maximum number of threads to use. Defaults to the number of available cores minus 1.

max_download_size_gb
: The maximum download size in gigabytes. Defaults to 1.

duckdb_target
: (Optional) The path to the duckdb file to save the data to, if a convertation from CSV is reuqested by the spod_convert function. If not specified, it will be set to ":memory:" and the data will be stored in memory.

temp_path
: The path to the temp folder for DuckDB for intermediate spilling in case the set memory limit and/or physical memory of the computer is too low to perform the query. By default this is set to the temp directory in the data folder defined by SPANISH_OD_DATA_DIR environment variable. Otherwise, for queries on folders of CSV files or parquet files, the temporary path would be set to the current R working directory, which probably is undesirable, as the current working directory can be on a slow storage, or storage that may have limited space, compared to the data folder.

Value

A DuckDB lazy table connection object of class tbl_duckdb_connection.

Examples

```
## Not run:

# create a connection to the v1 data
Sys.setenv(SPANISH_OD_DATA_DIR = "~/path/to/your/cache/dir")
dates <- c("2020-02-14", "2020-03-14", "2021-02-14", "2021-02-14", "2021-02-15")
od_dist <- spod_get(type = "od", zones = "distr", dates = dates)

# od dist is a table view filtered to the specified dates
```

```
# access the source connection with all dates
# list tables
DBI::dbListTables(od_dist$src$con)

## End(Not run)
```

`spod_get_valid_dates` *Get valid dates for the specified data version*

Description

Get valid dates for the specified data version

Usage

```
spod_get_valid_dates(ver = NULL)
```

Arguments

`ver`	Integer. Can be 1 or 2. The version of the data to use. v1 spans 2020-2021, v2 covers 2022 and onwards.

Value

A vector of type `Date` with all possible valid dates for the specified data version (v1 for 2020-2021 and v2 for 2020 onwards).

`spod_get_zones` *Get zones*

Description

Get spatial zones for the specified data version. Supports both v1 (2020-2021) and v2 (2022 onwards) data.

Usage

```
spod_get_zones(
  zones = c("districts", "dist", "distr", "distritos", "municipalities", "muni",
   "municip", "municipios", "lua", "large_urban_areas", "gau", "grandes_areas_urbanas"),
  ver = NULL,
  data_dir = spod_get_data_dir(),
  quiet = FALSE
)
```

Arguments

zones	The zones for which to download the data. Can be "districts" (or "dist", "distr", or the original Spanish "distritos") or "municipalities" (or "muni", "municip", or the original Spanish "municipios") for both data versions. Additionaly, these can be "large_urban_areas" (or "lua", or the original Spanish "grandes_areas_urbanas", or "gau") for v2 data (2022 onwards).
ver	Integer. Can be 1 or 2. The version of the data to use. v1 spans 2020-2021, v2 covers 2022 and onwards.
data_dir	The directory where the data is stored. Defaults to the value returned by spod_get_data_dir() which returns the value of the environment variable SPANISH_OD_DATA_DIR or a temporary directory if the variable is not set.
quiet	A logical value indicating whether to suppress messages. Default is FALSE.

Value

An sf object (Simple Feature collection).

The columns for v1 (2020-2021) data include:

id A character vector containing the unique identifier for each district, assigned by the data provider. This id matches the id_origin, id_destination, and id in district-level origin-destination and number of trips data.

census_districts A string with semicolon-separated identifiers of census districts classified by the Spanish Statistical Office (INE) that are spatially bound within the polygons for each id.

municipalities_mitma A string with semicolon-separated municipality identifiers (as assigned by the data provider) corresponding to each district id.

municipalities A string with semicolon-separated municipality identifiers classified by the Spanish Statistical Office (INE) corresponding to each id.

district_names_in_v2/municipality_names_in_v2 A string with semicolon-separated district names (from the v2 version of this data) corresponding to each district id in v1.

district_ids_in_v2/municipality_ids_in_v2 A string with semicolon-separated district identifiers (from the v2 version of this data) corresponding to each district id in v1.

geometry A MULTIPOLYGON column containing the spatial geometry of each district, stored as an sf object. The geometry is projected in the ETRS89 / UTM zone 30N coordinate reference system (CRS), with XY dimensions.

The columns for v2 (2022 onwards) data include:

id A character vector containing the unique identifier for each zone, assigned by the data provider.

name A character vector with the name of each district.

population A numeric vector representing the population of each district (as of 2022).

census_sections A string with semicolon-separated identifiers of census sections corresponding to each district.

census_districts A string with semicolon-separated identifiers of census districts as classified by the Spanish Statistical Office (INE) corresponding to each district.

municipalities A string with semicolon-separated identifiers of municipalities classified by the Spanish Statistical Office (INE) corresponding to each district.

municipalities_mitma A string with semicolon-separated identifiers of municipalities, as assigned by the data provider, that correspond to each district.

luas_mitma A string with semicolon-separated identifiers of LUAs (Local Urban Areas) from the provider, associated with each district.

district_ids_in_v1/municipality_ids_in_v1 A string with semicolon-separated district identifiers from v1 data corresponding to each district in v2. If no match exists, it is marked as `NA`.

geometry A `MULTIPOLYGON` column containing the spatial geometry of each district, stored as an sf object. The geometry is projected in the ETRS89 / UTM zone 30N coordinate reference system (CRS), with XY dimensions.

Index