# Package 'cleanEHR'

December 16, 2017

**Type** Package

**Title** The Critical Care Clinical Data Processing Tools

**Version** 1.0

**Author** Sinan Shi, David Perez-Suarez, Steve Harris, Niall MacCallum, David Brealey, Mervyn Singer, James Hetherington

**Maintainer** Sinan Shi <s.shi@ucl.ac.uk>

**Description** An electronic health care record (EHR) data cleaning and processing platform. It focus on heterogeneous high resolution longitudinal data. It works with Critical Care Health Informatics Collaborative (CCHIC) dataset. It is created to address various data reliability and accessibility problems of EHRs as such.

**Depends** R (>= 3.1.0)

**BugReports** https://github.com/CC-HIC/cleanEHR/issues

**License** GPL-3

**LinkingTo** Rcpp

**Suggests** testthat

**Imports** data.table, XML, yaml, Rcpp, ggplot2, methods, pander, knitr

**VignetteBuilder** knitr

**URL** https://github.com/CC-HIC/cleanEHR, http://www.hic.nihr.ac.uk

**RoxygenNote** 5.0.1

**Collate** 'RcppExports.R' 'ccRecord.R' 'ccTable.R' 'cchic_xml.R' 'data.R' 'data.quality.report.R' 'deltaTime.R' 'demographics.R' 'filter.categorical.R' 'filter.missingness.R' 'filter.range.R' 'imputation.R' 'stdid.R' 'utilities.R' 'zzz.R'

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2017-12-16 22:51:26 UTC

# R topics documented:

---

`+,ccRecord,ccEpisode-method`

*Adding one ccEpisode object to a ccRecord*

---

## Description

Adding one ccEpisode object to a ccRecord

## Usage

```
## S4 method for signature 'ccRecord,ccEpisode'
e1 + e2
```

## Arguments

| | |
|---|---|
| `e1` | ccRecord-class |
| `e2` | ccEpisode-class |

### Value

ccRecord-class

---

`+,ccRecord,ccRecord-method`

*Combine two ccRecord objects*

---

### Description

Combine two ccRecord objects

### Usage

```
## S4 method for signature 'ccRecord,ccRecord'
e1 + e2
```

### Arguments

| | |
|---|---|
| `e1` | ccRecord-class |
| `e2` | ccRecord-class |

### Value

ccRecord-class

---

`+,ccRecord,list-method`

*Adding a list of ccEpisode to ccRecord*

---

### Description

Adding a list of one or multiple ccEpisode objects to a ccRecord object, the information table (infotb) will be updated automatically. It is the more efficient way to add multiple ccEpisode objects.

### Usage

```
## S4 method for signature 'ccRecord,list'
e1 + e2
```

### Arguments

| | |
|---|---|
| `e1` | ccRecord |
| `e2` | a list of ccEpisode objects |

### Value

ccRecord

`+,ccRecord,NULL-method` *Adding nothing to a ccRecord object and return the original ccRecord*

## Description

Adding nothing to a ccRecord object and return the original ccRecord

## Usage

```
## S4 method for signature 'ccRecord,`NULL`'
e1 + e2
```

## Arguments

| | |
|---|---|
| `e1` | ccRecord-class |
| `e2` | NULL |

`as.number` *Convert standard IDs to numbers (character) which can be used for indexing.*

## Description

Convert standard IDs to numbers (character) which can be used for indexing.

## Usage

```
as.number(obj)
```

## Arguments

| | |
|---|---|
| `obj` | a StdId object. |

`ccd` *Synthetic example dataset*

## Description

This dataset has the same data strucutre that of the CCHIC data, though the data are synthetic.

## ccd_demographic_spell *Create demographic table with spell IDs*

### Description

same output like ccd_demographic_table but in addition with a spell ID.

### Usage

```
ccd_demographic_spell(rec, duration = 2)
```

### Arguments

| | |
|---|---|
| rec | ccRecord |
| duration | the maximum hours of transition period |

### Value

data.table demographic table with spell ID in column spell

## ccd_demographic_table *Create the demographic tables, which includes all non-time-varying variables.*

### Description

The data type of each column is in its corresponding data type.

