Package 'opa'

June 21, 2021

Type Package

Title Office of Planning & Analysis Human Resource Utilties

Version 1.1.0

Author Ian C Johnson

Maintainer Ian C Johnson <ian.johnson2@montana.edu>

Description General utility functions supporting Montana State University employee data extraction, transformation, and analysis.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

R topics documented:

`add_earn_codes` *Determine earn codes used for a set of jobs and given date*

Description

Determine the earn codes and associated earn code descriptions for a set of jobs on a specific date. This is particularly useful for adcomp jobs whose 'reason' is stored in the earn code description field.

Usage

```
add_earn_codes(job_key, ee_id_type = "PIDM", post_date, opt_bann_conn)
```

Arguments

`job_key`	a vector of job keys, defined as the GID or PIDM concatenated with Position Number and Suffix. Used as unique table keys for most Banner Payroll Tables.
`ee_id_type`	one of "PIDM" or "GID" used to specify the employee id used to create the job_key.
`post_date`	the as-of date. any payroll data processed after this date will be ignored.
`opt_bann_conn`	an optional banner connection object. If none is provided, the function will prompt for Banner login details and create a one-time use Banner connection.

Value

a dataframe containing PIDM, pidm-posn-suff job key, earn code and descriptions for the most recent payroll covering the as_of date.

add_factor_levels *Add factor level to a specified dataframe column*

Description

Add factor level to a specified dataframe column

Usage

```
add_factor_levels(x_col, new_level = "")
```

Arguments

x_col	the column of data in which the factor level will be added. If not already a factor, will return the column unaltered
new_level	the character value of the new factor level. Commonly NA or "".

Value

a column of data with teh new factor level added if the supplied vector of data was a factor

add_flsa_exmpt_status *Add a job's FLSA exemption status to a dataframe*

Description

Using a choosen Ecls column, map elcs to FLSA overtime exemption status. Depends on the 'msuopa::ecls_flsa_exmpt_tbl' dataframe being accessible to the package

Usage

```
add_flsa_exmpt_status(df, ecls_col_name)
```

Arguments

df	the dataframe containing the ecls column and to which the new column will be added.
ecls_col_name	The string vector containing the name of the eclass column to which the FLSA status will be mapped.

Value

The original dataframe with a new column named `FLSA OT Exempt`.

add_longevity_perc *Calculate Longevity Salary Bonus Percent*

Description

Given a column containing current hire date and an as_of date, calculate the appropriate percent bonus

Usage

```
add_longevity_perc(
  df,
  curr_hire_col = CURRENT_HIRE_DATE,
  as_of_date_col = date,
  job_group_col
)
```

Arguments

df	the dataframe containing employee records
curr_hire_col	the column containing current hire dates as POSIXct
as_of_date_col	the column containing the as_of_date as POSIXct
job_group_col	the column containing the row type, i.e. classified, proessional, faculty, etc. see msupa::classify_job_detailed

Value

a dataframe with an additional longevity_perc_bonus column

allee_dates_from_fnames *allee_dates_from_fnames*

Description

Extracts and formats the dates contained in the csv file names. The csv files must be in the form "YYYYMMDD All Employees.txt".

Usage

```
allee_dates_from_fnames(file_list)
```

Arguments

file_list	a character vector containing the names of all the txt files in the folder containing csv

Value

list of POSIXct dates contained in the input filenames

all_ee_col_types *All Employees Report Column Types*

Description

A named list of vectors of column names specifying column types as numeric or character. Additional date formating is necessary after loading. This column specification ensures that leading zeros are never dropped from certain fields such as GID or Zip Code.

Usage

```
all_ee_col_types(date)
```

Arguments

date	the date on which the All Employees report was run from Banner. This is necessary because the columns were expanded on 12/15/2017

Value

a named list of vectors assigning each column to a class by column name.

bann_form_tbl_xwalk *Banner Forms-Tables Crosswalk dataset*

Description

A list of all table datasources used to populate a given banner form.

Usage

```
bann_form_tbl_xwalk()
```

Value

a dataframe comprised of FORM NAME, FORM DESC, TABLE ACCESSED and TABLE MODULE

bann_tbl_form_xwalk *Banner Tables-Forms Crosswalk dataset*

Description

A list of all forms using a given table as a datasource

Usage

```
bann_tbl_form_xwalk()
```

Value

a dataframe comprised of TABLE, FORM NAME, and FORM DESC

build_cip_hier *build the CIP hierarchy of 6-digit, 4-digit, and 2-digit values*

Description

Draft - requires debugging

Usage

```
build_cip_hier(fpath = "X:/icj_dts/opaDataEE/raw/CIPCode2010.csv")
```

Arguments

fpath	the path to the raw CIPCode values.

