

# Package ‘MIMER’

March 23, 2026

**Title** Data Wrangling for Antimicrobial Resistance Studies

**Version** 1.0.6

**Description** Designed for analyzing the Medical Information Mart for Intensive Care(MIMIC) dataset, a repository of freely accessible electronic health records. MIMER(MIMIC-enabled Research) package, offers a suite of data wrangling functions tailored specifically for preparing the dataset for research purposes, particularly in antimicrobial resistance(AMR) studies. It simplifies complex data manipulation tasks, allowing researchers to focus on their primary inquiries without being bogged down by wrangling complexities.

**License** GPL (>= 3)

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**Imports** AMR, dplyr, fuzzyjoin, rlang, stringr, tidyr, testthat, data.table, reshape2

**Depends** R (>= 4.1.0)

**NeedsCompilation** no

**Author** Anoop Velluva [aut] (ORCID: <<https://orcid.org/0009-0001-6198-6016>>),  
Alessandro Gerada [cre, ctb] (ORCID:  
<<https://orcid.org/0000-0002-6743-4271>>),  
Alexander Howard [ctb] (ORCID: <<https://orcid.org/0000-0002-4195-6821>>)

**Maintainer** Alessandro Gerada <alessandro.gerada@liverpool.ac.uk>

**Repository** CRAN

**Date/Publication** 2026-03-23 11:40:08 UTC

## Contents

check_previous_events . . . . .	2
cleanse_urine_organism_names . . . . .	3
clean_antibiotics . . . . .	5
convert_all_directory_files_as_csv . . . . .	6

download_ndc_files_as_csv . . . . .	6
is_systemic_route . . . . .	7
ndc_is_antimicrobial . . . . .	7
ndc_to_antimicrobial . . . . .	8
remove_files . . . . .	9
transpose_microbioevents . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

check\_previous\_events *Add a column and check any previous events identified for a particular antibiotic*

---

## Description

This function helps to check any previous events identified or not (TRUE/FALSE)

## Usage

```
check_previous_events(df,
  cols,
  sort_by_col,
  patient_id_col,
  event_indi_value="R",
  new_col_prefix="pr_event_",
  time_period_in_days,
  minimum_prev_events,
  default_na_date='9999-12-31 00:00:00')
```

## Arguments

event\_indi\_value (optional) Event value indicating Resistance (Default 'R' )

df A data frame containing microbiology events

cols Columns for each antibiotics which contains events

sort\_by\_col A date column to order the input data frame

patient\_id\_col Patient Id Column

new\_col\_prefix (optional) Custom Prefix for new column(Default 'pr\_event\_')

time\_period\_in\_days (optional) to check any previous events in last 'n' days or not

minimum\_prev\_events (optional) to check any 'n' number of previous events happened or not

default\_na\_date (optional) replacement date string for NA values in sort\_by\_col eg: '9999-12-31 00:00:00'

**Value**

Data Frame

**Examples**

```

#Example -1
test_data <- data.frame(subject_id = c(1000032,
                                     10000280,
                                     10000280,
                                     10000280,
                                     10000280,
                                     10000826,
                                     10000826),
                       chartdate = c('2150-10-12',
                                     '2150-10-12',
                                     '2151-03-17',
                                     '2146-12-08',
                                     '2187-09-26',
                                     '2188-07-01'),
                       AMIKACIN=c('R', 'R', 'S', 'S', 'S', 'R'))

check_previous_events(test_data,
                      cols="AMIKACIN",
                      sort_by_col='chartdate',
                      patient_id_col='subject_id',
                      event_indi_value='R')

#Example -2

test_data <- data.frame(subject_id=c('10016742', '10016742', '10016742',
                                     '10016742', '10016742', '10038332',
                                     '10038332', '10038332', '10038332',
                                     '10038332', '10038332'),
                       chartdate= c('2178-07-03', '2178-08-01', '2178-07-22',
                                     '2178-08-03', '2178-09-25', '2164-07-31',
                                     '2164-12-22', '2164-12-22', '2165-01-07',
                                     '2165-04-17', '2165-05-05'),
                       CEFEPIME=c('R', 'S', 'R', 'S', 'S', 'R', 'R', 'R', 'S', 'S', 'S'),
                       CEFTAZIDIME=c('S', 'R', 'S', 'R', 'R', 'S', 'S', 'S', 'R', 'R', 'S'))

check_previous_events(test_data,
                      cols = c('CEFEPIME', 'CEFTAZIDIME'),
                      sort_by_col = 'chartdate',
                      patient_id_col = 'subject_id',
                      time_period_in_days = 62,
                      minimum_prev_events = 2)