### Usage

```
ccd_demographic_table(record, dtype = TRUE)
```

### Arguments

| | |
|---|---|
| record | ccRecord-class |
| dtype | logical column will be type aware, else all in character. |

## ccd_select_table *Create the table for ccTable from ccRecord*

### Description

Create the table for ccTable from ccRecord

### Usage

```
ccd_select_table(record, items_opt = NULL, items_obg = NULL, freq,
  return_list = FALSE)
```

### Arguments

| | |
|---|---|
| record | ccRecord |
| items_opt | character vectors. Items (HIC code) selected in item_opt are optional items, which will be automatically filled when item is missing. |
| items_obg | obligatory items that is obligatory; Any episode that does not contain item in this vector will be removed. |
| freq | numeric cadence in hour. |
| return_list | logical if TRUE return as a list. |

### Value

data.table

## ccd_unique_spell *find the unique spell ID.*

### Description

find the unique spell ID.

### Usage

```
ccd_unique_spell(rec, duration = 2)
```

### Arguments

| | |
|---|---|
| rec | ccRecord-class |
| duration | integer hours |

### Value

data.table contains spell id.

## ccEpisode-class *The S4 class which holds data of a single episode.*

### Description

The S4 class which holds data of a single episode.

### Fields

`site_id` character string. Site ID, if presented, otherwise "NA".

`episode_id` character string. Episode ID, if presented, otherwise "NA".

`nhs_number` character string. NHS number, if presented, otherwise "NA".

`pas_number` character string. PAS number, if presented, otherwise "NA".

`parse_file` character string. The source XML file. If the source is not a file then "NA".

`t_admission` POSIXct. Time of Admission to the ICU, if presented, otherwise NA.

`t_discharge` POSIXct. Time of discharge of the ICU, if presented, otherwise NA.

`parse_time` POSIXct. Parse time.

`data` A list which holds all the data of this episode which is indexed by NIHIC code.

## ccRecord-class *The S4 class which holds all the CCHIC patient record - served as a database.*

### Description

ccRecord is a class to hold the raw episode data parsed directly from XML or CSV files.

### Fields

`nepisodes` is an integer number indicates the total number of episode the record is holding.

`dmgtb` a data.table containing all the demographic information of each episode, including site_id, NHS number, PAS number, admission date/time, and discharge date/time. This field is usually left empty.

`infotb` a data.table holding the parsing information of each episode such as the parsing time and from which file it parsed from.

`episdoes` a list of ccEpisode objects.

## Examples

```
heart_rate <- data.frame(seq(10), rep(70, 10)) # NIHR_HIC_ICU_0108
site_id <- "Q70" #  NIHR_HIC_ICU_0002
episode_id <- "0000001" # NIHR_HIC_ICU_0005

# Create a new episode
ep <- new.episode(list(NIHR_HIC_ICU_0108=heart_rate,
                       NIHR_HIC_ICU_0002=site_id,
                       NIHR_HIC_ICU_0005=episode_id))

# modifying records
rec <- ccRecord() # a new record
rec <- rec + ep # adding a new episode to the record
rec <- rec + NULL # adding nothing to the record
rec <- rec + rec # adding a record to a record
# Adding a list of episodes
rec <- ccRecord()
ep1 <- new.episode()
ep2 <- new.episode()
eps.list <- list(ep1, ep2)
new.rec <- rec + eps.list
```

---

`ccTable-class` *Process the EHR data in table format*

---

## Description

ccRecord data are re-arranged into tables where the columns stands for data fields (e.g. heart rate, blood pressure) and the rows stands for each data record within a unique cadence. See ccTable_create_cctable. ccTable is the data processing platform. It stores both original data and processed data alongside with the process details. It also contains various commonly used data filters.

## Fields

`record` ccRecord.

`conf` the YAML style configuration.

`torigin` the original data table.

`tclean` the data table after cleaning processes.

