## Package ‘IPEDSuploadables’

November 1, 2023

**Title**  
Transforms Institutional Data into Text Files for IPEDS  
Automated Import/Upload

**Version** 2.8.7

**Description**  
Starting from user-supplied institutional data, these scripts transform, aggregate, and reshape the information to produce key-value pair data files that are able to be uploaded to IPEDS (Integrated Postsecondary Education Data System) through their submission portal [https://surveys.nces.ed.gov/ipeds/](https://surveys.nces.ed.gov/ipeds/). Starting data specifications can be found in the vignettes. Final files are saved locally to a location of the user’s choice. User-friendly readable files can also be produced for purposes of data review and validation.

**Note**  
Because IPEDS requirements may change from year to year, having the most recent version of this package is highly recommended. Old versions can be found as GitHub branches. The package can also be used to convert any correctly-prepared data into a key-value pair format for any survey (IPEDS or non-IPEDS).

**URL**  
https://github.com/AlisonLanski/IPEDSuploadables,  
https://alisonlanski.github.io/IPEDSuploadables/

**BugReports**  
https://github.com/AlisonLanski/IPEDSuploadables/issues

**License**  
MIT + file LICENSE

**Encoding**  
UTF-8

**LazyData** true

**RoxygenNote** 7.2.3

**Imports**  
dplyr (>= 1.0.0), lubridate, magrittr, purrr, rlang, stringr, svDialogs, tidyr (>= 1.0.0), utils

**Suggests**  
knitr, rmarkdown, kableExtra, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**Depends**  
R (>= 2.10)

**Config/testthat/edition** 2

**NeedsCompilation** no
R topics documented:

Author  Alison Lanski [aut, cre],
        Shiloh Fling [aut]
Maintainer Alison Lanski <alanski@nd.edu>
Repository CRAN
Date/Publication  2023-11-01 16:30:02 UTC

R topics documented:

apply_upload_format .................................................. 4
com_cips ........................................................................... 4
com_students ................................................................. 5
create_dummy_data_com .................................................... 5
create_dummy_data_e1d ..................................................... 6
create_dummy_data_efl ...................................................... 7
create_dummy_data_gr ...................................................... 8
create_dummy_data_gr200 .................................................. 8
create_dummy_data_hr ...................................................... 9
create_dummy_data_om ...................................................... 10
e1d_instr ........................................................................... 10
e1d_students ................................................................. 11
ef1Retention ................................................................. 11
ef1_students ............................................................... 12
get_ipeds_unitid ............................................................ 12
gr200_students ............................................................. 13
gr_students ................................................................. 13
hr_staff ........................................................................... 14
IPEDSuploadables ............................................................. 14
make_com_part_A ............................................................ 14
make_com_part_B ............................................................ 15
make_com_part_C ............................................................ 15
make_com_part_D ............................................................ 16
make_com_part_E ............................................................ 16
make_e1d_part_A ............................................................. 17
make_e1d_part_B ............................................................. 17
make_e1d_part_C ............................................................. 18
make_e1d_part_D ............................................................. 18
make_e1d_part_E ............................................................. 19
make_ef1_part_A ............................................................. 19
make_ef1_part_B ............................................................. 20
make_ef1_part_C ............................................................. 20
make_ef1_part_D ............................................................. 21
make_ef1_part_E ............................................................. 21
make_ef1_part_F ............................................................. 22
make_ef1_part_G ............................................................. 22
make_ef1_part_H ............................................................. 23
make_gr200 ................................................................. 23
make_gr_part_B ............................................................. 24
topics documented:

- make_gr_part_C .............................................. 24
- make_gr_part_E .............................................. 25
- make_hr_part_A .............................................. 25
- make_hr_part_A2 ............................................. 26
- make_hr_part_B1 ............................................. 26
- make_hr_part_B2 ............................................. 27
- make_hr_part_B3 ............................................. 27
- make_hr_part_D1 ............................................. 28
- make_hr_part_D2 ............................................. 28
- make_hr_part_D3 ............................................. 29
- make_hr_part_D4 ............................................. 29
- make_hr_part_G1 ............................................. 30
- make_hr_part_G2 ............................................. 30
- make_hr_part_H1 ............................................. 31
- make_hr_part_H2 ............................................. 31
- make_om_part_A ............................................. 32
- make_om_part_B ............................................. 32
- make_om_part_C ............................................. 33
- make_om_part_D ............................................. 33
- om_students .................................................. 34
- prep_com_data_frame ....................................... 34
- prep_ef1_data_frame ........................................ 35
- prep_hr_data_frame .......................................... 35
- prep_om_awards .............................................. 36
- prep_om_data_frame ......................................... 36
- produce_com_report .......................................... 37
- produce_e1d_report ......................................... 38
- produce_ef1_report .......................................... 39
- produce_gr200_report ....................................... 40
- produce_gr_report ............................................ 41
- produce_hr_report ............................................ 42
- produce_om_report .......................................... 42
- produce_other_report ....................................... 43
- set_report_path .............................................. 44
- specs_COM .................................................... 44
- specs_E1D ..................................................... 45
- specs_EF1 ..................................................... 45
- specs_GR ....................................................... 45
- specs_GR200 .................................................. 46
- specs_HR ....................................................... 46
- specs_OM ....................................................... 46
- write_report .................................................. 47
- write_report_csv ............................................. 47