Value

the cip hierarchy with each 6 digit cip including it's 4 and 2 digit rollups. Defaults to a wide dataset

build_emr_dataset *Union & Join all DMHS datafiles*

Description

Load, Union, and Join all dmhs files found in "X:/Employees/EMR Report (production)/DM csv files/". Save single csv file into the same folder as "full_dmhs_data"

Usage

```
build_emr_dataset(min_date, max_date)
```

Value

A .csv file containing the joined and unioned base, jobs datasets

build_new_df *Build a dataframe to store the orgn hierarchy*

Description

Build a dataframe to store the orgn hierarchy

Usage

```
build_new_df(
  key_vector,
  n_cols = 9,
  predicate_char = "Org",
  include_max_depth = T,
  sort_keys = T
)
```

Arguments

key_vector	the vector to use as the key values for hte new dataframe. This will be renamed 'seed' column.
n_cols	the number of columns to be added
predicate_char	the predicate for the column names
include_max_depth	a boolean specifying if a column should be added with the name 'max_depth' as used in the build_org_hierarchy function
sort_keys	a boolean specifying if the seed column should be sorted before return

Value

a dataframe containing a column of key seed values with empty columns specified by n_cols, predicate char, and include_max_depth

build_org_hierarchy *Build an unstandardized Org Hierarchy*

Description

The ftvorgn table uses FTVORGN_ORGN_CODE and FTVORGN_PRED_CODE to form hierarchical relationship between organization numbers. For each unique org code, i.e. row, follow the predecessor codes until it reaches the top of the organization. Requires the format_org_df to be useable outside of the package.

Usage

```
build_org_hierarchy(ftvorgn_data, new_col_name)
```

Arguments

ftvorgn_data	the ftvorgn corresponding to a single as-of date
new_col_name	the predicate for the new columns i.e. org, orgn, etc...

Value

the unformatted hierarchy with each row including a seed value and the corresponding org codes leading to the top of hte hierarchy as determined by the PRED code.

---

build_org_hierarchy_lu

Build the FTVORGN Hierarchy for each unique Organization Code/Number

---

Description

Build a lookup table storing each unique org code's hierarchy wihtin the BZ organization. BZ campus filters are currently hardcoded. Will break if the org hierarchy depth ever exceeds 8

Usage

```
build_org_hierarchy_lu(
  as_of_date = Sys.Date(),
  opt_bann_conn,
  opt_ftvorgn_data,
  include_names = T,
  new_col_name = "Org"
)
```

Arguments

as_of_date	the date used to filter the FTVORGN data to ensure that historical or future records are not used. Defaults to Sys.Date()
opt_bann_conn	an optional banner connection object
opt_ftvorgn_data	an optional ftvorgn dataframe in the format supplied by msoupa::get_ftvorgn_data. If no supplied, will pull.
include_names	a boolean specifying whether to join org titles to the output dataframe
new_col_name	a predicate to use on teh new column names. Defaults to 'Org' creating columns titled 'Org1', 'Org2',... 'Org7'

Value

a dataframe containing a 'seed' column containing the original org code used to build the hierarchy. Hierarchy is specified from least to most granular. Org1 is always 400000, i.e. "montana state university -bozeman".Org2 then represents the Division, Org3 college, Org4 dept, etc. If titles are included, each column includes an additional Orgx_desc, seed_desc, ... column containing the unit's name

build_osu_hierarchy *Build OSU CIP hierarchy*

Description

For each CIP-Rank combination in the OSU salary datafile, compile the 2-digit, four-digit cip code associated with each cip

Usage

```
build_osu_hierarchy(year = 2019)
```

Arguments

year	the year (fall data) corresponding to the osu salary survey. 2019 refers to data submitted in fall of 2019.

Value

a dataframe containing the cip code n-counts and average salary for the cip-rank combination at the 2, 4, 6 digit cip codes.

calc_agg_fund_type *Classify job labor distribution rows into aggregate categories*

Description

Using the index and program codes to determine 'fund source' for a job. The categories defined by the following regular expressions are verified yearly wiht the business/finance office. last updated Fall 2019

Usage

```
calc_agg_fund_type(
  df,
  index_col_name = FUNDING_INDX,
  program_col_name = FUNDING_PROG,
  percent_fund_col_name = FUNDING_PERCENT,
  consolidate_rows = T,
  consolidation_key_col_name = job_date_key,
  date_col_name = as_of_date
)
```

Arguments

df	the dataframe which contains index and program data
index_col_name	the unquoted name of the index column
program_col_name	the unquoted name of the program column
percent_fund_col_name	the unquoted name of the column containing the percent of the total job funding associated with the index, program data.
consolidate_rows	return a single row per the consolidation key
consolidation_key_col_name	the unique identifier used to simplify the dataset if consolidate_rows is True
date_col_name	the field which specifies the as of date for the nbrjlbd data entered from the dataframe. typically 'as_of_date' or simply 'date'

Value

a dataframe containing new boolean columns indicating the fund source type.

`check_fix_dupe_name` *check_fix_dupe_name checks a string vector for the existence of a particular string. If found, it modifies the query string so that it does not match an existing string int he vector. It does this be appending an underscore and numeric value.*

Description

check_fix_dupe_name checks a string vector for the existence of a particular string. If found, it modifies the query string so that it does not match an existing string int he vector. It does this be appending an underscore and numeric value.