```

---

cleanse\_urine\_organism\_names

*Cleanse organism names in Urine samples*


---

**Description**

Function to preprocess organism names in urine samples. Removes specified strings, maps certain values to standard ones, and filters out unwanted values.

**Usage**

```
cleanse_urine_organism_names(
  data,
  column_name = "org_name",
  strings_to_remove = NULL,
  standard_mapping = NULL,
  filter_values = NULL
)
```

**Arguments**

`data` The dataframe containing urine sample data.

`column_name` The name of the column containing organism names.

`strings_to_remove` A character vector of strings to be removed from the organism names.

`standard_mapping` A named character vector specifying mappings of values to standard ones.

`filter_values` A character vector of values to be filtered out from the organism names.

**Value**

The preprocessed dataframe.

**Examples**

```
data <- data.frame(org_name = c("PRESUMPTIVELY Streptococcus",
  "MODERATE Escherichia coli",
  "S. AUREUS POSITIVE",
  "CANCELLED Influenza A"))
data <- cleanse_urine_organism_names(data,
  column_name = "org_name",
  strings_to_remove = c("POSITIVE FOR",
    "PRESUMPTIVELY", "PRESUMPTIVE",
    "PROBABLE", "IDENTIFICATION",
    "RESEMBLING", "SEEN",
    "MODERATE", "FEW", "BETA",
    "METHICILLIN RESISTANT",
    "NUTRITIONALLY VARIANT",
    "NOT C. PERFRINGENS OR C. SEPTICUM",
    "-LACTAMASE POSITIVE",
    "-LACTAMASE NEGATIVE",
    "VIRAL ANTIGEN",
    "CANDIDA INCONSPICUA",
    "/POSADASII",
```

```

"NOT FUMIGATUS, FLAVUS OR NIGER",
"MRSA POSITIVE", "MRSA NEGATIVE",
"HISTOLYTICA/DISPAR"),
standard_mapping = c(
  "NON-FERMENTER" = "STREPTOCOCCUS",
  "ABIOTROPHIA/GRANULICATELLA" =
    "STREPTOCOCCUS",
  "S. AUREUS POSITIVE" =
    "STAPHYLOCOCCUS AUREUS",
  "ASPERGILLUS FUMIGATUS COMPLEX" =
    "ASPERGILLUS FUMIGATUS",
  "(CRYPTOSPORIDIUM PARVUM OOCYSTS|
  CUNNINGHAMELLA BERTHOLLETIAE|
  EPIDERMOPHYTON FLOCCOSUM|
  EXOPHIALA JEANSELMEI COMPLEX|
  SCEDOSPORIUM|
  NEOASCOCHYTA DESMAZIERI|
  NEOSCYTALIDIUM DIMIDIATUM|
  LOMENTOSPORA|NEUROSPORA|
  PERONEUTYPA SCOPARIA|
  SPOROTHRIX SCHENCKII COMPLEX|
  ZYGOSACCHAROMYCES FERMENTATI)" =
    "UNKNOWN FUNGUS"
),
filter_values = c('CANCELLED|VIRUS|SIMPLEX|PARAINFLUENZA|
  INFLUENZA A|INFLUENZA B|TICK|
  AFB GROWN|GRAM VARIABLE RODS|HYMENOLEPIS'))