Index 49
**apply_upload_format**  
*Shortcut function to turn a dataframe into key-value pairs*

**Description**
Shortcut function to turn a dataframe into key-value pairs

**Usage**
```
apply_upload_format(df)
```

**Arguments**
- `df` dataframe with upload-compatible column names in upload-compatible order

**Value**
a dataframe with one column and upload-compatible rows

---

**com_cips**  
*Dummy cip data for Completions functions*

**Description**
Contains sample values for extra cip codes

**Usage**
```
com_cips
```

**Format**
A data frame with 3 rows and 10 columns

**Details**
See complete information by running `?create_dummy_data_com.R`
**com_students**

*Dummy student data for Completions functions*

**Description**

Contains sample values for students

**Usage**

```
com_students
```

**Format**

A data frame with 105 rows and 13 columns

**Details**

See complete information by running `?create_dummy_data_com.R`

---

**create_dummy_data_com**  
Create dummy data for testing the completions functions

**Description**

Creates a prepared dataframe to test scripts related to IPEDS Completions reporting. Produces either a student/degree dataframe or a dataframe of cips previously reported but not in the current student data, depending on the argument you select

**Usage**

```
create_dummy_data_com(df_type = "student")
```

**Arguments**

- `df_type`  
a string: "student" to get the main df needed, "cip" to get extracips

**Value**

a dataframe ready for the rest of the comp scripts
The final dataset has 60 students with 105 majors. Students 100-130, 140, 150 have 1 major for 1 degree (journalism) Students 131-139 have 2 majors for 1 degree (journalism + parks) Students 141-149 have 3 majors for 1 degree (journalism, parks, linguistics) Students 151-159 have 3 majors for 2 degrees (1 degree with journalism/parks, 1 MBA degree) Note: 1 student has a faulty birthdate; this will show the warning "1 failed to parse"

Two rows (level 18 linguistics) are flagged as distance education

To fully process completions, we will need to include an example of a CIP code that is a possible major but has no completers and a CIP code in an award level that is possible but has no completers

This is the second piece of dummy df produced

Examples

```r
set.seed(1892)

# one date fails to parse:
# this is to provide an example of missing
# data which is acceptable to IPEDS
students <- create_dummy_data_e1d()

additional_cips <- create_dummy_data_e1d(df_type = "cip")
```

---

**Description**

Creates a prepared dataframe to test scripts related to IPEDS 12 Month Enrollment reporting. Produces either a student dataframe or a dataframe of instructional activity, depending on the argument you select

**Usage**

```r
create_dummy_data_e1d(df_type = "student")
```

**Arguments**

- `df_type` a string: "student" to get the main df needed, "instr" to get instructional activity

**Value**

a dataframe ready for the rest of the e1d scripts
Note
The final dataset has 100 students: 60 UG students (40 FT, 20 PT; 26 seeking degrees, 34 not) UG include: 20 first time, 20 transfer, 20 continuing/returning; 40 Grad Students (10 FT, 30 PT; 24 seeking degrees, 16 not).
For simplicity, only 1 race-ethnicity category is used: 5 UG and 5 Grad are set to be fully distance ed; 10 UG are set to be partially distance ed.

Examples
```r
set.seed(1892)

student_df <- create_dummy_data_ef1()

instr_df <- create_dummy_data_ef1(df_type = "instr")
```

create_dummy_data_ef1 Create dummy data for testing the fall enrollment functions

Description
Creates students and retention dataframes for use in parts A, B, C, D, E, G, H. Student-faculty ratio (part G) will ask for a number when the function is run and does not need to exist here. To create both dataframes, run the function twice with different arguments, and save results into separate objects.

Usage
```r
create_dummy_data_ef1(df_type = "students", n = 100)
```

Arguments
- **df_type** A string with the dummy data requested ("students" for parts A-D & G-H or "retention" for part E)
- **n** A number

Value
A text file

Examples
```r
set.seed(1234)

#default creates 100 students
students <- create_dummy_data_ef1()

#change the dataframe
retention <- create_dummy_data_ef1(df_type = "retention")
```
#change the population size
more_students <- create_dummy_data_ef1(df_type = "students", n = 250)

---

**create_dummy_data_gr**  
Create dummy data for testing the Grad Rates functions

**Description**

Creates dummy data for testing the Grad Rates functions

**Usage**

create_dummy_data_gr(n = 100)

**Arguments**

- **n**
  - Number of rows of data to synthesize

**Value**

- a dataframe ready for the rest of the Grad Rates functions

**Examples**

#use this seed to reproduce the dummy data saved to the package
set.seed(4567)

#default makes 100 students
graduated <- create_dummy_data_gr()

more_graduated <- create_dummy_data_gr(n = 500)