`dfilter` list contains data filtering information.

`dquality` list contains data quality.

`summary` list

`base_cadence` the base cadence is specified in hours

## Methods

`apply_filters(warnings = TRUE)` Apply all filters specified in the configuration to update the clean table (tclean)

`create_table(freq)` Create a table contains the selected items in the conf with a given frequency (in hour)

`export_csv(file = NULL)` Export the cleaned table to a CSV file.

`filter_categories()` Check individual entries if they are the in the categories specified in conf.

`filter_missingness(recount = FALSE)` filter out the where missingness is too low.

`filter_nodata()` Exclude episodes when no data is presented in certain fields

`imputation()` Filling missing data to a time series data by performing a given imputation method on a selected window period nearby the missing data.

`reload_conf(conf)` reload yaml configuration.

## Examples

```
rec <- ccRecord()
cctable <- create_cctable(rec, freq=1)
cctable <- cctable$clean()
#table <- cctable$tclean
```

---

# ccTable_apply_filters *Apply all the setup filters.*

---

## Description

Once filters are applied, the processed data will be stored in tclean. Note, running filtering function before apply_filters is necessary. This function will have no effect on tclean if no filter is ran prior. Filters will decide to preserve or remove particular entries or episodes.

## Arguments

| | |
|---|---|
| `warnings` | logical value to indicate more or less messages with an default value TRUE. |

## Examples

```
## Not run:
tb <- create_cctable(ccd, conf, 1)
tb$range_filter()
tb$apply_filter() # apply only the range filter ragardless of the conf.

## End(Not run)
```

## ccTable_clean *Apply all the filters*

### Description

All the filters in configuration will be applied to create the clean dataset. The filters include range, categories, missingness, no_data.

### Examples

```
## Not run:
tb <- create_cctable(ccd, conf, 1)
tb$clean()

## End(Not run)
```

## ccTable_create_cctable *Create a ccTable object*

### Description

This is a member function of ccTable-class. Using create_cctable is a safer and easier way to create the ccTable. See create_cctable.

## ccTable_export_csv *Export the clean table as a CSV file*

### Description

Export tclean as a CSV file.

### Arguments

| | |
|---|---|
| file | the full path of the output CSV file. |

## ccTable_filter_categories

*Categorical data filter*

### Description

Categorical variables only allow a set of values to appear in the variable . Due to various reasons, a categorical variable may contain values that are not standard. The allowed values can be set in the YAML configuration while initialising the ccTable (see ccTable-class, create_cctable). In the following example, we can see how to set up the categorical filter for the variable dead_icu (NIHR_HIC_ICU_0097) which only allows its value to be A, D, E.

### Examples

```
## Not run:
# Example for categorical filter setup in the YAML configuration
NIHR_HIC_ICU_0097:
 category:
   levels:
     A: Alive
     D: Dead
     E: Alive - not discharged
   apply: drop_entry

# Run the filter on ccTable ct
ct$filter_categories() # run the filter
ct$apply_filters()     # apply the filter and create the clean table

## End(Not run)
```

## ccTable_filter_missingness

*Data missing filter*

### Description

Deal with data when insufficient data points are supported. There are two key items to be set in the YAML configuration file. 1) labels – time interval. 2) accept_2d – the accept present ratio. So if we set the labels is 24, and accept_2d is 70. It means we accept all the missing rate that is lower than 30

### Arguments

| | |
|---|---|
| recount | logical value. Recount the missingness if TRUE. |

## ccTable_filter_nodata *No data filter*

### Description

Remove the episode when a particular field is not presented. It need to be set up in the YAML configuration file.

## ccTable_filter_range *Numerical range filter*

### Description

Range filter can only be applied on numerical fields. For those fields which requires a range filter to be applied, one needs to set a series ranges from the broadest to the narrowest in the YAML configuration. We can set three levels (labels) of ranges, red, amber, and green. It is also OK to set only one range instead of three. The range filter will first assign a label to every data entry.

