# Package 'Achilles'

January 20, 2025

**Type** Package

**Title** Achilles Data Source Characterization

**Version** 1.7.2

**Date** 2023-05-11

**Maintainer** Frank DeFalco <fdefalco@ohdsi.org>

**Description** Automated Characterization of Health Information at Large-Scale Longitudinal Evidence Systems. Creates a descriptive statistics summary for an Observational Medical Outcomes Partnership Common Data Model standardized data source. This package includes functions for executing summary queries on the specified data source and exporting reporting content for use across a variety of Observational Health Data Sciences and Informatics community applications.

**Depends** DatabaseConnector (>= 2.0.0), R (>= 4.0.0)

**Imports** SqlRender (>= 1.6.0), dplyr, jsonlite, ParallelLogger, readr, data.table, lubridate, tseries, rlang

**Suggests** DT, magrittr, tidyr, knitr, rmarkdown, testthat (>= 3.0.0), withr

**VignetteBuilder** knitr

**License** Apache License

**RoxygenNote** 7.2.3

**Encoding** UTF-8

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** Frank DeFalco [aut, cre],
Patrick Ryan [aut],
Martijn Schuemie [aut],
Vojtech Huser [aut],
Chris Knoll [aut],
Ajit Londhe [aut],
Taha Abdul-Basser [aut],
Anthony Molinaro [aut],
Observational Health Data Science and Informatics [cph]

**Repository** CRAN

**Date/Publication** 2023-05-11 16:50:02 UTC

# Contents

achilles *achilles*

## Description

achilles creates descriptive statistics summary for an entire OMOP CDM instance.

## Usage

```
achilles(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema = cdmDatabaseSchema,
  scratchDatabaseSchema = resultsDatabaseSchema,
  vocabDatabaseSchema = cdmDatabaseSchema,
  tempEmulationSchema = resultsDatabaseSchema,
  sourceName = "",
  analysisIds,
  createTable = TRUE,
  smallCellCount = 5,
  cdmVersion = "5",
  createIndices = TRUE,
  numThreads = 1,
  tempAchillesPrefix = "tmpach",
  dropScratchTables = TRUE,
  sqlOnly = FALSE,
  outputFolder = "output",
  verboseMode = TRUE,
  optimizeAtlasCache = FALSE,
  defaultAnalysesOnly = TRUE,
  updateGivenAnalysesOnly = FALSE,
  excludeAnalysisIds,
  sqlDialect = NULL
)
```

## Arguments

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

cdmDatabaseSchema
: Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

resultsDatabaseSchema
: Fully qualified name of database schema that we can write final results to. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'.

`scratchDatabaseSchema`
: Fully qualified name of the database schema that will store all of the intermediate scratch tables, so for example, on SQL Server, 'cdm_scratch.dbo'. Must be accessible to/from the cdmDatabaseSchema and the resultsDatabaseSchema. Default is resultsDatabaseSchema. Making this "#" will run Achilles in single-threaded mode and use temporary tables instead of permanent tables.

`vocabDatabaseSchema`
: String name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'.

`tempEmulationSchema`
: Formerly oracleTempSchema. For databases like Oracle where you must specify the name of the database schema where you want all temporary tables to be managed. Requires create/insert permissions to this database.

`sourceName`
: String name of the data source name. If blank, CDM_SOURCE table will be queried to try to obtain this.

`analysisIds`
: (OPTIONAL) A vector containing the set of Achilles analysisIds for which results will be generated. If not specified, all analyses will be executed. Use `getAnalysisDetails` to get a list of all Achilles analyses and their Ids.

`createTable`
: If true, new results tables will be created in the results schema. If not, the tables are assumed to already exist, and analysis results will be inserted (slower on MPP).

`smallCellCount`
: To avoid patient identification, cells with small counts (<= smallCellCount) are deleted. Set to 0 for complete summary without small cell count restrictions.

`cdmVersion`
: Define the OMOP CDM version used: currently supports v5 and above. Use major release number or minor number only (e.g. 5, 5.3)

`createIndices`
: Boolean to determine if indices should be created on the resulting Achilles tables. Default= TRUE

`numThreads`
: (OPTIONAL, multi-threaded mode) The number of threads to use to run Achilles in parallel. Default is 1 thread.

`tempAchillesPrefix`
: (OPTIONAL, multi-threaded mode) The prefix to use for the scratch Achilles analyses tables. Default is "tmpach"

`dropScratchTables`
: (OPTIONAL, multi-threaded mode) TRUE = drop the scratch tables (may take time depending on dbms), FALSE = leave them in place for later removal.