---

**create_dummy_data_gr200**  
Create dummy data for testing the Grad Rates 200 function

**Description**

Dummy data for Grad Rates 200 testing

**Usage**

create_dummy_data_gr200(n = 1000)
create_dummy_data_hr

**Arguments**

n  
A number that will be used as the length of the data frame

**Value**

a dataframe ready for the rest of the Grad Rates 200 functions

**Examples**

```r
set.seed(4567)

#default creates 1000 students
graduates <- create_dummy_data_gr200()
more_graduates <- create_dummy_data_gr200(n = 100)
```

create_dummy_data_hr  
*Create dummy data for testing the hr functions*

**Description**

to do: save this out into the package and make it accessible as package data

**Usage**

```r
create_dummy_data_hr()
```

**Value**

a dataframe ready for the rest of the hr scripts

**Examples**

```r
set.seed(4567)
hr_pop <- create_dummy_data_hr()
```
**create_dummy_data_om**  Create dummy data for testing the outcome measures functions

**Description**

Creates a prepared dataframe to test scripts related to IPEDS Outcome Measures reporting. Produces either a student/status dataframe

**Usage**

```r
create_dummy_data_om()
```

**Details**

Remember: want to save this data out into the package so it’s available

**Value**

A dataframe ready for the rest of the om scripts

**Note**

The final dataset has 20 students covering most statuses

**Examples**

```r
#creates a very specific population
#function does not allow for anything to be updated at time of run
#in other words: will always create a fixed-value dataframe
dat <- create_dummy_data_om()
```

**e1d_instr**  Dummy aggregated data for 12 Month Enrollment part B

**Description**

Contains sample values for credit hours generated and doctors-professional FTE

**Usage**

```r
e1d_instr
```

**Format**

A data frame with 1 row and 5 columns

**Details**

See complete information by running `?create_dummy_data_e1d.R`
**e1d_students**

*Dummy student-level data for 12 Month Enrollment parts A, C, and E*

**Description**
Contains 100 fictional student records with all required data

**Usage**
e1d_students

**Format**
A data frame with 100 rows (students) and 13 columns

**Details**
See complete information by running `?create_dummy_data_e1d.R`

---

**ef1_retention**

*Dummy student retention data for Fall Enrollment scripts part E*

**Description**
This data provides aggregated counts in a dataframe suitable for use in the retention component of the Fall Enrollment survey.

**Usage**
ef1_retention

**Format**
A data frame with 2 rows and 6 columns
**Description**

Using the default number of students, this data provides a population that touches most available categories of student reporting. Some columns use only a selection of possible values to reduce complexity.

**Usage**

```r
ef1_students
```

**Format**

A data frame with 100 rows and 25 columns

**Note**

To recreate the saved dataframe exactly, use seed 1234 with 100 students.

---

**Description**

Grab institution’s UNITID from supplied data to populate missing-data rows

**Usage**

```r
get_ipeds_unitid(df)
```

**Arguments**

- `df` : a dataframe with ipeds data and one unitid

**Value**

- a character unitid
**gr200_students**  
*Dummy student data for Graduation Rates 200 functions*

**Description**
Contains sample values for students

**Usage**
gr200_students

**Format**
A data frame with 1000 rows and 5 columns

**Details**
See complete information by running `?create_dummy_data_gr200.R`

---

**gr_students**  
*Dummy student data for the Graduation Rates scripts*

**Description**
Dummy student data for the Graduation Rates scripts

**Usage**
gr_students

**Format**
A data frame with 101 rows and 14 columns

**Details**
Includes only 3 Race/Ethnicity categories [6, 7, 8] for simpler code; one student (a program-switcher) has a 4th category [1] for easy tracking
### hr_staff

**Description**
Contains sample values for staff

**Usage**
hr_staff

**Format**
A data frame with 3600 rows and 13 columns

**Details**
See complete information by running `?create_dummy_data_hr.R`

---

### IPEDSuploadables

**Description**
Tools to assist data formatting for upload to IPEDS surveys

**Details**
See the README on GitHub

---

### make_com_part_A

**Description**
Make Completions Part A

**Usage**
`make_com_part_A(df, extracips = NULL)`

**Arguments**
- `df` A dataframe of student/degree information
- `extracips` A dataframe of cips offered by the institution but not in `df`
**make_com_part_B**

**Value**

A text file

---

**make_com_part_B**  
*Make Completions Part B*

**Description**

Make Completions Part B

**Usage**

```r
make_com_part_B(df, extracips = NULL)
```

**Arguments**

- `df`: A dataframe of student/degree information
- `extracips`: A dataframe of cips offered by the institution but not in 'df'

**Value**

A text file

---

**make_com_part_C**  
*Make Completions Part C*

**Description**

Make Completions Part C

**Usage**

```r
make_com_part_C(df)
```

**Arguments**

- `df`: A dataframe of student/degree information

**Value**

A text file
**make_com_part_D**

*Make Completions Part D*

**Description**
Make Completions Part D

**Usage**

```r
make_com_part_D(df, extracips = NULL)
```

**Arguments**

- **df**
  A dataframe of student/degree information
- **extracips**
  A dataframe of cips offered by the institution but not in 'df'

