Package 'TestGenerator'

May 26, 2026

Type Package

Title Integration Unit Tests for Pharmacoepidemiological Studies

Version 0.8.0

Maintainer Cesar Barboza <c.barboza@darwin-eu.org>

Description An R interface to load testing data in the 'OMOP' Common Data Model ('CDM'). An input file, csv or xlsx, can be converted to a 'CDMConnector' object. This object can be used to execute and test studies that use the 'CDM' <https://www.ohdsi.org/data-standardization/>.

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports jsonlite, readxl, readr, CDMConnector (>= 2.5.1), DBI, dplyr, checkmate, glue, duckdb, cli, rlang, withr, tibble, testthat, openxlsx, omopgenerics, RPostgres, fs, stringr, usethis

Suggests knitr, rmarkdown, curl, httr, odbc, CohortConstructor

VignetteBuilder knitr

Config/testthat/edition 3

URL https://github.com/darwin-eu/TestGenerator, https://darwin-eu.github.io/TestGenerator/

BugReports https://github.com/darwin-eu/TestGenerator/issues

NeedsCompilation no

Author Cesar Barboza [aut, cre] (ORCID: <https://orcid.org/0009-0002-4453-3071>), Ioanna Nika [aut], Ger Inberg [aut] (ORCID: <https://orcid.org/0000-0001-8993-8748>), Adam Black [aut] (ORCID: <https://orcid.org/0000-0001-5576-8701>)

Repository CRAN

Date/Publication 2026-05-26 13:20:21 UTC

Contents

`checkRemoteFileAvailable` *Check if a given remote file is available for download*

Description

Check if a given remote file is available for download

Usage

```
checkRemoteFileAvailable(remoteFile)
```

Arguments

`remoteFile`	a remote resource

Value

NULL if the remote resource is not available, other "success"

`checkTablesColumns` *Check if the given tables and columns are valid and return the loaded data.*

Description

Check if the given tables and columns are valid and return the loaded data.

Usage

```
checkTablesColumns(cdmVersion, filePath, extraTable = FALSE)
```

Arguments

cdmVersion	cdm version
filePath	Path to the test patient data in xlsx format
extraTable	if extra tables are provided or not, default FALSE

Value

a named list containing the loaded data

---

cleanupTestCdm *Clean up a test CDM on a remote database*

Description

Drops the schema containing the test CDM and disconnects from the database.

Usage

```
cleanupTestCdm(cdm)
```

Arguments

cdm	A CDM reference object created by patientsCDM() with a remote database backend.

Value

Invisibly returns NULL.

---

downloadTestData *Download Test Data Files*

Description

Download Test Data Files

Usage

```
downloadTestData(
  datasetName = "mimicIV",
  cdmVersion = "5.3",
  pathToData = Sys.getenv("STUDY_DATASETS"),
  overwrite = FALSE
)
```

Arguments

datasetName	The data set name as found on https://github.com/darwin-eu/EunomiaDatasets. The data set name corresponds to the folder with the data set ZIP files
cdmVersion	The OMOP CDM version. This version will appear in the suffix of the data file, for example: synpuf_5.3.zip. Default: '5.3'
pathToData	The path where the Eunomia data is stored on the file system., By default the value of the environment variable "EUNOMIA_DATA_FOLDER" is used.
overwrite	Control whether the existing archive file will be overwritten should it already exist.

Value

Invisibly returns the destination if the download was successful.

Examples

```
downloadTestData(pathToData = tempdir())
```

---

generateTestTables *Generates an Excel file with sheets that correspond to an OMOP-CDM tables.*

Description

Generates an Excel file with sheets that correspond to an OMOP-CDM tables.

Usage

```
generateTestTables(
  tableNames = c("person", "observation_period", "visit_occurrence", "visit_detail",
    "condition_occurrence", "drug_exposure", "procedure_occurrence", "measurement",
    "observation", "death", "drug_era", "condition_era", "dose_era", "location",
    "care_site", "provider"),
  cdmVersion,
  outputFolder,
  filename = paste0("test_cdm_", cdmVersion)
)
```

Arguments

tableNames	A list specifying the table names to include in the Excel file.
cdmVersion	The CDM version to use for creating the requested tables (either 5.3 or 5.4).
outputFolder	The folder where the Excel file will be saved.
filename	The name of the Excel file. It does not include the extension .xlsx.

Value

An Excel file with the tables requested.

`graphCohort` *Deprecated cohort timeline visualisation*

Description

`graphCohort()` is deprecated and will be removed in a future release.

Usage

```
graphCohort(subject_id, cohorts = list())
```

Arguments

`subject_id`	Deprecated.
`cohorts`	Deprecated.

Value

Invisibly returns `NULL`.

`patientsCDM` *Pushes test population into a blank CDM.*

Description

Pushes test population into a blank CDM.