`sqlOnly`
: Boolean to determine if Achilles should be fully executed. TRUE = just generate SQL files, don't actually run, FALSE = run Achilles

`outputFolder`
: Path to store logs and SQL files

`verboseMode`
: Boolean to determine if the console will show all execution steps. Default = TRUE

`optimizeAtlasCache`
: Boolean to determine if the atlas cache has to be optimized. Default = FALSE

`defaultAnalysesOnly`
: Boolean to determine if only default analyses should be run. Including non-default analyses is substantially more resource intensive. Default = TRUE

updateGivenAnalysesOnly
: Boolean to determine whether to preserve the results of the analyses NOT specified with the analysisIds parameter. To update only analyses specified by analysisIds, set createTable = FALSE and updateGivenAnalysesOnly = TRUE. By default, updateGivenAnalysesOnly = FALSE, to preserve the original behavior of Achilles when supplied analysisIds.

excludeAnalysisIds
: (OPTIONAL) A vector containing the set of Achilles analyses to exclude.

sqlDialect
: (OPTIONAL) String to be used when specifying sqlOnly = TRUE and NOT supplying the connectionDetails parameter. if the connectionDetails parameter is supplied, sqlDialect is ignored. If the connectionDetails parameter is not supplied, sqlDialect must be supplied to enable SqlRender to translate properly. sqlDialect takes the value normally supplied to connectionDetails$dbms. Default = NULL.

## Details

achilles creates descriptive statistics summary for an entire OMOP CDM instance.

## Value

An object of type achillesResults containing details for connecting to the database containing the results

## Examples

```
## Not run:
connectionDetails <- createConnectionDetails(dbms = "sql server", server = "some_server")
achillesResults <- achilles(connectionDetails = connectionDetails,
  cdmDatabaseSchema = "cdm",
  resultsDatabaseSchema = "results",
  scratchDatabaseSchema = "scratch",
  sourceName = "Some Source",
  cdmVersion = "5.3",
  numThreads = 10,
  outputFolder = "output")

## End(Not run)
```

---

# createIndices *Create indicies*

---

## Description

Create indicies

### Usage

```
createIndices(
  connectionDetails,
  resultsDatabaseSchema,
  outputFolder,
  sqlOnly = FALSE,
  verboseMode = TRUE,
  achillesTables = c("achilles_results", "achilles_results_dist")
)
```

### Arguments

| | |
|---|---|
| connectionDetails | An R object of type connectionDetails created using the function createConnectionDetails in the DatabaseConnector package. |
| resultsDatabaseSchema | Fully qualified name of database schema that we can write final results to. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'. |
| outputFolder | Path to store logs and SQL files |
| sqlOnly | TRUE = just generate SQL files, don't actually run, FALSE = run Achilles |
| verboseMode | Boolean to determine if the console will show all execution steps. Default = TRUE |
| achillesTables | Which achilles tables should be indexed? Default is both achilles_results and achilles_results_dist. |

### Details

Post-processing, create indices to help performance. Cannot be used with Redshift.

### Value

A collection of queries that were executed to drop any existing indices and create new indicies as specified.

---

## createTimeSeries *createTimeSeries*

---

### Description

createTimeSeries Creates a monthly multivariate time series object given a data frame in the proper format.

### Usage

```
createTimeSeries(temporalData)
```

## Arguments

| | |
|---|---|
| temporalData | A data frame from which to create the time series |

## Details

createTimeSeries Requires the following:

1. The given data frame must contain four columns: START_DATE, COUNT_VALUE, PREVALENCE, and PROPORTION_WITHIN_YEAR. 2. START_DATE must be in the YYYYMMDD format. 3. COUNT_VALUE, PREVALENCE, and PROPORTION_WITHIN_YEAR contain only numeric data.

The individual monthly univariate time series can be extracted by specifying the correct column name (see example).