### Arguments

| | |
|---|---|
| `select` | the range label - "red", "amber", "green" If I give "yellow to select, it means I only want the values which is labeled as "yellow" to be in the clean table. |

### Details

The range in the YAML configuration file can be (l, h), [l, h], (l, h], [h, l) standing for close, open and half open intervals.

### Examples

```
## Not run:
# YAML example
NIHR_HIC_ICU_0108:
   shortName: h_rate
   dataItem: Heart rate
   range:
    labels:
     red: (0, 300)     # broader
     amber: (11, 170)
     green: (60, 100)  # narrower
    apply: drop_entry
# apply range filter on ccTable ct
ct$filter_range("yellow")
ct$apply_filters

## End(Not run)
```

## ccTable_reload_conf *Reload the YAML configuration file*

### Description

Note, this function will also reset all the operations and remove the tclean.

### Arguments

| | |
|---|---|
| conf | full path of the YAML configuration file or the parsed config list. |

### Examples

```
## Not run:
tb$reload_conf("REF.yaml")

## End(Not run)
```

## ccTable_reset *Reset the ccTable*

### Description

Restore the object to its initial status. All the filters, quality and the cleaned table will be removed.

## code2stname *Convert NHIC codes to the short names*

### Description

Convert NHIC codes to the short names

### Usage

```
code2stname(code)
```

### Arguments

| | |
|---|---|
| code | character NIHC code, e.g. NIHR_HIC_ICU_0108 |

### Value

shortname character e.g. h_rate

## create2dclean *Clean table - low memory*

### Description

The cleaning process is specified by the YAML configuration. All the filters presented in the configuration will be applied. It returns only the cleaned data. However all the data quality information will be lost. This function is useful when the memory is not sufficiently enough to hold all the information.

### Usage

```
create2dclean(record, config, freq = 1, nchunks = 1)
```

### Arguments

| | |
|---|---|
| record | ccRecord |
| config | the full path of the YAML configuration file |
| freq | table cadence |
| nchunks | integer number. The larger the nchunks the less memory requirement. |

### Value

A cleaned 2d wide table

## create_cctable *Create a ccTable object*

### Description

Re-arrange the ccRecord object to table format where each column stands for a variable and each row a record data point. The number of rows will depend on the sampling frequency set in this function. If the original data has a higher recording frequency than the set frequency (freq), the closest data point will be taken. It is suggested the ‘freq‘ should not be set lower than the maximum recording frequency in the original dataset.

### Usage

```
create_cctable(rec, conf = NULL, freq = 1)
```

### Arguments

| | |
|---|---|
| rec | ccRecord |
| conf | either the path of YAML configuration file or the configuration |
| freq | a unique sampling frequency (in hours) for all variables. e.g. if freq is set to 1, each row in ccTable will represent a record of one hour. |

### Value

ccTable

---

`data.checklist` *This a reference table of NHIC data items.*

---

### Description

This a reference table of NHIC data items.

### Author(s)

Sinan Shi <s.shi@ucl.ac.uk>

---

`data.quality.report` *Create the data quality report*

---

### Description

Create a detailed data quality report, including file summary, site summary, data completeness, and density plot. The result can be found in work_dir/report/data_quality_report.pdf/md. Using this function, one can also create a site/trust specified report, see the argument "site". You need to make sure that you have the right to write into the work_dir.

### Usage

```
data.quality.report(ccd, site = NULL, file = NULL, pdf = TRUE,
  out = "report")
```

### Arguments

| | |
|---|---|
| `ccd` | ccRecord |
| `site` | a vector of the site ids for the site specified report. |
| `file` | character a list of XML file origins. |
| `pdf` | logical create the pdf version of the DQ report, otherwise stay in markdown format |
| `out` | character output path |

### Examples

```
## Not run: data.quality.report(ccd, c("Q70", "C90"))
```

## data.quality.report.brc

*Create the data quality report*

### Description

Create a detailed data quality report, including file summary, site summary, data completeness, and density plot. The result can be found in path/report/data_quality_report.pdf/md. Using this function, one can also create a site/trust specified report, see the argument "site". You need to make sure that you have the right to write into the work_dir.