**Value**
A text file

---

**make_com_part_E**

*Make Completions Part E (gender details)*

**Description**
Make Completions Part E (gender details)

**Usage**

```r
make_com_part_E(df, ugender, ggender)
```

**Arguments**

- **df**
  A dataframe of student/degree information
- **ugender**
  A boolean: TRUE means you are collecting and able to report "another gender" for undergraduate completers, even if you have no (or few) such students. Set as FALSE if necessary
- **ggender**
  A boolean: TRUE means you are collecting and able to report "another gender" for graduate completers, even if you have no (or few) such students. Set as FALSE if necessary

**Value**
A text file
**make_e1d_part_A**  
*Make 12 Month Enrollment Part A*

**Description**

Make 12 Month Enrollment Part A

**Usage**

make_e1d_part_A(df)

**Arguments**

| df | A dataframe of student/degree information |

**Value**

A text file

---

**make_e1d_part_B**  
*Make 12 Month Enrollment Part B*

**Description**

Make 12 Month Enrollment Part B

**Usage**

make_e1d_part_B(df)

**Arguments**

| df | A dataframe with summarized credit hours and student information |

**Value**

A text file
**make_e1d_part_C**  
*Make 12 Month Enrollment Part C*

**Description**

Make 12 Month Enrollment Part C

**Usage**

make_e1d_part_C(df)

**Arguments**

- **df**: A dataframe of student/degree information

**Value**

A text file

---

**make_e1d_part_D**  
*Make 12 Month Enrollment Part D (gender details)*

**Description**

Make 12 Month Enrollment Part D (gender details)

**Usage**

make_e1d_part_D(df, ugender, ggender)

**Arguments**

- **df**: A dataframe of student/degree information
- **ugender**: A boolean: TRUE means you are collecting and able to report "another gender" for undergraduate students, even if you have no (or few) such students. Set as FALSE if necessary
- **ggender**: A boolean: TRUE means you are collecting and able to report "another gender" for graduate students, even if you have no (or few) such students. Set as FALSE if necessary

**Value**

A text file
**make_e1d_part_E**  

*Make 12 Month Enrollment Part E*

**Description**

R/E and Gender counts for dual enrollment (high school students)

**Usage**

```r
make_e1d_part_E(df)
```

**Arguments**

- **df**
  
  A dataframe of student/degree information

**Value**

A text file

---

**make_ef1_part_A**  

*Make Fall Enrollment Part A*

**Description**

Breakdown of students level and demographics; also by designated CIPs in required years

**Usage**

```r
make_ef1_part_A(df, cips = TRUE)
```

**Arguments**

- **df**
  
  A dataframe of student information

- **cips**
  
  A logical indicating if part A needs to provide breakdowns by particular CIPs

**Value**

A text file
make_ef1_part_B  Make Fall Enrollment Part B

Description
Student Counts by Age/gender

Usage
make_ef1_part_B(df)

Arguments
df  A dataframe of student information

Value
A text file

make_ef1_part_C  Make Fall Enrollment Part C

Description
State of origin for first time students

Usage
make_ef1_part_C(df)

Arguments
df  A dataframe of student/degree information

Value
A text file
**Make Fall Enrollment Part D**

**Description**
Count of new non-degree students

**Usage**

```r
make_ef1_part_D(df)
```

**Arguments**

- `df`
  A dataframe of student/degree information

**Value**

A text file

---

**Make Fall Enrollment Part E**

**Description**
Retention counts

**Usage**

```r
make_ef1_part_E(df)
```

**Arguments**

- `df`
  A dataframe of student/degree information

**Value**

A text file
make_ef1_part_F  
*Make Fall Enrollment Part F*

**Description**

Student Faculty Ratio

**Usage**

```
make_ef1_part_F(df)
```

**Arguments**

- `df`  
  A dataframe (either "students" or "retention") as a unitid source

**Value**

A text file

---

make_ef1_part_G  
*Make Fall Enrollment Part G*

**Description**

Distance Ed counts

**Usage**

```
make_ef1_part_G(df)
```

**Arguments**

- `df`  
  A dataframe of student/degree information

**Value**

A text file


### make_ef1_part_H

**Make Fall Enrollment Part H (gender details)**

**Description**

Make Fall Enrollment Part H (gender details)

**Usage**

```r
make_ef1_part_H(df, ugender, ggender)
```

**Arguments**

- `df`: A dataframe of student enrollment information
- `ugender`: A boolean: TRUE means you are collecting and able to report "another gender" for undergraduate completers, even if you have no (or few) such students. Set as FALSE if necessary
- `ggender`: A boolean: TRUE means you are collecting and able to report "another gender" for graduate completers, even if you have no (or few) such students. Set as FALSE if necessary

**Value**

A text file

### make_gr200

**Make Graduation Rates 200**

**Description**

Make Graduation Rates 200

**Usage**

```r
make_gr200(df)
```

**Arguments**

- `df`: A dataframe of student/degree information

**Value**

A text file
Description
Make Graduation Rates Part B

Usage
make_gr_part_B(df)