## Value

A multivariate time series object

## Examples

```
# Example 1:
temporalData <- data.frame(START_DATE = seq.Date(as.Date("20210101", "%Y%m%d"),
                                                 as.Date("20231201",
 "%Y%m%d"), by = "month"), COUNT_VALUE = round(runif(36, 1, 1000)), PREVALENCE = round(runif(36,
 0, 10), 2), PROPORTION_WITHIN_YEAR = round(runif(36, 0, 1), 2), stringsAsFactors = FALSE)
dummyTs <- createTimeSeries(temporalData)
dummyTs.cv <- dummyTs[, "COUNT_VALUE"]
dummyTs.pv <- dummyTs[, "PREVALENCE"]
dummyTs.pwy <- dummyTs[, "PROPORTION_WITHIN_YEAR"]

## Not run:
# Example 2:
pneumonia <- 255848
temporalData <- getTemporalData(connectionDetails = connectionDetails, cdmDatabaseSchema = "cdm",
  resultsDatabaseSchema = "results", conceptId = pneumonia)
pneumoniaTs <- createTimeSeries(temporalData)
pneumoniaTs.cv <- pneumoniaTs[, "COUNT_VALUE"]
pneumoniaTs.pv <- pneumoniaTs[, "PREVALENCE"]
pneumoniaTs.pwy <- pneumoniaTs[, "PROPORTION_WITHIN_YEAR"]

## End(Not run)
```

`dropAllScratchTables` *Drop all possible scratch tables*

## Description

Drop all possible scratch tables

## Usage

```
dropAllScratchTables(
  connectionDetails,
  scratchDatabaseSchema,
  tempAchillesPrefix = "tmpach",
  numThreads = 1,
  tableTypes = c("achilles"),
  outputFolder,
  verboseMode = TRUE,
  defaultAnalysesOnly = TRUE
)
```

## Arguments

| | |
|---|---|
| `connectionDetails` | An R object of type `connectionDetails` created using the function `createConnectionDetails` in the `DatabaseConnector` package. |
| `scratchDatabaseSchema` | string name of database schema that Achilles scratch tables were written to. |
| `tempAchillesPrefix` | The prefix to use for the "temporary" (but actually permanent) Achilles analyses tables. Default is "tmpach" |
| `numThreads` | The number of threads to use to run this function. Default is 1 thread. |
| `tableTypes` | The types of Achilles scratch tables to drop: achilles |
| `outputFolder` | Path to store logs and SQL files |
| `verboseMode` | Boolean to determine if the console will show all execution steps. Default = TRUE |
| `defaultAnalysesOnly` | Boolean to determine if only default analyses should be run. Including non-default analyses is substantially more resource intensive. Default = TRUE |

## Details

Drop all possible Achilles scratch tables

## Value

No return value, called to drop interim scratch tables.

exportConditionEraToJson *exportConditionEraToJson*

## Description

exportConditionEraToJson Exports Achilles Condition Era report into a JSON form for reports.

## Usage

```
exportConditionEraToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| cdmDatabaseSchema | Name of the database schema that contains the vocabulary files |
| resultsDatabaseSchema | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| outputPath | folder location to save the JSON files. Default is current working folder |
| vocabDatabaseSchema | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for Condition Era report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportConditionEraToJson(connectionDetails,
                         cdmDatabaseSchema = "cdm4_sim",
                         outputPath = "your/output/path")

## End(Not run)
```

---

`exportConditionToJson` *exportConditionToJson*

---

## Description

`exportConditonToJson` Exports Achilles Condition report into a JSON form for reports.

## Usage

```
exportConditionToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| `connectionDetails` | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| `cdmDatabaseSchema` | Name of the database schema that contains the vocabulary files |
| `resultsDatabaseSchema` | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| `outputPath` | folder location to save the JSON files. Default is current working folder |
| `vocabDatabaseSchema` | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for Condition report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportConditionToJson(connectionDetails,
                      cdmDatabaseSchema = "cdm4_sim",
                      outputPath = "your/output/path")

## End(Not run)
```

---

`exportDashboardToJson` *exportDashboardToJson*

---

## Description

`exportDashboardToJson` Exports Achilles Dashboard report into a JSON form for reports.

## Usage

```
exportDashboardToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| `connectionDetails` | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| `cdmDatabaseSchema` | Name of the database schema that contains the vocabulary files |
| `resultsDatabaseSchema` | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| `outputPath` | folder location to save the JSON files. Default is current working folder |
| `vocabDatabaseSchema` | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for Dashboard report found in Achilles.Web. NOTE: This function reads the results from the other exports and aggregates them into a single file. If other reports are not genreated, this function will fail.

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportDashboardToJson(connectionDetails,
                      cdmDatabaseSchema = "cdm4_sim",
                      outputPath = "your/output/path")

## End(Not run)
```

---

`exportDataDensityToJson` *exportDataDensityToJson*

---

## Description

`exportDataDensityToJson` Exports Achilles Data Density report into a JSON form for reports.