### Usage

```
data.quality.report.brc(ccd, pdf = TRUE, brc = NULL, path = NULL)
```

### Arguments

| | |
|---|---|
| ccd | ccRecord |
| pdf | logical create the pdf version of the DQ report, otherwise stay in markdown format |
| brc | character BRC names which can be Cambridge, GSTT, Imperial, Oxford, and UCLH. |
| path | report export path |

## deltaTime

*Convert calendar date-time to the time difference comparing to the ICU admission time.*

### Description

Convert calendar date-time to the time difference comparing to the ICU admission time.

### Usage

```
deltaTime(record, pseudotime = FALSE, units = "hours", tdiff = FALSE)
```

### Arguments

| | |
|---|---|
| record | ccRecord |
| pseudotime | logical If pseudotime is set to be TRUE, then the admission and discharge time will be set as the earliest and latest data stamp in the record. |
| units | units of delta time, which can be "hours", "mins", "days". |
| tdiff | if false the delta time will be written in numeric format. |

## demg.distribution *demg.distribution Create a plot of the distribution of numerical demographic data.*

### Description

demg.distribution Create a plot of the distribution of numerical demographic data.

### Usage

```
demg.distribution(demg, names)
```

### Arguments

| | |
|---|---|
| demg | ccRecord or demographic table created by ccd_demographic_table() |
| names | character vector of short names of numerical demographic data. |

### Examples

```
## Not run: tdemg.distribution(ccd, "HCM")
```

## demographic.data.completeness *Create a demographic completeness table (in pander table)*

### Description

Create a demographic completeness table (in pander table)

### Usage

```
demographic.data.completeness(demg, names = NULL, return.data = FALSE)
```

### Arguments

| | |
|---|---|
| demg | data.table the demographic data table created by ccd_demographic_table() |
| names | short name of selected items |
| return.data | logical return the table if TRUE |

## extract_file_origin *Extract the original file name from a path and file removing all the suffixes.*

### Description

Extract the original file name from a path and file removing all the suffixes.

### Usage

```
extract_file_origin(pathfile, removestr = ".xml")
```

### Arguments

| | |
|---|---|
| pathfile | a particular file name which may have a suffix |
| removestr | last bit from the original filename |

### Value

string

## extract_info *Extract information from data.checklist*

### Description

Extract information from data.checklist

### Usage

```
extract_info()
```

### Value

list of time [data.frame(id, idt)], meta [data.frame(id, idmeta)], nontime [numeric], MAX_NUM_NHIC

## file.summary *Produce a file summary table*

### Description

Produce a file summary table

### Usage

```
file.summary(ccd)
```

### Arguments

| | |
|---|---|
| ccd | ccRecord-class |

### Value

data.table

## for_each_episode *loop over all episodes of a ccRecord object*

### Description

loop over all episodes of a ccRecord object

### Usage

```
for_each_episode(record, fun)
```

### Arguments

| | |
|---|---|
| record | ccRecord |
| fun | function |

`getEpisodePeriod` *Get the length of stay based on the first and the last data point.*

## Description

Get the length of stay based on the first and the last data point.

## Usage

```
getEpisodePeriod(e, unit = "hours")
```

## Arguments

| | |
|---|---|
| `e` | ccEpisode object. |
| `unit` | character string. Units in which the results are desired. Can be abbreviated. |

## Value

length of stay

`getXmlepisode` *get the episode data from xml*

## Description

get the episode data from xml

## Usage

```
getXmlepisode(xml.root, id)
```

## Arguments

| | |
|---|---|
| `xml.root` | root of xml data returned by xmlLoad() |
| `id` | integer |

## icnarc2diagnosis *Convert ICNARC codes to diagnosis (text)*

### Description

NOTE: There are still ~600 code missing. see issue #133

### Usage

```
icnarc2diagnosis(icnarc, surgery = TRUE, levels = NULL)
```

### Arguments

| | |
|---|---|
| icnarc | the ICNARC code, e.g. 1.1.1.1.1 |
| surgery | T/F with or without surgical information |
| levels | category level, from [1 - 5]. TODO level 4. |

### Value

character ICNARC diagnosis

## icnarc_table *ICNARC diagnosis reference table*

### Description

ICNARC diagnosis reference table

### References

https://www.icnarc.org/Our-Audit/Audits/Cmp/Resources/Icm-Icnarc-Coding-Method

`inrange` *Check if the values of a vector v is in the given ranges.*

## Description

Check if the values of a vector v is in the given ranges.

