Package 'CohortDiagnostics'

February 1, 2022

Type Package

Title Diagnostics for OHDSI Cohorts

Version 3.0.0

Date 2021-11-13

Maintainer Gowtham Rao <rao@ohdsi.org>

Description
Diagnostics for cohorts that use the OMOP Common Data Model and the OHDSI tools.

Depends DatabaseConnector (¿= 4.0.0),
FeatureExtraction (¿= 3.1.1),
R (¿= 4.0.0)

Imports Andromeda,
checkmate,
clock,
digest,
dplyr (¿= 1.0.0),
methods,
ParallelLogger (¿= 2.0.0),
readr (¿= 2.0.1),
RJSONIO,
rlang,
ROhdsiWebApi (¿= 1.2.0),
SqlRender (¿= 1.7.0),
stringr,
tidyr (¿= 1.0.0),
CohortGenerator (¿= 0.2.0)

Suggests CirceR,
DT,
Eunomia,
ggiraph,
ggplot2,
htmltools,
knitr,
lubridate,
pool,
plotly,
purrr,
RColorBrewer,
remotes,

rmarkdown,
RSQLite (¿= 2.2.1),
scales,
shiny,
shinydashboard,
shinyWidgets,
testthat,
withr,
zip

Remotes ohdsi/Eunomia,
ohdsi/FeatureExtraction,
ohdsi/ROhdsiWebApi,
ohdsi/CirceR,
ohdsi/CohortGenerator

License Apache License

VignetteBuilder knitr

URL https://ohdsi.github.io/CohortDiagnostics, https://github.com/OHDSI/CohortDiagnostics

BugReports https://github.com/OHDSI/CohortDiagnostics/issues

RoxygenNote 7.1.2

Encoding UTF-8

Language en-US

R topics documented:

checkInputFileEncoding 2
createMergedResultsFile 3
createResultsDataModel 3
executeDiagnostics 4
getCohortCounts 7
getResultsDataModelSpecifications 8
launchCohortExplorer 8
launchDiagnosticsExplorer 9
loadCohortsFromPackage 10
runCohortDiagnostics 11
uploadResults 15

`checkInputFileEncoding` *Check character encoding of input file*

Description

For its input files, CohortDiagnostics only accepts UTF-8 or ASCII character encoding. This function can be used to check whether a file meets these criteria.

Usage

```
checkInputFileEncoding(fileName)
```

Arguments

fileName	The path to the file to check

Value

Throws an error if the input file does not have the correct encoding.

---

createMergedResultsFile

Merge Shiny diagnostics files into sqlite database

Description

This function combines diagnostics results from one or more databases into a single file. The result is an sqlite database that can be used as input for the Diagnostics Explorer Shiny app.

It also checks whether the results conform to the results data model specifications.

Usage

```
createMergedResultsFile(
  dataFolder,
  sqliteDbPath = "MergedCohortDiagnosticsData.sqlite",
  overwrite = FALSE
)
```

Arguments

dataFolder	folder where the exported zip files for the diagnostics are stored. Use the runCohortDiagnostics function to generate these zip files. Zip files containing results from multiple databases may be placed in the same folder.
sqliteDbPath	Output path where sqlite database is placed
overwrite	(Optional) overwrite existing sqlite lite db if it exists.

---

createResultsDataModel

Create the results data model tables on a database server.

Description

Create the results data model tables on a database server.

Usage

```
createResultsDataModel(connection = NULL, connectionDetails = NULL, schema)
```

Arguments

connection	An object of type connection as created using the connect function in the DatabaseConnector package. Can be left NULL if connectionDetails is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.
schema	The schema on the postgres server where the tables will be created.

Details

Only PostgreSQL servers are supported.

---

executeDiagnostics *Execute cohort diagnostics functions*

Description

Execute cohort diagnostics on a set of cohorts. Assumes that cohorts have already been instantiated in advance.

Note that none of the references passed to this function are used, they are required for results.