## Usage

```
exportDataDensityToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

`connectionDetails`
: An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port)

`cdmDatabaseSchema`
: Name of the database schema that contains the vocabulary files

`resultsDatabaseSchema`
: Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema

`outputPath` folder location to save the JSON files. Default is current working folder

`vocabDatabaseSchema`
: name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'.

## Details

Creates individual files for Data Density report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                 server = "yourserver")
exportDataDensityToJson(connectionDetails,
                        cdmDatabaseSchema = "cdm4_sim",
                        outputPath = "your/output/path")

## End(Not run)
```

---

`exportDeathToJson` *exportDeathToJson*

---

## Description

`exportDeathToJson` Exports Achilles Death report into a JSON form for reports.

## Usage

```
exportDeathToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

`connectionDetails`
: An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port)

cdmDatabaseSchema
: Name of the database schema that contains the vocabulary files

resultsDatabaseSchema
: Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema

outputPath
: folder location to save the JSON files. Default is current working folder

vocabDatabaseSchema
: name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'.

## Details

Creates individual files for Death report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                 server = "yourserver")
exportDeathToJson(connectionDetails,
                  cdmDatabaseSchema = "cdm4_sim",
                  outputPath = "your/output/path")

## End(Not run)
```

---

# exportDrugEraToJson *exportDrugEraToJson*

---

## Description

exportDrugEraToJson Exports Achilles Drug Era report into a JSON form for reports.

## Usage

```
exportDrugEraToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

### Arguments

| | |
|---|---|
| connectionDetails | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| cdmDatabaseSchema | Name of the database schema that contains the vocabulary files |
| resultsDatabaseSchema | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| outputPath | folder location to save the JSON files. Default is current working folder |
| vocabDatabaseSchema | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

### Details

Creates individual files for Drug Era report found in Achilles.Web

### Value

none

### Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                 server = "yourserver")
exportDrugEraToJson(connectionDetails,
                    cdmDatabaseSchema = "cdm4_sim",
                    outputPath = "your/output/path")

## End(Not run)
```

---

## exportDrugToJson *exportDrugToJson*

---

### Description

exportDrugToJson Exports Achilles Drug report into a JSON form for reports.

## Usage

```
exportDrugToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| cdmDatabaseSchema | Name of the database schema that contains the vocabulary files |
| resultsDatabaseSchema | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| outputPath | folder location to save the JSON files. Default is current working folder |
| vocabDatabaseSchema | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for Drug report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                 server = "yourserver")
exportDrugToJson(connectionDetails,
                 cdmDatabaseSchema = "cdm4_sim",
                 outputPath = "your/output/path")

## End(Not run)
```

exportMeasurementToJson *exportMeasurementToJson*

## Description

exportMeasurementToJson Exports Measurement report into a JSON form for reports.

## Usage

```
exportMeasurementToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| cdmDatabaseSchema | Name of the database schema that contains the vocabulary files |
| resultsDatabaseSchema | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| outputPath | folder location to save the JSON files. Default is current working folder |
| vocabDatabaseSchema | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for Measurement report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportMeasurementToJson(connectionDetails,
                        cdmDatabaseSchema = "cdm4_sim",
                        outputPath = "your/output/path")

## End(Not run)
```

---

exportMetaToJson *exportMetaToJson*

---

## Description

exportMetaToJson Exports Achilles META report into a JSON form for reports.

## Usage

```
exportMetaToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| cdmDatabaseSchema | Name of the database schema that contains the vocabulary files |
| resultsDatabaseSchema | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| outputPath | folder location to save the JSON files. Default is current working folder |
| vocabDatabaseSchema | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for Achilles META report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                 server = "yourserver")
exportMetaToJson(connectionDetails,
                 cdmDatabaseSchema = "cdm4_sim",
                 outputPath = "your/output/path")

## End(Not run)
```

---

`exportObservationPeriodToJson` *exportObservationPeriodToJson*

---

## Description

`exportObservationPeriodToJson` Exports Achilles Observation Period report into a JSON form for reports.

## Usage

```
exportObservationPeriodToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| `connectionDetails` | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| `cdmDatabaseSchema` | Name of the database schema that contains the vocabulary files |
| `resultsDatabaseSchema` | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| `outputPath` | folder location to save the JSON files. Default is current working folder |
| `vocabDatabaseSchema` | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for Observation Period report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportObservationPeriodToJson(connectionDetails,
                              cdmDatabaseSchema = "cdm4_sim",
                              outputPath = "your/output/path")

## End(Not run)
```

---

`exportObservationToJson` *exportObservationToJson*

---

## Description

`exportObservationToJson` Exports Achilles Observation report into a JSON form for reports.