```

---

clean_antibiotics	<i>Clean Antibiotics using Fuzzy Matching with AMR Packages's Antibiotics Data.</i>
-------------------	---

---

### Description

This utility helps to identify and clean-up antibiotics names passed to it. Also helps to identify whether a medicine is antibiotic or not

### Usage

```
clean_antibiotics(x , ...)
```

### Arguments

x	character vector or a dataframe containing medicine details
...	column name with drug details (required only if first parameter is a dataframe)

### Value

Character Vector or Data Frame

**Examples**

```
clean_antibiotics(c("Amoxicilin", "Amoxicillin", "Paracetamol"))

df <- data.frame(medicine = c("Amoxicilin", "Amoxicillin", "Paracetamol"))
clean_antibiotics(df, drug_col=medicine)
```

---

```
convert_all_directory_files_as_csv
```

*Convert All Directory Files to CSV - To be used in the Manual Build Process*

---

**Description**

This function converts all text files in a directory to CSV format.

**Usage**

```
convert_all_directory_files_as_csv(extract_dir, delete_text_files = FALSE)
```

**Arguments**

`extract_dir` The directory containing the text files to be converted.  
`delete_text_files` Logical indicating whether to delete the original text files after conversion.

---

```
download_ndc_files_as_csv
```

*Download and Convert NDC Files to CSV - To be used in the Manual Build Process*

---

**Description**

This function downloads a zip file containing NDC (National Drug Code) text files from a specified URL, extracts the files to a directory, converts them to CSV format, and optionally deletes the original text files.

**Usage**

```
download_ndc_files_as_csv(
  zip_file_url = "https://www.accessdata.fda.gov/cder/ndctext.zip",
  zip_file_name = "ndctext.zip",
  extract_dir = "inst/extdata/ndctext"
)
```

**Arguments**

zip_file_url	The URL of the zip file containing NDC text files to download.
zip_file_name	The name of the downloaded zip file.
extract_dir	The directory where the NDC text files will be extracted.

---

is_systemic_route	<i>Check 'route' is systemic or not</i>
-------------------	---

---

**Description**

Function to check 'route' is Systemic or not.

**Usage**

```
is_systemic_route(route, class_names)
```

**Arguments**

route	A vector containing route code.
class_names	A vector containing relevant_routes_administration class <ul style="list-style-type: none"><li>• Eg: PO/NG</li></ul>

**Details**

is\_systemic\_route

**Value**

Boolean

---

ndc_is_antimicrobial	<i>Check 'ndc' code is belongs to any Antimicrobial.</i>
----------------------	--

---

**Description**

Function to check input 'ndc' code is belongs to any Antimicrobial or not.

**Usage**

```
ndc_is_antimicrobial(ndc,  
  class_names,  
  include_missing_NDCs = TRUE)
```

**Arguments**

ndc                    A vector containing ndc codes. Will be coerced to character vector.

class\_names         A vector containing antibacterial classes

- eg: c("antimicrobial", "antibacterial")

include\_missing\_NDCs

                      includes a hardcoded database of NDCs that are present in MIMIC-IV but not in NDC database.

**Details**

ndc\_is\_antimicrobial

#' Note: Certain medications, such as Rifaximin, may not be considered antibiotics for all purposes despite their classification. Specific cases should be handled based on context and requirements.

**Value**

Boolean vector for whether input ndc code corresponds to an antimicrobial

---

ndc\_to\_antimicrobial    *Convert 'ndc' code to corresponding Antibiotic code.*

---

**Description**

Function to convert 'ndc' code to corresponding Antibiotic code.

**Usage**

```
ndc_to_antimicrobial(ndc,
  class_names,
  include_missing_NDCs = TRUE)
```

**Arguments**

ndc                    A vector containing ndc codes. Will be coerced to character.

class\_names         A vector containing antibacterial class names - eg: c("antimicrobial", "antibacterial").

include\_missing\_NDCs

                      includes a hardcoded database of NDCs that are present in MIMIC-IV but not in NDC database.