Usage

```
executeDiagnostics(
  cohortDefinitionSet,
  exportFolder,
  databaseId,
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  cohortDatabaseSchema,
  cohortTable = "cohort",
 cohortTableNames = CohortGenerator::getCohortTableNames(cohortTable = cohortTable),
  vocabularyDatabaseSchema = cdmDatabaseSchema,
  cohortIds = NULL,
  inclusionStatisticsFolder = NULL,
  databaseName = databaseId,
  databaseDescription = databaseId,
  cdmVersion = 5,
  runInclusionStatistics = TRUE,
  runIncludedSourceConcepts = TRUE,
  runOrphanConcepts = TRUE,
  runTimeDistributions = TRUE,
  runVisitContext = TRUE,
  runBreakdownIndexEvents = TRUE,
```

```
  runIncidenceRate = TRUE,
  runTimeSeries = FALSE,
  runCohortOverlap = TRUE,
  runCohortCharacterization = TRUE,
  covariateSettings = createDefaultCovariateSettings(),
  runTemporalCohortCharacterization = TRUE,
 temporalCovariateSettings = createTemporalCovariateSettings(useConditionOccurrence =
  TRUE, useDrugEraStart = TRUE, useProcedureOccurrence = TRUE, useMeasurement = TRUE,
  temporalStartDays = c(-365, -30, 0, 1, 31), temporalEndDays = c(-31, -1, 0, 30, 365)),
 minCellCount = 5,
 incremental = FALSE,
 incrementalFolder = file.path(exportFolder, "incremental")
)
```

Arguments

`cohortDefinitionSet`	Data.frame of cohorts must include columns cohortId, cohortName, json, sql
`exportFolder`	The folder where the output will be exported to. If this folder does not exist it will be created.
`databaseId`	A short string for identifying the database (e.g. 'Synpuf').
`connectionDetails`	An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.
`connection`	An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
`cdmDatabaseSchema`	Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.
`tempEmulationSchema`	Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.
`cohortDatabaseSchema`	Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
`cohortTable`	Name of the cohort table.
`cohortTableNames`	Cohort Table names used by CohortGenerator package
`vocabularyDatabaseSchema`	Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.
`cohortIds`	Optionally, provide a subset of cohort IDs to restrict the diagnostics to.

`inclusionStatisticsFolder`
: The folder where the inclusion rule statistics are stored. Can be left NULL if `runInclusionStatistics = FALSE`.

`databaseName`
: The full name of the database. If NULL, defaults to databaseId.

`databaseDescription`
: A short description (several sentences) of the database. If NULL, defaults to databaseId.

`cdmVersion`
: The version of the OMOP CDM. Default 5. (Note: only 5 is supported.)

`runInclusionStatistics`
: Generate and export statistic on the cohort inclusion rules?

`runIncludedSourceConcepts`
: Generate and export the source concepts included in the cohorts?

`runOrphanConcepts`
: Generate and export potential orphan concepts?

`runTimeDistributions`
: Generate and export cohort time distributions?

`runVisitContext`
: Generate and export index-date visit context?

`runBreakdownIndexEvents`
: Generate and export the breakdown of index events?

`runIncidenceRate`
: Generate and export the cohort incidence rates?

`runTimeSeries`
: Generate and export the cohort prevalence rates?

`runCohortOverlap`
: Generate and export the cohort overlap? Overlaps are checked within cohortIds that have the same phenotype ID sourced from the CohortSetReference or cohortToCreateFile.

`runCohortCharacterization`
: Generate and export the cohort characterization? Only records with values greater than 0.0001 are returned.

`covariateSettings`
: Either an object of type `covariateSettings` as created using one of the createCovariateSettings function in the FeatureExtraction package, or a list of such objects.

`runTemporalCohortCharacterization`
: Generate and export the temporal cohort characterization? Only records with values greater than 0.001 are returned.

`temporalCovariateSettings`
: Either an object of type `covariateSettings` as created using one of the createTemporalCovariateSettings function in the FeatureExtraction package, or a list of such objects.

`minCellCount`
: The minimum cell count for fields contains person counts or fractions.