## Usage

```
inrange(v, range)
```

## Arguments

| | |
|---|---|
| `v` | vector numeric |
| `range` | A string contains the numeric ranges in a form such as (low, up) for open range and [low, up] for close range. Multiple ranges should be separated by semi-columns which is equivalent to logical OR e.g. (low1, up1); (low2, up2) |

`is.demographic` *Check if the item NHIC code or short name belongs to the demographic category.*

## Description

Check if the item NHIC code or short name belongs to the demographic category.

## Usage

```
is.demographic(item_name)
```

## Arguments

| | |
|---|---|
| `item_name` | character the NHIC code or the short name |

## Value

logical

`is.drugs` *Check if the item NHIC code or short name belongs to the drugs category.*

## Description

Check if the item NHIC code or short name belongs to the drugs category.

## Usage

```
is.drugs(item_name)
```

## Arguments

| | |
|---|---|
| `item_name` | character the NHIC code or the short name |

## Value

logical

`is.laboratory` *Check if the item NHIC code or short name belongs to the Laboratory category.*

## Description

Check if the item NHIC code or short name belongs to the Laboratory category.

## Usage

```
is.laboratory(item_name)
```

## Arguments

| | |
|---|---|
| `item_name` | character the NHIC code or the short name |

## Value

logical

## is.physiology — *Check if the item NHIC code or short name belongs to the physiology category.*

### Description

Check if the item NHIC code or short name belongs to the physiology category.

### Usage

```
is.physiology(item_name)
```

### Arguments

| | |
|---|---|
| item_name | character the NHIC code or the short name |

### Value

logical

## ITEM_REF — *Field reference table*

### Description

Field reference table

## lenstay — *Calculate the length of stay in the ICU.*

### Description

Calculate the length of stay in the ICU and append it to the original demographic table.

### Usage

```
lenstay(demg, units = "hours")
```

### Arguments

| | |
|---|---|
| demg | data.table the demographic table which should at least contain column DAICU and DDICU |
| units | character The unit of lenstay column, by default the output will be in hours |

### Value

data.table It is the original data.table with lenstay column (in difftime) appended.

## long2stname *Convert long names to short names.*

### Description

Convert long names to short names.

### Usage

```
long2stname(l)
```

### Arguments

| | |
|---|---|
| l | long name such as "heart rate" |

### Value

short name character such as "h_rate"

## lookup.items *Lookup items information by keywords*

### Description

This function tries to match keywords in short names, long names and NHIC code. The matched items will be displayed.

### Usage

```
lookup.items(keyword, style = "grid")
```

### Arguments

| | |
|---|---|
| keyword | character e.g. "h_rate", "heart", "108". |
| style | character, the style of the table output which can be "simple", "rmarkdown", and "grid" |

### Value

character the short names of the selected items.

## new.episode *Create a new episode*

### Description

create a new ccEpisode object by given the episode data as a list. The list should be organised in data items and indexed with NIHC code, e.g. NIHR_HIC_ICU_0108.

### Usage

```
new.episode(lt = list(), parse_file = "NA", parse_time = as.POSIXct(NA))
```

### Arguments

| | |
|---|---|
| lt | is a list |
| parse_file | the file location from which the episode comes from. |
| parse_time | the parse date and time of the episode. |

### Value

ccEpisode object

### Examples

```
eps <- list()
eps[["NIHR_HIC_ICU_0018"]] <- data.frame(time=seq(10), rep(70, 10))
new.episode(eps)
```

## physio.distribution *Plot the physiological data distribution.*

### Description

Plot the physiological data distribution.