Arguments
df A dataframe of student/degree information

Value
A text file

Description
Make Graduation Rates Part C

Usage
make_gr_part_C(df)

Arguments
df A dataframe of student/degree information

Value
A text file
**make_gr_part_E**  
*Make Graduation Rates Part E (gender details)*

**Description**
Make Graduation Rates Part E (gender details)

**Usage**
make_gr_part_E(df, ugender)

**Arguments**
- **df**: A dataframe of student/degree information for unduplicated undergraduates
- **ugender**: A boolean: TRUE means you are collecting and able to report "another gender" for undergraduate students, even if you have no (or few) such students. Set as FALSE if necessary

**Value**
A df aggregated for the survey part

---

**make_hr_part_A1**  
*Make Human Resources Part A1*

**Description**
Part A1 — COUNT of FT INSTRUCTIONAL staff by tenure status, academic rank, and race/ethnicity/gender

**Usage**
make_hr_part_A1(df)

**Arguments**
- **df**: a dataframe

**Value**
a txt file
**make_hr_part_A2**

*Make Human Resources Part A2*

**Description**

Part A2 — COUNT of FT instructional staff by tenure status, medical school, and function

**Usage**

`make_hr_part_A2(df)`

**Arguments**

- `df` a dataframe

**Value**

a txt file

---

**make_hr_part_B1**

*Make Human Resources Part B1*

**Description**

HR Part B1 — COUNT of FT Non-instructional staff by occupational category

**Usage**

`make_hr_part_B1(df)`

**Arguments**

- `df` a dataframe

**Value**

a txt file
**make_hr_part_B2**  
*Make Human Resources Part B2*

**Description**

Part B2 — Full-time non-instructional staff by tenure, medical school, and occupational category

**Usage**

```r
make_hr_part_B2(df)
```

**Arguments**

- `df`  
  a dataframe

**Value**

a txt file

---

**make_hr_part_B3**  
*Make Human Resources Part B3*

**Description**

Part B3 — Full-time non-instructional staff by medical school, and occupational category

**Usage**

```r
make_hr_part_B3(df)
```

**Arguments**

- `df`  
  a dataframe

**Value**

a txt file
make_hr_part_D1  Make Human Resources Part D1

Description

Part D1 — Part-time staff by occupational category

Usage

make_hr_part_D1(df)

Arguments

df  a dataframe

Value

a txt file

make_hr_part_D2  Make Human Resources Part D2

Description

Part D2 — Graduate assistants by occupational category and race/ethnicity/gender

Usage

make_hr_part_D2(df)

Arguments

df  a dataframe

Value

a txt file
**make_hr_part_D3**

*Make Human Resources Part D3*

**Description**

Part D3 — Part-time staff by tenure, medical school, and occupational category

**Usage**

```python
make_hr_part_D3(df)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df</td>
<td>a dataframe</td>
</tr>
</tbody>
</table>

**Value**

a txt file

---

**make_hr_part_D4**

*Make Human Resources Part D4*

**Description**

Part D4 — Part-time Non-instructional staff by medical school, and occupational category

**Usage**

```python
make_hr_part_D4(df)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>df</td>
<td>a dataframe</td>
</tr>
</tbody>
</table>

**Value**

a txt file
**Make Human Resources Part G1**

**Description**

Part G1 — Salaries of INSTRUCTIONAL staff

**Usage**

`make_hr_part_G1(df)`

**Arguments**

- **df** a dataframe

**Value**

a txt file

---

**Make Human Resources Part G2**

**Description**

Part G2 — Salaries of non-instructional staff

**Usage**

`make_hr_part_G2(df)`

**Arguments**

- **df** a dataframe

**Value**

a txt file
**Description**

Part H1 — Full-time new hire instructional staff by tenure status and race/ethnicity/gender

**Usage**

make_hr_part_H1(df)

**Arguments**

- **df**: a dataframe

**Value**

- a txt file

---

**Description**

Part H2 — New hires by occupational category, Race/Ethnicity/Gender

**Usage**

make_hr_part_H2(df)

**Arguments**

- **df**: a dataframe

**Value**

- a txt file
**make_om_part_A**  
*Make Outcome Measures Part A*

**Description**

Establishing the Outcome Measures cohorts

**Usage**

```r
make_om_part_A(df)
```

**Arguments**

df  
A dataframe of student statuses

**Value**

A text file ready for IPEDS upload

---

**make_om_part_B**  
*Make Outcome Measures Part B*

**Description**

Award Status at Four Years after Entry

**Usage**

```r
make_om_part_B(df)
```

**Arguments**

df  
A dataframe of student statuses

**Value**

A text file ready for IPEDS upload
**make_om_part_C**

Make Outcome Measures Part C

**Description**

Award Status at Six Years after Entry

**Usage**

`make_om_part_C(df)`

**Arguments**

- `df` A dataframe of student statuses

**Value**

A text file ready for IPEDS upload

**make_om_part_D**

Make Outcome Measures Part D

**Description**

Award Status and Enrollment at Eight Years after Entry

**Usage**

`make_om_part_D(df)`

**Arguments**

- `df` A dataframe of student statuses

**Value**

A text file ready for IPEDS upload
om_students  
**Dummy data for Outcome Measures functions**