`incremental`
: Create only cohort diagnostics that haven't been created before?

`incrementalFolder`
: If `incremental = TRUE`, specify a folder where records are kept of which cohort diagnostics has been executed.

Examples

```
## Not run:
# Load cohorts (assumes that they have already been instantiated)
cohortTableNames <- CohortGenerator::getCohortTableNames(cohortTable = "cohort")
cohorts <- loadCohortsFromPackage(packageName = "MyGreatPackage")
connectionDetails <- createConnectionDetails(dbms = "postgresql",
                                             server = "ohdsi.com",
                                             port = 5432,
                                             user = "me",
                                             password = "secure")

executeDiagnostics(cohorts = cohorts,
                   exportFolder = "export",
                   cohortTableNames = cohortTableNames,
                   cohortDatabaseSchema = "results",
                   cdmDatabaseSchema = "cdm",
                   databaseId = "mySpecialCdm",
                   connectionDetails = connectionDetails)

# Use a custom set of cohorts defined in a data.frame
cohorts <- data.frame(
  cohortId = c(100),
  cohortName = c("Cohort Name"),
  logicDescription = c("My Cohort"),
  sql = c(readLines("path_to.sql")),
  json = c(readLines("path_to.json"))
)
executeDiagnostics(cohorts = cohorts,
                   exportFolder = "export",
                   cohortTable = "cohort",
                   cohortDatabaseSchema = "results",
                   cdmDatabaseSchema = "cdm",
                   databaseId = "mySpecialCdm",
                   connectionDetails = connectionDetails)

## End(Not run)
```

`getCohortCounts` *Count the cohort(s)*

Description

Computes the subject and entry count per cohort

Usage

```
getCohortCounts(
  connectionDetails = NULL,
  connection = NULL,
  cohortDatabaseSchema,
  cohortTable = "cohort",
  cohortIds = c()
)
```

Arguments

`connectionDetails`	An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package. Can be left NULL if `connection` is provided.
`connection`	An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.
`cohortDatabaseSchema`	Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
`cohortTable`	Name of the cohort table.
`cohortIds`	The cohort Id(s) used to reference the cohort in the cohort table. If left empty, all cohorts in the table will be included.

Value

A tibble with cohort counts

`getResultsDataModelSpecifications` *Get specifications for Cohort Diagnostics results data model*

Description

Get specifications for Cohort Diagnostics results data model

Usage

```
getResultsDataModelSpecifications()
```

Value

A tibble data frame object with specifications

`launchCohortExplorer` *Launch the CohortExplorer Shiny app*

Description

Launch the CohortExplorer Shiny app

Usage

```
launchCohortExplorer(
  connectionDetails,
  cdmDatabaseSchema,
  cohortDatabaseSchema,
  cohortTable,
  cohortId,
  sampleSize = 100,
  subjectIds = NULL
)
```

Arguments

connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package.
cdmDatabaseSchema	Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.
cohortDatabaseSchema	Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.
cohortTable	Name of the cohort table.
cohortId	The ID of the cohort.
sampleSize	Number of subjects to sample from the cohort. Ignored if subjectIds is specified.
subjectIds	A vector of subject IDs to view.

Details

Launches a Shiny app that allows the user to explore a cohort of interest.

---

launchDiagnosticsExplorer

Launch the Diagnostics Explorer Shiny app

Description

Launch the Diagnostics Explorer Shiny app

Usage

```
launchDiagnosticsExplorer(
  sqliteDbPath = "MergedCohortDiagnosticsData.sqlite",
  connectionDetails = NULL,
  resultsDatabaseSchema = NULL,
  vocabularyDatabaseSchema = NULL,
  vocabularyDatabaseSchemas = resultsDatabaseSchema,
```

```
  aboutText = NULL,
  runOverNetwork = FALSE,
  port = 80,
  launch.browser = FALSE
)
```