Usage

```
patientsCDM(
  pathJson = NULL,
  testName = NULL,
  cdmVersion = "5.4",
  cdmName = NULL,
  dbms = "duckdb",
  writeSchema = NULL
)
```

Arguments

pathJson	Directory where the sample populations in json are located. If NULL, gets the default inst/testCases directory.
testName	Name of the sample population JSON file. If NULL it will push the first sample population in the testCases directory.
cdmVersion	cdm version, default "5.4".
cdmName	Name of the cdm, default NULL.
dbms	Database management system to use. One of "duckdb", "databricks", "sqlserver" or "postgresql". Default is "duckdb" which creates a local DuckDB CDM. For remote databases the function creates the CDM locally, trims the vocabulary, uploads to a new schema on the remote database, and returns the remote CDM reference. Remote databases require environment variables to be set. Call `usethis::edit_r_environ()` to set them.
writeSchema	Optional schema name to use on the remote database. If NULL (default), a unique schema is created automatically. Only used when dbms is not "duckdb".

Value

A CDM reference object with a sample population.

Examples

```
filePath <- system.file("extdata", "testPatientsRSV.xlsx", package = "TestGenerator")
TestGenerator::readPatients(filePath = filePath, outputPath = tempdir())
cdm <- TestGenerator::patientsCDM(pathJson = tempdir(), testName = "test")
duckdb::duckdb_shutdown(duckdb::duckdb())
```

---

readPatients *Converts a sample of patients into Unit Testing Definition JSON file.*

Description

Converts a sample of patients into Unit Testing Definition JSON file.

Usage

```
readPatients(
  filePath = NULL,
  testName = "test",
  outputPath = NULL,
  cdmVersion = "5.4",
  extraTable = FALSE
)
```

Arguments

filePath	Path to the test patient data in Excel format. The Excel has sheets that represent tables from the OMOP-CDM, e.g. person, drug_exposure, condition_ocurrence, etc.
testName	A name of the test population in character.
outputPath	Path of the output file, if NULL, a folder will be created in the project folder inst/testCases.
cdmVersion	cdm version, default "5.4".
extraTable	Name of non-standard tables to be included in the test CDM.

Value

A JSON file with sample patients inside the project directory.

Examples

```
filePath <- system.file("extdata", "testPatientsRSV.xlsx", package = "TestGenerator")
readPatients(filePath = filePath, outputPath = tempdir())
```

---

readPatients.csv *Converts a sample of patients in CSV format into a Unit Testing Definition JSON file.*

Description

Converts a sample of patients in CSV format into a Unit Testing Definition JSON file.

Usage

```
readPatients.csv(
  filePath = NULL,
  testName = "test",
  outputPath = NULL,
  cdmVersion = "5.4",
  reduceLargeIds = FALSE
)
```

Arguments

filePath	Path to the test patient data in CSV format. Multiple CSV files representing tables tables from the OMOP-CDM must be provided, e.g. person.csv, drug_exposure.csv, condition_ocurrence.csv, etc.
testName	Name for the test population file in character.
outputPath	Path of the output file, if NULL, a folder will be created in the project folder inst/testCases.

cdmVersion	cdm version, default "5.4".
reduceLargeIds	Reduces the length of very long ids generally in int64 format, such as those found in the MIMIC-IV database.

Value

A JSON file with sample patients inside the project directory.

Examples

```
filePath <- system.file("extdata", "mimic_sample", package = "TestGenerator")
readPatients.csv(
  filePath = filePath,
  outputPath = tempdir(),
  cdmVersion = "5.3"
  )
```

---

readPatients.xl *Converts a sample of patients in XLSX format into Unit Testing Definition JSON file.*

Description

Converts a sample of patients in XLSX format into Unit Testing Definition JSON file.

Usage

```
readPatients.xl(
  filePath = NULL,
  testName = "test",
  outputPath = NULL,
  cdmVersion = "5.4",
  extraTable = FALSE
)
```

Arguments

filePath	Path to the test patient data in Excel format. The Excel has sheets that represent tables from the OMOP-CDM, e.g. person, drug_exposure, condition_ocurrence, etc.
testName	A name of the test population in character.
outputPath	Path to write the test JSON files. If NULL, the files will be written at the project's testthat folder, i.e. tests/testthat/testCases.
cdmVersion	cdm version, default "5.4".
extraTable	TRUE or FALSE. If TRUE, non-standard tables will be included in the test CDM.

Value

A directory with the test JSON files with sample patients inside the project directory.

Examples

```
filePath <- system.file("extdata", "testPatientsRSV.xlsx", package = "TestGenerator")
readPatients.xl(filePath = filePath, outputPath = tempdir())
```

---

useGithubAction *Create GitHub Action workflows for specific DBMS*

Description

Copies GitHub Action workflow templates from the package's `inst/workflows` directory to the `.github/workflows` directory of the current project. The copied workflows can be used to run backend-specific tests for PostgreSQL, SQL Server, and Databricks.

Usage

```
useGithubAction(
  dbms_type = c("postgresql", "sqlserver", "databricks"),
  overwrite = FALSE
)
```

Arguments

`dbms_type`	A character vector of supported DBMS types to process. Supported values are `"postgresql"`, `"sqlserver"`, and `"databricks"`.
`overwrite`	A logical value indicating whether to overwrite existing workflow files in the `.github/workflows` directory. Defaults to `FALSE`.

Value

Invisibly returns `TRUE` when the selected workflow files are copied successfully.

Examples

```
## Not run:
useGithubAction(dbms_type = "postgresql")
useGithubAction(
  dbms_type = c("postgresql", "sqlserver", "databricks"),
  overwrite = TRUE
)

## End(Not run)
```

Index