**Details**

ndc\_to\_antimicrobial

**Value**

Vector of antimicrobials in antibiotic class from AMR package.

---

remove_files	<i>Remove Files - To be used in the Manual Build Process</i>
--------------	--

---

**Description**

This function removes files specified by their filenames from a given directory.

**Usage**

```
remove_files(extract_dir, filenames)
```

**Arguments**

extract_dir	The directory from which files will be removed.
filenames	A character vector containing the names of the files to be removed.

---

transpose_microbioevents	<i>Transpose microbiology events dataset</i>
--------------------------	--

---

**Description**

This function helps to transpose (rows to columns) microbiology events.

**Usage**

```
transpose_microbioevents(raw_df,
                          key_columns,
                          required_columns,
                          transpose_key_column,
                          transpose_value_column,
                          fill="NA",
                          non_empty_filter_column,
                          remove_duplicates=TRUE)
```

**Arguments**

key_columns	(Optional) Primary Key/ Key columns for duplicate check : Default Value = c('subject_id','micro_specimen_id','isolate_num','org_name','ab_itemid')
raw_df	A data frame containing microbiology events
required_columns	(Optional) columns should contain in final dataset : Default Value c('subject_id','hadm_id','micro_specimen_id','order_provider_id','chartdate','charttime','spec_itemid','spec_type_desc','storedate','storetime','test_itemid','test_name','org_itemid','isolate_num','org_name')

transpose\_key\_column  
 (Optional) The column that should be transposed ( - distinct values of that column will become separate columns) :Default 'ab\_name'

transpose\_value\_column  
 (optional) Values of 'transpose\_key\_column' column :Default 'interpretation'

fill  
 (optional) Fill character for empty columns- Default : "NA"

non\_empty\_filter\_column  
 (optional) Filter input dataframe where 'non\_empty\_filter\_column' is not empty or na. Default : 'ab\_itemid'

remove\_duplicates  
 (optional) Default :TRUE

**Value**

Data Frame

**Examples**

```
test_data <- data.frame(subject_id=c('10016742', '10016742', '10016742',
                                     '10016742', '10016742', '10038332',
                                     '10038332', '10038332', '10038332',
                                     '10038332', '10038332'),
  chartdate= c('2178-07-03', '2178-08-01', '2178-08-01',
               '2178-08-01', '2178-09-25', '2164-07-31',
               '2164-12-22', '2164-12-22', '2165-01-07',
               '2165-04-17', '2165-05-05'),
  ab_name=c('CEFEPIME', 'CEFTAZIDIME', 'CEFEPIME',
            'CEFEPIME', 'CEFTAZIDIME', 'CEFTAZIDIME',
            'CEFEPIME', 'CEFEPIME', 'CEFTAZIDIME',
            'CEFTAZIDIME', 'CEFEPIME'),
  interpretation=c('S', 'R', 'S', 'R', 'R', 'S', 'S', 'S', 'R', 'R', 'S'))

transpose_microbioevents(test_data,
  key_columns = c('subject_id', 'chartdate', 'ab_name'),
  required_columns = c('subject_id', 'chartdate'),
  transpose_key_column = 'ab_name',
  transpose_value_column = 'interpretation',
  fill = "N/A",
  non_empty_filter_column = 'subject_id')
```

# Index

`add_events (check_previous_events)`, 2

`check_previous_events`, 2

`clean_antibiotics`, 5

`cleanse_urine_organism_names`, 3

`convert_all_directory_files_as_csv`, 6

`download_ndc_files_as_csv`, 6

`duplicated_microbioevents_records`  
    (`transpose_microbioevents`), 9

`is_systemic_route`, 7

`ndc_is_antimicrobial`, 7

`ndc_to_antimicrobial`, 8

`remove_files`, 9

`transpose_microbioevents`, 9