Arguments

sqliteDbPath	Path to merged sqlite file. See createMergedResultsFile to create file.
connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package, specifying how to connect to the server where the CohortDiagnostics results have been uploaded using the uploadResults function.
resultsDatabaseSchema	The schema on the database server where the CohortDiagnostics results have been uploaded.
vocabularyDatabaseSchema	(Deprecated) Please use vocabularyDatabaseSchemas.
vocabularyDatabaseSchemas	(optional) A list of one or more schemas on the database server where the vocabulary tables are located. The default value is the value of the resultsDatabaseSchema. We can provide a list of vocabulary schema that might represent different versions of the OMOP vocabulary tables. It allows us to compare the impact of vocabulary changes on Diagnostics. Not supported with an sqlite database.
aboutText	Text (using HTML markup) that will be displayed in an About tab in the Shiny app. If not provided, no About tab will be shown.
runOverNetwork	(optional) Do you want the app to run over your network?
port	(optional) Only used if runOverNetwork = TRUE.
launch.browser	Should the app be launched in your default browser, or in a Shiny window. Note: copying to clipboard will not work in a Shiny window.

Details

Launches a Shiny app that allows the user to explore the diagnostics

---

loadCohortsFromPackage *Load Cohort Definitions From A Study Package*

Description

Load cohort references for usage in executeDiagnostics.

Expects a csv file (e.g one placed in "inst/settings/cohortsToCreate.csv" for the pacakge) with headers: atlasId, referentConceptId, webApiCohortId, cohortId, name, cohortName, logicDescription

For example "' atlasId,referentConceptId,webApiCohortId,cohortId,name,cohortName,logicDescription,phenot
17492,192671,17492,17492,17492,GI bleed,GI bleed,192671000 "'

Usage

```
loadCohortsFromPackage(
  packageName,
  cohortToCreateFile = "settings/cohortsToCreate.csv",
  cohortIds = NULL,
  errorMessage = NULL
)
```

Arguments

packageName	The name of the package containing the cohort definitions. Can be left NULL if baseUrl and cohortSetReference have been specified.
cohortToCreateFile	The location of the cohortToCreate file within the package. Is ignored if baseUrl and cohortSetReference have been specified. The cohortToCreateFile must be .csv file that is expected to be read into a dataframe object identical to requirements for cohortSetReference argument. This csv file is expected to be encoded in either ASCII or UTF-8, if not, an error message will be displayed and process stopped.
cohortIds	Optionally, provide a subset of cohort IDs to restrict the diagnostics to.
errorMessage	checkmate assert collection, used internally for error checks

---

runCohortDiagnostics *Run cohort diagnostics*

Description

Runs the cohort diagnostics on all (or a subset of) the cohorts instantiated using the ROhdsiWebApi::insertCohortDefinitionSetInPackage function. Assumes the cohorts have already been instantiated.

Characterization: If runTemporalCohortCharacterization argument is TRUE, then the following default covariateSettings object will be created using RFeatureExtraction::createTemporalCovariateS
Alternatively, a covariate setting object may be created using the above as an example.

Usage

```
runCohortDiagnostics(
  packageName = NULL,
  cohortToCreateFile = "settings/CohortsToCreate.csv",
  cohortDefinitionSet = NULL,
  baseUrl = NULL,
  cohortSetReference = NULL,
  connectionDetails = NULL,
  connection = NULL,
  cdmDatabaseSchema,
  oracleTempSchema = NULL,
  tempEmulationSchema = getOption("sqlRenderTempEmulationSchema"),
  cohortDatabaseSchema,
  vocabularyDatabaseSchema = cdmDatabaseSchema,
  cohortTable = "cohort",
```

```
 cohortTableNames = CohortGenerator::getCohortTableNames(cohortTable = cohortTable),
 cohortIds = NULL,
 inclusionStatisticsFolder = NULL,
 exportFolder,
 databaseId,
 databaseName = databaseId,
 databaseDescription = databaseId,
 cdmVersion = 5,
 runInclusionStatistics = TRUE,
 runIncludedSourceConcepts = TRUE,
 runOrphanConcepts = TRUE,
 runTimeDistributions = TRUE,
 runVisitContext = TRUE,
 runBreakdownIndexEvents = TRUE,
 runIncidenceRate = TRUE,
 runTimeSeries = FALSE,
 runCohortOverlap = TRUE,
 runCohortCharacterization = TRUE,
 covariateSettings = createDefaultCovariateSettings(),
 runTemporalCohortCharacterization = TRUE,
 temporalCovariateSettings = createTemporalCovariateSettings(useConditionOccurrence =
  TRUE, useDrugEraStart = TRUE, useProcedureOccurrence = TRUE, useMeasurement = TRUE,
  temporalStartDays = c(-365, -30, 0, 1, 31), temporalEndDays = c(-31, -1, 0, 30, 365)),
 minCellCount = 5,
 incremental = FALSE,
 incrementalFolder = file.path(exportFolder, "incremental")
)
```