## Usage

```
exportObservationToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| `connectionDetails` | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| `cdmDatabaseSchema` | Name of the database schema that contains the vocabulary files |
| `resultsDatabaseSchema` | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| `outputPath` | folder location to save the JSON files. Default is current working folder |

vocabDatabaseSchema
: name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'.

## Details

Creates individual files for Observation report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                 server = "yourserver")
exportObservationToJson(connectionDetails,
                        cdmDatabaseSchema = "cdm4_sim",
                        outputPath = "your/output/path")

## End(Not run)
```

---

# exportPerformanceToJson

*exportPerformanceToJson exportPerformanceToJson*

---

## Description

exportPerformanceToJson Exports Achilles performance report into a JSON form for reports.

## Usage

```
exportPerformanceToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

connectionDetails
: An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port)

cdmDatabaseSchema
: Name of the database schema that contains the vocabulary files

resultsDatabaseSchema
: Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema

outputPath
: folder location to save the JSON files. Default is current working folder

vocabDatabaseSchema
: name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'.

## Details

Creates performance report including how long each Achilles result took to generate.

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportPerformanceToJson(connectionDetails,
                        cdmDatabaseSchema = "cdm4_sim",
                        outputPath = "your/output/path")

## End(Not run)
```

---

# exportPersonToJson *exportPersonToJson*

---

## Description

exportPersonToJson Exports Achilles Person report into a JSON form for reports.

## Usage

```
exportPersonToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

### Arguments

connectionDetails
: An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port)

cdmDatabaseSchema
: Name of the database schema that contains the vocabulary files

resultsDatabaseSchema
: of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema

outputPath
: folder location to save the JSON files. Default is current working folder

vocabDatabaseSchema
: name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'.

### Details

Creates individual files for Person report found in Achilles.Web

### Value

none

### Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportPersonToJson(connectionDetails,
                   cdmDatabaseSchema = "cdm4_sim",
                   outputPath = "your/output/path")

## End(Not run)
```

---

## exportProcedureToJson *exportProcedureToJson*

---

### Description

exportProcedureToJson Exports Achilles Procedure report into a JSON form for reports.

## Usage

```
exportProcedureToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| cdmDatabaseSchema | Name of the database schema that contains the vocabulary files |
| resultsDatabaseSchema | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| outputPath | folder location to save the JSON files. Default is current working folder |
| vocabDatabaseSchema | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for Procedure report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                 server = "yourserver")
exportProcedureToJson(connectionDetails,
                      cdmDatabaseSchema = "cdm4_sim",
                      outputPath = "your/output/path")

## End(Not run)
```

exportResultsToCSV *exportResultsToCSV*

## Description

exportResultsToCSV exports all results to a CSV file

## Usage

```
exportResultsToCSV(
  connectionDetails,
  resultsDatabaseSchema,
  analysisIds = c(),
  minCellCount = 5,
  exportFolder
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type connectionDetails created using the function createConnectionDetails in the DatabaseConnector package. |
| resultsDatabaseSchema | Fully qualified name of database schema that we can write final results to. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'. |
| analysisIds | (OPTIONAL) A vector containing the set of Achilles analysisIds for which results will be generated. If not specified, all analyses will be executed. Use getAnalysisDetails to get a list of all Achilles analyses and their Ids. |
| minCellCount | To avoid patient identification, cells with small counts (<= minCellCount) are deleted. Set to 0 for complete summary without small cell count restrictions. |
| exportFolder | Path to store results |

## Details

exportResultsToCSV writes a CSV file with all results to the export folder.

## Value

No return value. Called to export CSV file to the file system.

exportToAres *exportToAres*

## Description

exportToAres Exports Achilles statistics for ARES

## Usage

```
exportToAres(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  vocabDatabaseSchema,
  outputPath,
  reports = c()
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| cdmDatabaseSchema | Name of the database schema that contains the OMOP CDM. |
| resultsDatabaseSchema | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| vocabDatabaseSchema | string name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |
| outputPath | A folder location to save the JSON files. Default is current working folder |
| reports | vector of reports to run, c() defaults to all reports<br>See showReportTypes for a list of all report types |

## Details

Creates export files

## Value

none

exportToJson *exportToJson*

## Description

exportToJson Exports Achilles statistics into a JSON form for reports.