### Usage

```
physio.distribution(cctb, names)
```

### Arguments

| | |
|---|---|
| cctb | ccTable-class, see create.cctable(). |
| names | character vector of short names of numerical demographic data. |

## plot_episode *Individual episode chart*

### Description

Create an individual episode chart for its diagnosis, drugs and physiological variables. Diagnosis and drugs are always included, while the user can select other longitudinal data.

### Usage

```
plot_episode(r, v)
```

### Arguments

| | |
|---|---|
| r | ccEpisode-class |
| v | short name of longitudinal data. While v is not given, the chart will only display h_rate, spo2, bilirubin, platelets, pao2_fio2, gcs_total. |

### Value

a table of selected vars of an episode

### Examples

```
## Not run:
plot_episode(ccd@episodes[[1]]) # plot first episode with default variables.
plot_episode(ccd@episodes[[1]], "h_rate") # plot first episode heart rate

## End(Not run)
```

## plot_episode,ccEpisode,character-method *Episode chart*

### Description

Episode chart

### Usage

```
## S4 method for signature 'ccEpisode,character'
plot_episode(r, v)
```

### Arguments

| | |
|---|---|
| r | ccEpisode-class |
| v | character |

## plot_episode,ccEpisode,missing-method *Episode chart default fields*

### Description

Episode chart default fields

### Usage

```
## S4 method for signature 'ccEpisode,missing'
plot_episode(r)
```

### Arguments

| | |
|---|---|
| r | ccEpisode-class |

## reallocateTimeRecord *Propagate a numerical delta time interval record.*

### Description

Propagate a numerical delta time interval record.

### Usage

```
reallocateTimeRecord(record, delta = 0.5)
```

### Arguments

| | |
|---|---|
| record | ccRecord |
| delta | time frequency in hours |

### Details

when discharge time and admission time are missing, the latest and the earliest data time stamp will be used instead.

## samplerate2d *Produce a pander table of sample rate of longitudinal data.*

### Description

Produce a pander table of sample rate of longitudinal data.

### Usage

```
samplerate2d(cctb)
```

### Arguments

| | |
|---|---|
| cctb | ccTable-class, see create.cctable(). |

## site.info *Produce a site id reference table.*

### Description

Produce a site id reference table.

### Usage

```
site.info()
```

### Value

data.frame

## StdId *S4 class to hold standard IDs such as "NIHR_HIC_ICU_0001"*

### Description

S4 class to hold standard IDs such as "NIHR_HIC_ICU_0001"

constructor of StdId class

### Usage

```
StdId(text)
```

```
StdId(text)
```

### Arguments

| | |
|---|---|
| text | NIHC code which should be in a format like NIHR_HIC_ICU_xxxx |

### Slots

ids single or multiple characters

---

## stname2code *Convert short names to NHIC codes*

---

### Description

Convert short names to NHIC codes

### Usage

```
stname2code(stname)
```

### Arguments

| | |
|---|---|
| stname | character short names of data item h_rate |

### Value

NIHC code character such as NIHR_HIC_ICU_0108

---

## stname2longname *Convert short names to long names.*

---

### Description

Convert short names to long names.

### Usage

```
stname2longname(stname)
```

### Arguments

| | |
|---|---|
| stname | character short names of data item h_rate |

### Value

longname character such as "heart rate"

## table1 *Produce the item specified table one.*

### Description

Produce the item specified table one.

### Usage

```
table1(demg, names, return.data = FALSE)
```

### Arguments

| | |
|---|---|
| demg | demographic table created by ccd_demographic_table() |
| names | character string. Short names of data items, e.g. h_rate. |
| return.data | logical, FALSE: printing the pander table, TRUE: return the table but not print out the pander table. |

### Value

if return.data is TRUE, return data.table

## total.data.point *Return total data point of the ccRecord object.*

### Description

Return total data point of the ccRecord object.