**Description**
Contains sample values for students

**Usage**

```
om_students
```

**Format**
A data frame with 20 rows and 9 columns

**Details**
See complete information by running `?create_dummy_data_om.R`

---

prep_com_data_frame  
**Some initial recoding for Completions**

**Description**
Some initial recoding for Completions

**Usage**

```
prep_com_data_frame(df)
```

**Arguments**

```
df a dataframe of student level data or cip information
```

**Value**

```
df
```
**Description**
Some initial recoding for Fall Enrollment

**Usage**

```r
prep_ef1_data_frame(df)
```

**Arguments**

- `df` a dataframe of student level data

**Value**

`df`

---

**Description**
Some initial recoding for Human Resources

**Usage**

```r
prep_hr_data_frame(df)
```

**Arguments**

- `df` a dataframe

**Value**

a dataframe
prep_om_awards

Set up extra_awards df for Outcome Measures part B, C, D

Description
Select correct year, ensure all award levels end up with a column

Usage
prep_om_awards(df, award)

Arguments
- df: A dataframe of student statuses
- award: A string with the df column to use for processing depending on the OM part

Value
A df ready for use in the make_om_part functions B-D

prep_om_data_frame

Some initial recoding for OutcomeMeasures

Description
Some initial recoding for OutcomeMeasures

Usage
prep_om_data_frame(df)

Arguments
- df: A dataframe of student level data

Value
df ready for om report scripts
produce_com_report  

Shortcut function with all steps to provide a Completions report

Description

Shortcut function with all steps to provide a Completions report

Usage

```r
produce_com_report(
  df,
  extracips = NULL,
  part = "ALL",
  format = "uploadable",
  ugender = TRUE,
  ggender = TRUE
)
```

Arguments

- **df**: A dataframe set up according to the readme
- **extracips**: A dataframe set up according to the readme (optional)
- **part**: A string with what part of the report you want to produce: 'all', 'A', etc.
- **format**: A string ("uploadable" will produce a properly formatted upload file. "readable" will produce a csv of the upload file (only works for one part at a time). "both" will provide both options, but only works with one part at a time.
- **ugender**: A boolean: TRUE means you are collecting and able to report "another gender" for undergraduate completers, even if you have no (or few) such students. Set as FALSE if necessary
- **ggender**: A boolean: TRUE means you are collecting and able to report "another gender" for graduate completers, even if you have no (or few) such students. Set as FALSE if necessary

Value

A txt or csv file at the path of your choice

Examples

```r
# entire report
produce_com_report(com_students, com_cips)

# one part as csv instead of key-value
produce_com_report(com_students, com_cips, part = "A", format = "readable")
```
produce_e1d_report  Shortcut function with all steps to provide a 12 Month Enrollment report

Description

Shortcut function with all steps to provide a 12 Month Enrollment report

Usage

produce_e1d_report(
    df,
    hrs,
    part = "ALL",
    format = "uploadable",
    ugender = TRUE,
    ggender = TRUE
)

Arguments

df  A dataframe set up according to the readme for students
hrs  A dataframe set up according to the readme for instructional activity
part  A string with what part of the report you want to produce: ‘all’, ‘A’, etc.
format  A string ("uploadable" will produce a properly formatted upload file. "readable" will produce a csv of the upload file (only works for one part at a time). "both" will provide both options, but only works with one part at a time.
ugender  A boolean: TRUE means you are collecting and able to report "another gender" for undergraduate students, even if you have no (or few) such students. Set as FALSE if necessary
ggender  A boolean: TRUE means you are collecting and able to report "another gender" for graduate students, even if you have no (or few) such students. Set as FALSE if necessary

Value

A txt or csv file at the path of your choice
Examples

# entire report
produce_e1d_report(e1d_students, e1d_instr)

# one part, as csv instead of key-value file
produce_e1d_report(e1d_students, part = "A", format = "readable")

produce_ef1_report  
Shortcut function with all steps to provide a Fall Enrollment report

Description

Shortcut function with all steps to provide a Fall Enrollment report

Usage

produce_ef1_report(
    students,
    retention,
    part = "ALL",
    include_optional = FALSE,
    format = "uploadable",
    ugender = TRUE,
    ggender = TRUE
)

Arguments

students  A dataframe set up according to the readme with student data
retention A dataframe set up according to the readme with retention data
part  A string with what part of the report you want to produce: 'all', 'A', etc.
include Optional  A boolean flag for whether optional parts should be included
format  A string ("uploadable" will produce a properly formatted upload file. "readable" will produce a csv of the upload file (only works for one part at a time). "both" will provide both options, but only works with one part at a time.
ugender  A boolean: TRUE means you are collecting and able to report "another gender" for undergraduate completers, even if you have no (or few) such students. Set as FALSE if necessary
ggender  A boolean: TRUE means you are collecting and able to report "another gender" for graduate completers, even if you have no (or few) such students. Set as FALSE if necessary
produce_gr200_report