## Usage

```
exportToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  reports = getAllReports(),
  vocabDatabaseSchema = cdmDatabaseSchema,
  compressIntoOneFile = FALSE
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| cdmDatabaseSchema | Name of the database schema that contains the OMOP CDM. |
| resultsDatabaseSchema | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| outputPath | A folder location to save the JSON files. Default is current working folder |
| reports | A character vector listing the set of reports to generate. Default is all reports. |
| vocabDatabaseSchema | string name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |
| compressIntoOneFile | Boolean indicating if the JSON files should be compressed into one zip file. Please note that in Windows, the zip application must be stored in the system environment, e.g. Sys.setenv("R_ZIPCMD", "some_path_to_zip"). Due to recursion, the actual Achilles files and folders will be embedded in any parent directories that the source folder has. See showReportTypes for a list of all report types |

## Details

Creates individual files for each report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportToJson(connectionDetails, cdmDatabaseSchema = "cdm4_sim", outputPath = "your/output/path")

## End(Not run)
```

---

`exportVisitDetailToJson` *exportVisitDetailToJson*

---

## Description

`exportVisitDetailToJson` Exports Achilles VISIT_DETAIL report into a JSON form for reports.

## Usage

```
exportVisitDetailToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| `connectionDetails` | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| `cdmDatabaseSchema` | Name of the database schema that contains the vocabulary files |
| `resultsDatabaseSchema` | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| `outputPath` | folder location to save the JSON files. Default is current working folder |
| `vocabDatabaseSchema` | name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |

## Details

Creates individual files for VISIT_DETAIL report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportVisitDetailToJson(connectionDetails,
                        cdmDatabaseSchema = "cdm4_sim",
                        outputPath = "your/output/path")

## End(Not run)
```

---

`exportVisitToJson` *exportVisitToJson*

---

## Description

`exportVisitToJson` Exports Achilles Visit report into a JSON form for reports.

## Usage

```
exportVisitToJson(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  outputPath,
  vocabDatabaseSchema = cdmDatabaseSchema
)
```

## Arguments

| | |
|---|---|
| `connectionDetails` | An R object of type ConnectionDetail (details for the function that contains server info, database type, optionally username/password, port) |
| `cdmDatabaseSchema` | Name of the database schema that contains the vocabulary files |
| `resultsDatabaseSchema` | Name of the database schema that contains the Achilles analysis files. Default is cdmDatabaseSchema |
| `outputPath` | folder location to save the JSON files. Default is current working folder |

vocabDatabaseSchema
: name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'.

## Details

Creates individual files for Visit report found in Achilles.Web

## Value

none

## Examples

```
## Not run:
connectionDetails <- DatabaseConnector::createConnectionDetails(dbms = "sql server",
                                                                server = "yourserver")
exportVisitToJson(connectionDetails,
                  cdmDatabaseSchema = "cdm4_sim",
                  outputPath = "your/output/path")

## End(Not run)
```

---

# getAnalysisDetails *Get all analysis details*

---

## Description

Get all analysis details

## Usage

```
getAnalysisDetails()
```

## Details

Get a list of all analyses with their analysis IDs and strata.

## Value

A data.frame with the analysis details.

## getSeasonalityScore *Get the seasonality score for a given monthly time series*

### Description

The seasonality score of a monthly time series is computed as its departure from a uniform distribution.

### Usage

```
getSeasonalityScore(tsData)
```

### Arguments

| | |
|---|---|
| tsData | A time series object. |

### Details

The degree of seasonality of a monthly time series is based on its departure from a uniform distribution. If the number of cases for a given concept is uniformly distributed across all time periods (in this case, all months), then its monthly proportion would be approximately constant. In this case, the time series would be considered "strictly non-seasonal" and its "seasonality score" would be zero. Similarly, if all cases recur at a single point in time (that is, in a single month), such a time series would be considered "strictly seasonal" and its seasonality score would be 1. All other time series would have a seasonality score between 0 and 1. Currently, only monthly time series are supported.

### Value

A numeric value between 0 and 1 (inclusive) representing the seasonality of a time series.

## getTemporalData *getTemporalData*

### Description

getTemporalData Retrieve specific monthly analyses data to support temporal characterization.

### Usage

```
getTemporalData(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  analysisIds = NULL,
  conceptId = NULL
)
```

## Arguments

| | |
|---|---|
| connectionDetails | An R object of type connectionDetails created using the function createConnectionDetails in the DatabaseConnector package. |
| cdmDatabaseSchema | Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'. |
| resultsDatabaseSchema | Fully qualified name of database schema that we can write final results to. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'. |
| analysisIds | (OPTIONAL) A vector containing the set of Achilles analysisIds for which results will be returned. The following are supported: 202,402,602,702,802,1802,2102. If not specified, data for all analysis will be returned. Ignored if conceptId is given. |
| conceptId | (OPTIONAL) A SNOMED concept_id from the CONCEPT table for which a monthly Achilles analysis exists. If not specified, all concepts for a given analysis will be returned. |

## Details

getTemporalData Assumes achilles has been run.