### Usage

```
total.data.point(ccd)
```

### Arguments

| | |
|---|---|
| ccd | ccRecord-class |

## which.classification *Identify the classification - classification1*

### Description

Identify the classification of a given item code or short name. Classification1 has 5 labels: [1] "Demographic", [2] "Physiology" [3] "Drugs" [4] "Nursing_other" [5] "Laboratory"

### Usage

```
which.classification(item_name)
```

### Arguments

| | |
|---|---|
| item_name | NHIC code or the short name |

### Value

character the item classification

## whichIsCode *give id number from NHIC code like "NIHR_HIC_ICU_xxxx"*

### Description

give id number from NHIC code like "NIHR_HIC_ICU_xxxx"

### Usage

```
whichIsCode(nhic)
```

### Arguments

| | |
|---|---|
| nhic | NHIC code |

---

`xml.file.duration.plot` *plot the duration of XML files.*

---

## Description

plot the duration of XML files.

## Usage

```
xml.file.duration.plot(ccd)
```

## Arguments

| | |
|---|---|
| `ccd` | ccRecord-class |

---

`xml.site.duration.plot` *Plot the XML duration in terms of sites.*

---

## Description

Plot the XML duration in terms of sites.

## Usage

```
xml.site.duration.plot(ccd)
```

## Arguments

| | |
|---|---|
| `ccd` | ccRecord-class |

## xml2Data *Convert the XML file to ccRecord*

### Description

Convert the XML file to ccRecord. For more details, see ccRecord-class.

### Usage

```
xml2Data(file, select.episode = NULL, quiet = TRUE, xml = NULL,
  file_origin = "NA", parse_time = Sys.time())
```

### Arguments

| | |
|---|---|
| file | character string. The path of XML file. |
| select.episode | integer vector. Load only a selected number of episodes. It is NULL by default which loads all the episodes in a file. |
| quiet | logical. Switch on/off the progress bar. |
| xml | XML object. Usually not needed. |
| file_origin | character string. The XML file name. The file name will be extracted automatically while argument xml is NULL. |
| parse_time | POSIXct. By default is the time of the execution of this function. |

### Value

ccRecord-class

## xmlLoad *load xml clinical data*

### Description

load xml clinical data

### Usage

```
xmlLoad(file)
```

### Arguments

| | |
|---|---|
| file | character string. The path of the XML file. |

### Value

the root of the xml data.

## xmlTime2POSIX *Convert time from xml to POSIX format.*

### Description

Convert the XML time The XML time format to POSIXct.

### Usage

```
xmlTime2POSIX(xml.time, allow = FALSE)
```

### Arguments

| | |
|---|---|
| xml.time | character. Time in XML format such as 2014-02-01T03:00:00 |
| allow | logical. Wrong format will be accepted when allow is set to be TRUE and NA will be the return value, otherwise return error. It is useful while dealing with pseudonymous data where the time format is not presented correctly. |

## [,ccRecord,ANY-method *Create a subset of ccRecord object from the original one via specifying the row number of episodes.*

### Description

Create a subset of ccRecord object from the original one via specifying the row number of episodes.

### Usage

```
## S4 method for signature 'ccRecord,ANY'
x[i]
```

### Arguments

| | |
|---|---|
| x | ccRecord-class |
| i | integer vector |

## [,ccRecord,character-method *Create a ccRecord subsetting via selected sites.*

### Description

Create a ccRecord subsetting via selected sites.

### Usage

```
## S4 method for signature 'ccRecord,character'
x[i]
```

### Arguments

| | |
|---|---|
| x | ccRecord-class |
| i | character vector which contains site_ids, e.g. c("Q70", "Q70W") |

## [[,ccRecord-method *Subsetting a ccRecord object and return a list of ccEpisode objects.*

### Description

Subsetting a ccRecord object and return a list of ccEpisode objects.

### Usage

```
## S4 method for signature 'ccRecord'
x[[i]]
```

### Arguments

| | |
|---|---|
| x | ccRecord-class |
| i | integer vector |

# Index