Arguments

packageName	The name of the package containing the cohort definitions. Can be left NULL if baseUrl and cohortSetReference have been specified.
cohortToCreateFile	The location of the cohortToCreate file within the package. Is ignored if baseUrl and cohortSetReference have been specified. The cohortToCreateFile must be .csv file that is expected to be read into a dataframe object identical to requirements for cohortSetReference argument. This csv file is expected to be encoded in either ASCII or UTF-8, if not, an error message will be displayed and process stopped.
cohortDefinitionSet	Data.frame of cohorts must include columns cohortId, cohortName, json, sql
baseUrl	The base URL for the WebApi instance, for example: "http://server.org:80/WebAPI". Can be left NULL if packageName and cohortToCreateFile have been specified.
cohortSetReference	A data frame with four columns, as described in the details. Can be left NULL if packageName and cohortToCreateFile have been specified.
connectionDetails	An object of type connectionDetails as created using the createConnectionDetails function in the DatabaseConnector package. Can be left NULL if connection is provided.

`connection` An object of type `connection` as created using the `connect` function in the DatabaseConnector package. Can be left NULL if `connectionDetails` is provided, in which case a new connection will be opened at the start of the function, and closed when the function finishes.

`cdmDatabaseSchema`
Schema name where your patient-level data in OMOP CDM format resides. Note that for SQL Server, this should include both the database and schema name, for example 'cdm_data.dbo'.

`oracleTempSchema`
DEPRECATED by DatabaseConnector: use `tempEmulationSchema` instead.

`tempEmulationSchema`
Some database platforms like Oracle and Impala do not truly support temp tables. To emulate temp tables, provide a schema with write privileges where temp tables can be created.

`cohortDatabaseSchema`
Schema name where your cohort table resides. Note that for SQL Server, this should include both the database and schema name, for example 'scratch.dbo'.

`vocabularyDatabaseSchema`
Schema name where your OMOP vocabulary data resides. This is commonly the same as cdmDatabaseSchema. Note that for SQL Server, this should include both the database and schema name, for example 'vocabulary.dbo'.

`cohortTable` Name of the cohort table.

`cohortTableNames`
Cohort Table names used by CohortGenerator package

`cohortIds` Optionally, provide a subset of cohort IDs to restrict the diagnostics to.

`inclusionStatisticsFolder`
The folder where the inclusion rule statistics are stored. Can be left NULL if `runInclusionStatistics = FALSE`.

`exportFolder` The folder where the output will be exported to. If this folder does not exist it will be created.

`databaseId` A short string for identifying the database (e.g. 'Synpuf').

`databaseName` The full name of the database. If NULL, defaults to databaseId.

`databaseDescription`
A short description (several sentences) of the database. If NULL, defaults to databaseId.

`cdmVersion` The version of the OMOP CDM. Default 5. (Note: only 5 is supported.)

`runInclusionStatistics`
Generate and export statistic on the cohort inclusion rules?

`runIncludedSourceConcepts`
Generate and export the source concepts included in the cohorts?

`runOrphanConcepts`
Generate and export potential orphan concepts?

`runTimeDistributions`
Generate and export cohort time distributions?

`runVisitContext`
Generate and export index-date visit context?