Currently supported
Achilles monthly analyses are: 202 - Visit Occurrence 402 - Condition occurrence 602 - Procedure Occurrence 702 - Drug Exposure 802 - Observation 1802 - Measurement 2102 - Device

## Value

A data frame of query results from DatabaseConnector

## Examples

```
## Not run:
pneumonia <- 255848
monthlyResults <- getTemporalData(connectionDetails = connectionDetails,
                                  cdmDatabaseSchema = "cdm",

  resultsDatabaseSchema = "results", conceptId = pneumonia)

## End(Not run)
```

## isStationary *Determine whether or not a time series is stationary in the mean*

### Description

Uses the Augmented Dickey-Fuller test to determine when the time series has a unit root.

### Usage

```
isStationary(tsData)
```

### Arguments

| | |
|---|---|
| tsData | A time series object. |

### Details

A time series must have a minimum of three complete years of data. For details on the implementation of the Augmented Dickey-Fuller test, see the tseries package on cran.

### Value

A boolean indicating whether or not the given time series is stationary.

## listMissingAnalyses *listMissingAnalyses*

### Description

`listMissingAnalyses` Find and return analyses that exist in `getAnalysisDetails`, but not in achilles_results or achilles_results_dist

### Usage

```
listMissingAnalyses(connectionDetails, resultsDatabaseSchema)
```

### Arguments

| | |
|---|---|
| connectionDetails | An R object of type `connectionDetails` created using the function `createConnectionDetails` in the `DatabaseConnector` package. |
| resultsDatabaseSchema | Fully qualified name of database schema that contains achilles_results and achilles_results_dist tables. |

## Value

A dataframe which is a subset of `getAnalysisDetails`

## Examples

```
## Not run:
Achilles::listMissingAnalyses(connectionDetails = connectionDetails,
                              resultsDatabaseSchema = "results")

## End(Not run)
```

---

`optimizeAtlasCache` *Optimize atlas cache*

---

## Description

Optimize atlas cache

## Usage

```
optimizeAtlasCache(
  connectionDetails,
  resultsDatabaseSchema,
  vocabDatabaseSchema = resultsDatabaseSchema,
  outputFolder = "output",
  sqlOnly = FALSE,
  verboseMode = TRUE,
  tempAchillesPrefix = "tmpach"
)
```

## Arguments

| | |
|---|---|
| `connectionDetails` | An R object of type `connectionDetails` created using the function `createConnectionDetails` in the `DatabaseConnector` package. |
| `resultsDatabaseSchema` | Fully qualified name of database schema that we can write final results to. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'. |
| `vocabDatabaseSchema` | String name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'. |
| `outputFolder` | Path to store logs and SQL files |
| `sqlOnly` | TRUE = just generate SQL files, don't actually run, FALSE = run Achilles |

verboseMode Boolean to determine if the console will show all execution steps. Default = TRUE

tempAchillesPrefix
The prefix to use for the "temporary" (but actually permanent) Achilles analyses tables. Default is "tmpach"

## Details

Post-processing, optimize data for atlas cache in separate table to help performance.

## Value

The SQL statement executed to update cache tables is returned.

---

# performTemporalCharacterization *performTemporalCharacterization*

---

## Description

performTemporalCharacterization Perform temporal characterization on a concept or family of concepts belonging to a supported Achilles analysis.

## Usage

```
performTemporalCharacterization(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema,
  analysisIds = NULL,
  conceptId = NULL,
  outputFile = "temporal-characterization.csv"
)
```

## Arguments

connectionDetails
An R object of type connectionDetails created using the function createConnectionDetails in the DatabaseConnector package.

cdmDatabaseSchema
Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

resultsDatabaseSchema
Fully qualified name of database schema that we can write final results to. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'.

| | |
|---|---|
| analysisIds | (OPTIONAL) A vector containing the set of Achilles analysisIds for which results will be returned. The following are supported: 202,402,602,702,802,1802,2102. If not specified, data for all analysis will be returned. Ignored if conceptId is given. |
| conceptId | (OPTIONAL) A SNOMED concept_id from the CONCEPT table for which a monthly Achilles analysis exists. If not specified, all concepts for a given analysis will be returned. |
| outputFile | CSV file where temporal characterization will be written. Default is temporalcharacterization.csv. |

## Details

performTemporalAnalyses Assumes achilles has been run.

```
Currently supported Achilles analyses for temporal analyses are:
202  - Visit Occurrence
402  - Condition occurrence
602  - Procedure Occurrence
702  - Drug Exposure
802  - Observation
1802 - Measurement
2102 - Device
```