Value

A txt or csv file at the path of your choice

Examples

#entire report
produce_ef1_report(ef1_students, ef1_retention)

#entire report with optional sections
produce_ef1_report(ef1_students, ef1_retention, include_optional = TRUE)

#one part as csv instead of key-value
produce_ef1_report(ef1_students, part = 'D', format = 'readable')

produce_gr200_report  Shortcut function with all steps to provide a Grad Rates 200 report

Description

Shortcut function with all steps to provide a Grad Rates 200 report

Usage

produce_gr200_report(df, format = "uploadable")

Arguments

df  a dataframe set up according to the readme
format  A string ("uploadable" will produce a properly formatted upload file. "readable" will produce a csv of the upload file (only works for one part at a time). "both" will provide both options, but only works with one part at a time.

Value

A txt or csv file at the path of your choice

Examples

#entire report
produce_gr200_report(gr200_students)
produce_gr_report

Shortcut function with all steps to provide a Graduation Rates report

Description

Shortcut function with all steps to provide a Graduation Rates report

Usage

produce_gr_report(df, part = "ALL", format = "uploadable", ugender = TRUE)

Arguments

df  a dataframe set up according to the readme
part a string with what part of the report you want to produce "all", "A1", etc.
format A string ("uploadable" will produce a properly formatted upload file. "readable"
 will produce a csv of the upload file (only works for one part at a time). "both"
 will provide both options, but only works with one part at a time.
ugender A boolean: TRUE means you are collecting and able to report "another gender"
 for undergraduate students, even if you have no (or few) such students. Set as
 FALSE if necessary

Value

A txt or csv file at the path of your choice

Examples

#entire report
produce_gr_report(gr_students)

#one part in csv format instead of key-value
produce_gr_report(gr_students, part = "B", format = "readable")
produce_hr_report  
*Shortcut function with all steps to provide a Human Resources report*

**Description**

Shortcut function with all steps to provide a Human Resources report

**Usage**

```
produce_hr_report(df, part = "all", format = "uploadable")
```

**Arguments**

- **df**
  - a dataframe set up according to the readme
- **part**
  - a string with what part of the report you want to produce "all", "A1", etc.
- **format**
  - A string ("uploadable" will produce a properly formatted upload file. "readable" will produce a csv of the upload file (only works for one part at a time). "both" will provide both options, but only works with one part at a time.)

**Value**

A txt or csv file at the path of your choice

**Examples**

```
#entire report
produce_hr_report(hr_staff)

#subsection with csv output instead of key-value txt
produce_hr_report(hr_staff, part = "A1", format = "readable")
```

produce_om_report  
*Shortcut function with all steps to provide an Outcome Measures report*

**Description**

Shortcut function with all steps to provide an Outcome Measures report

**Usage**

```
produce_om_report(df, part = "ALL", format = "uploadable")
```
produce_other_report

Arguments

- **df**: A dataframe set up according to the readme
- **part**: A string with what part of the report you want to produce: 'all', 'A', etc.
- **format**: A string ("uploadable" will produce a properly formatted upload file. "readable" will produce a csv of the upload file (only works for one part at a time). "both" will provide both options, but only works with one part at a time.

Value

A txt or csv file at the path of your choice

Examples

```r
# entire report
produce_om_report(om_students)

# one part with csv output instead of key-value
produce_om_report(om_students, part = 'A', format = 'readable')
```

---

produce_other_report  Produce an upload-compatible txt file from pre-aggregated files

Description

Use this function to create a key-value pair uploadable file from your own prepared dataframes, instead of using a different (provided) produce function. Your dataframes must be prepped to match final submission requirements as laid out by IPEDS (or whatever survey you will use this for). Use this function for one survey at a time, and add a separate dataframe for each part to the ... argument. See vignette for more details.

Usage

```r
produce_other_report(..., survey = "MySurvey", part = "AllParts")
```

Arguments

- **...**: dataframes (one for each survey part, in order)
- **survey**: string with the survey name you’d like in your filename
- **part**: string with the part name (subname) you’d like your file name

Value

A txt file on your computer with the title [survey]_[part]_[today’s date].txt
Note
You must name the arguments for survey and part if using non-default value. If the arguments are unnamed, the function will assume their values are additional dataframes.

Examples

```r
#With built-in R data
produce_other_report(mtcars[1:5,], iris[1:5,], ToothGrowth[1:5,], survey = 'FakeSurvey')
```

```r
#Will not execute properly (argument unnamed)
produce_other_report(mtcars[1:5,], iris[1:5,], ToothGrowth[1:5,], 'FakeSurvey')
```

---

**set_report_path**  
Set the path for where the reports will be saved to.

### Description
Set the path for where the reports will be saved to.

### Usage
```r
set_report_path()
```

### Value
`path`

---

**specs_COM**  
Table of data requirements for Completions starting dataframe

### Description
Table of data requirements for Completions starting dataframe

### Usage
```r
specs_COM
```

### Format
A data frame with 21 rows and 4 columns
specs_E1D

Table of data requirements for 12 Month Enrollment starting dataframes

Description
Table of data requirements for 12 Month Enrollment starting dataframes