`runBreakdownIndexEvents`
: Generate and export the breakdown of index events?

`runIncidenceRate`
: Generate and export the cohort incidence rates?

`runTimeSeries`
: Generate and export the cohort prevalence rates?

`runCohortOverlap`
: Generate and export the cohort overlap? Overlaps are checked within cohortIds that have the same phenotype ID sourced from the CohortSetReference or cohortToCreateFile.

`runCohortCharacterization`
: Generate and export the cohort characterization? Only records with values greater than 0.0001 are returned.

`covariateSettings`
: Either an object of type `covariateSettings` as created using one of the createCovariateSettings function in the FeatureExtraction package, or a list of such objects.

`runTemporalCohortCharacterization`
: Generate and export the temporal cohort characterization? Only records with values greater than 0.001 are returned.

`temporalCovariateSettings`
: Either an object of type `covariateSettings` as created using one of the createTemporalCovariateSettings function in the FeatureExtraction package, or a list of such objects.

`minCellCount`
: The minimum cell count for fields contains person counts or fractions.

`incremental`
: Create only cohort diagnostics that haven't been created before?

`incrementalFolder`
: If `incremental = TRUE`, specify a folder where records are kept of which cohort diagnostics has been executed.

Details

Currently two ways of executing this function are supported, either (1) [Package Mode] embedded in a study package, assuming the cohort definitions are stored in that package using the `ROhdsiWebApi::insertCohortDefinitionSetInPackage`, or (2) [WebApi Mode] By using a WebApi interface to retrieve the cohort definitions.

When using this function in Package Mode: Use the `packageName` and `cohortToCreateFile` to specify the name of the study package, and the name of the cohortToCreate file within that package, respectively

When using this function in WebApi Mode: use the `baseUrl` and `cohortSetReference` to specify how to connect to the WebApi, and which cohorts to fetch, respectively.

Note: if the parameters for both Package Mode and WebApi Mode are provided, then Package mode is preferred.

The `cohortSetReference` argument must be a data frame with the following columns:

cohortId The cohort Id is the id used to identify a cohort definition. This is required to be unique. It will be used to create file names. It is recommended to be (referrentConceptId * 1000) + a number between 3 to 999

atlasId Cohort Id in the webApi/atlas instance. It is a required field to run Cohort Diagnostics in WebApi mode. It is discarded in package mode.

cohortName The full name of the cohort. This will be shown in the Shiny app.

logicDescription A human understandable brief description of the cohort definition. This logic does not have to a fully specified description of the cohort definition, but should provide enough context to help user understand the meaning of the cohort definition

referentConceptId A standard omop concept id that serves as the referent phenotype definition for the cohort Id (optional)

`uploadResults` *Upload results to the database server.*

Description

Requires the results data model tables have been created using the `createResultsDataModel` function.

Set the POSTGRES_PATH environmental variable to the path to the folder containing the psql executable to enable bulk upload (recommended).

Usage

```
uploadResults(
  connectionDetails = NULL,
  schema,
  zipFileName,
  forceOverWriteOfSpecifications = FALSE,
  purgeSiteDataBeforeUploading = TRUE,
  tempFolder = tempdir()
)
```

Arguments

`connectionDetails`
: An object of type `connectionDetails` as created using the `createConnectionDetails` function in the DatabaseConnector package.

`schema`
: The schema on the postgres server where the tables have been created.

`zipFileName`
: The name of the zip file.

`forceOverWriteOfSpecifications`
: If TRUE, specifications of the phenotypes, cohort definitions, and analysis will be overwritten if they already exist on the database. Only use this if these specifications have changed since the last upload.

`purgeSiteDataBeforeUploading`
: If TRUE, before inserting data for a specific databaseId all the data for that site will be dropped. This assumes the input zip file contains the full data for that data site.

`tempFolder`
: A folder on the local file system where the zip files are extracted to. Will be cleaned up when the function is finished. Can be used to specify a temp folder on a drive that has sufficient space if the default system temp space is too limited.