## Value

A csv file with temporal analyses for each time series

## Examples

```
## Not run:
# Example 1:
pneumonia <- 255848
performTemporalCharacterization(
connectionDetails     = connectionDetails,
cdmDatabaseSchema     = "cdm",
resultsDatabaseSchema = "results",
conceptId             = pneumonia,
 outputFolder          = "output/pneumoniaTemporalChar.csv")

# Example 2:
performTemporalCharacterization(
connectionDetails     = connectionDetails,
cdmDatabaseSchema     = "cdm",
resultsDatabaseSchema = "results",
analysisIds           = c(402,702),
 outputFolder          = "output/conditionAndDrugTemporalChar.csv")

# Example 3:
performTemporalCharacterization(
connectionDetails     = connectionDetails,
```

```
cdmDatabaseSchema     = "cdm",
resultsDatabaseSchema = "results",
 outputFolder          = "output/CompleteTemporalChar.csv")

## End(Not run)
```

---

runMissingAnalyses *runMissingAnalyses*

---

## Description

runMissingAnalyses Automatically find and compute analyses that haven't been executed.

## Usage

```
runMissingAnalyses(
  connectionDetails,
  cdmDatabaseSchema,
  resultsDatabaseSchema = cdmDatabaseSchema,
  scratchDatabaseSchema = resultsDatabaseSchema,
  vocabDatabaseSchema = cdmDatabaseSchema,
  tempEmulationSchema = resultsDatabaseSchema,
  outputFolder = "output",
  defaultAnalysesOnly = TRUE
)
```

## Arguments

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

cdmDatabaseSchema
: Fully qualified name of database schema that contains OMOP CDM schema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_instance.dbo'.

resultsDatabaseSchema
: Fully qualified name of database schema that we can write final results to. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example, on SQL Server, 'cdm_results.dbo'.

scratchDatabaseSchema
: Fully qualified name of the database schema that will store all of the intermediate scratch tables, so for example, on SQL Server, 'cdm_scratch.dbo'. Must be accessible to/from the cdmDatabaseSchema and the resultsDatabaseSchema. Default is resultsDatabaseSchema. Making this "#" will run Achilles in single-threaded mode and use temporary tables instead of permanent tables.

vocabDatabaseSchema
: String name of database schema that contains OMOP Vocabulary. Default is cdmDatabaseSchema. On SQL Server, this should specifiy both the database and the schema, so for example 'results.dbo'.

tempEmulationSchema
: Formerly tempEmulationSchema. For databases like Oracle where you must specify the name of the database schema where you want all temporary tables to be managed. Requires create/insert permissions to this database.

outputFolder
: Path to store logs and SQL files

defaultAnalysesOnly
: Boolean to determine if only default analyses should be run. Including non-default analyses is substantially more resource intensive. Default = TRUE

## Value

No return value. Run to execute analyses currently missing from results.

## Examples

```
## Not run:
Achilles::runMissingAnalyses(connectionDetails = connectionDetails,
                             cdmDatabaseSchema = "cdm",
                             resultsDatabaseSchema = "results",

  outputFolder = "/tmp")

## End(Not run)
```

---

# showReportTypes *showReportTypes*

---

## Description

showReportTypes Displays the Report Types that can be passed as vector values to exportToJson.

## Usage

```
showReportTypes()
```

## Details

exportToJson supports the following report types: "CONDITION","CONDITION_ERA", "DASHBOARD", "DATA_DENSITY", "DEATH", "DRUG", "DRUG_ERA", "META", "OBSERVATION", "OBSERVATION_PERIOD", "PERSON", "PROCEDURE","VISIT"

## Value

none (opens the allReports vector in a View() display)

## Examples

```
## Not run:
showReportTypes()

## End(Not run)
```

---

`sumAcrossYears` *For a monhtly time series, compute sum and proportion by month across all years*

---

## Description

For a monhtly time series, compute sum and proportion by month across all years

## Usage

```
sumAcrossYears(tsData)
```

## Arguments

| | |
|---|---|
| `tsData` | A time series object |

## Value

A data frame reporting the monthly sum across all years and the proportion this sum contributes to the total.

---

`tsCompleteYears` *Trim a monthly time series object to so that partial years are removed*

---

## Description

Trim a monthly time series object to so that partial years are removed

## Usage

```
tsCompleteYears(tsData)
```

## Arguments

| | |
|---|---|
| `tsData` | A time series object |

## Details

This function is only supported for monthly time series

## Value

A time series with partial years removed.

# Index