Usage
specs_E1D

Format
A data frame with 18 rows and 4 columns

specs_EF1

Table of data requirements for Fall Enrollment starting dataframes

Description
Table of data requirements for Fall Enrollment starting dataframes

Usage
specs_EF1

Format
A data frame with 23 rows and 4 columns

specs_GR

Table of data requirements for Graduation Rates starting dataframe

Description
Table of data requirements for Graduation Rates starting dataframe

Usage
specs_GR

Format
A data frame with 14 rows and 3 columns
Table of data requirements for Grad Rates 200 starting dataframe

Description
Table of data requirements for Grad Rates 200 starting dataframe

Usage
specs_GR200

Format
A data frame with 5 rows and 3 columns

Table of data requirements for HR starting dataframe

Description
Table of data requirements for HR starting dataframe

Usage
specs_HR

Format
A data frame with 13 rows and 3 columns

Table of data requirements for OM starting dataframe

Description
Table of data requirements for OM starting dataframe

Usage
specs_OM

Format
A data frame with 9 rows and 3 columns
**write_report**

Write the prepared data to a txt file in key-value format

**Description**

Write the prepared data to a txt file in key-value format

**Usage**

```python
write_report(..., survey, part, output_path)
```

**Arguments**

- `...`: dataframes (one for each survey part, in order)
- `survey`: a string (which [IPEDS] survey)
- `part`: a string (which upload part of the survey)
- `output_path`: a file path (where the file should be saved)

**Value**

a txt file (at the path location)

**Note**

All arguments for this function are required and must be named. Dataframes must have the key as the column name (with appropriate capitalization) and the value in the cells

---

**write_report_csv**

Write the prepared data to a csv file

**Description**

Write the prepared data to a csv file

**Usage**

```python
write_report_csv(df, survey, part, output_path)
```

**Arguments**

- `df`: a dataframe (prepared via the 'make' scripts)
- `survey`: a string (which IPEDS survey)
- `part`: a string (which upload part of the survey)
- `output_path`: a path (which folder the report should go in)
Value

a csv file (at the path location)

Note

All arguments for this function are required. The dataframe must have the key as the column name (with appropriate capitalization) and the value in the cells
Index

* datasets
  com_cips, 4
  com_students, 5
  e1d_instr, 10
  e1d_students, 11
  efl_retention, 11
  efl_students, 12
  gr200_students, 13
  gr_students, 13
  hr_staff, 14
  om_students, 34
  specs_COM, 44
  specs_E1D, 45
  specs_EF1, 45
  specs_GR, 45
  specs_GR200, 46
  specs_HR, 46
  specs_OM, 46

apply_upload_format, 4

com_cips, 4
com_students, 5
create_dummy_data_com, 5
create_dummy_data_e1d, 6
create_dummy_data.ef1, 7
create_dummy_data.gr, 8
create_dummy_data.gr200, 8
create_dummy_data.hr, 9
create_dummy_data.om, 10

e1d_instr, 10

e1d_students, 11
efl_retention, 11
efl_students, 12

get_ipeds_unitid, 12
gr200_students, 13
gr_students, 13
hr_staff, 14

IPEDSuploadables, 14
IPEDSuploadables-package
  (IPEDSuploadables), 14

make_com_part_A, 14
make_com_part_B, 15
make_com_part_C, 15
make_com_part_D, 16
make_com_part_E, 16
make_e1d_part_A, 17
make_e1d_part_B, 17
make_e1d_part_C, 18
make_e1d_part_D, 18
make_e1d_part_E, 19
make_efl_part_A, 19
make_efl_part_B, 20
make_efl_part_C, 20
make_efl_part_D, 21
make_efl_part_E, 21
make_efl_part_F, 22
make_efl_part_G, 22
make_efl_part_H, 23
make_gr200, 23
make_gr_part_B, 24
make_gr_part_C, 24
make_gr_part_E, 25
make_hr_part_A1, 25
make_hr_part_A2, 26
make_hr_part_B1, 26
make_hr_part_B2, 27
make_hr_part_B3, 27
make_hr_part_D1, 28
make_hr_part_D2, 28
make_hr_part_D3, 29
make_hr_part_D4, 29
make_hr_part_G1, 30
make_hr_part_G2, 30
make_hr_part_H1, 31
make_hr_part_H2, 31
make_om_part_A, 32
make_om_part_B, 32
make_om_part_C, 33
make_om_part_D, 33
om_students, 34
prep_com_data_frame, 34
prep_ef1_data_frame, 35
prep_hr_data_frame, 35
prep_om_awards, 36
prep_om_data_frame, 36
produce_com_report, 37
produce_e1d_report, 38
produce_ef1_report, 39
produce_gr200_report, 40
produce_gr_report, 41
produce_hr_report, 42
produce_om_report, 42
produce_other_report, 43
set_report_path, 44
specs_COM, 44
specs_E1D, 45
specs_EF1, 45
specs_GR, 45
specs_GR200, 46
specs_HR, 46
specs_OM, 46
write_report, 47
write_report_csv, 47