Usage

```
check_fix_dupe_name(name, curr_names)
```

Arguments

name	the single string name that will be searched for and modified if necessary
curr_names	the vector of strings that will be searched for the single string 'name'.

Value

if necessary, a modified string that is not duplicated in the input vector. otherwise, the input name parameter

classify_adcomp_type *Classify Additional Compensation into categories*

Description

Use eclass, position number, suffix and eclass to classify adcomp into one of the following categories: 'Adcomp', 'Stipend', 'One-time Payment', Overload/Overtime and 'Car Allowance/Other'

Usage

```
classify_adcomp_type(
  df,
  posn_col_name = POSN,
  suff_col_name = SUFF,
  job_title_col_name = JOB_TITLE,
  ecls_col_name = ECLS_JOBS
)
```

Arguments

df	the dataframe whose adcomp rows will be categorized
posn_col_name	the unquoted column name containing position numbers
suff_col_name	the unquoted column name containing job suffix
job_title_col_name	the unquoted column name containing job titles
ecls_col_name	the unquoted column name containing job or person eclass

Value

a dataframe with an additional 'adcomp_type' column specifying the type

classify_job *Classify Job into aggregated groups*

Description

Use a combination of Eclass, Suffix, position number, rank, tenure, position title, to group like jobs. Categories include Classified, Professional, Executive, Temporary, Fixed-Term, Faculty NTT, Faculty TT/T and Additional Compensation. Depends on classify_job_detailed and classify_job_by_ecls.

Usage

```
classify_job(
  df,
  pidm_col_name = PIDM,
  posn_col_name = POSN,
  suff_col_name = SUFF,
  ecls_col_name = ECLS_JOBS,
```

```
  jobtitle_col_name = JOB_TITLE,
  posntitle_col_name = POSITION_TITLE,
  date_col_name = date,
  opt_rank_records,
  opt_tenure_records,
  opt_bann_conn
)
```

Arguments

df	the data frame containing the required variables (PIDM, Position Number, Suffix, Job Title, Position Title, Eclass, Rank, Tenure) if missing rank, tenure, or position title, will derive from Banner based on the supplied PIDM
pidm_col_name	the unquoted name of the column containing PIDM values
posn_col_name	the unquoted name of the column containing Position Number values
suff_col_name	the unquoted name of the column containing Suffix values
jobtitle_col_name	the unquoted name of the column containing Job Title values
posntitle_col_name	the unquoted name of the column containing Position Title values
date_col_name	the column containing the as-of date used for pulling historical rank/tenure info
opt_rank_records	all rank records from PERRANK table use opa::get_rank_records()
opt_tenure_records	all tenure records from banner use opa::get_tenure_records()
opt_bann_conn	an optional banner connection object

Value

the original df with an additional column 'job_type' containing the new data

Author(s)

Ian C Johnson

See Also

classify_job_detailed classify_job_by_ecls

---

classify_job_by_ecls *classify the type of job based on it's eclass*

Description

using the elcass, determine the type of job. Does not handle situations where the job type is not properly contained in th eclass. examples includes NTT vs TT/T and some types of non-jobpayment.

Usage

```
classify_job_by_ecls(df, ecls_col_name, new_col_name = job_type)
```

Arguments

df	the dataframe containing the eclass and to which the new column will be appended
ecls_col_name	the unquoted name of the eclass column
new_col_name	the unquoted name of the new column to be added

Value

the original datafram with a 'job-type' column added

---

classify_job_detailed *Classify Job into aggregated categories*

Description

Group like jobs into common categories such as Classified, Professional, Temporary, Fixed-Term, Faculty TT/T, etc. Supports tidyr quasiquotation. All input parameters default to snapshot names. Uses classify_job_by_ecls as the primary function to assign job-types. Some categories, such as faculty and non-job payments, require additional non-ecls logic found in this function

Usage

```
classify_job_detailed(
  df,
  posn_number_col_name = POSN,
  suffix_col_name = SUFF,
  job_title_col_name = JOB_TITLE,
  posn_title_col_name = POSITION_TITLE,
  ecls_col_name = ECLS_JOBS,
  rank_col_name = rank_code,
  tenure_col_name = tenure_code,
  new_col_name = job_type
)
```

Arguments

df	the dataframe containing job rows to be categorized
posn_number_col_name	the unquoted name of the position number column
suffix_col_name	the unquoted name of the suffix number column
job_title_col_name	the unquoted name of the job title column
posn_title_col_name	the unquoted name of the job title column
ecls_col_name	the unquoted name of the ecls column
rank_col_name	the unquoted name of the rank column containing numeric aggregated ranks (1,2,3,4...)
new_col_name	the unquoted name of the new column to be added
eskl_col_name	the unquoted name of the column containing e-skill

Value

a dataframe containing an additional column, as specified by the new_col_name parameter.

Author(s)

Ian C Johnson

See Also

classify_job, classify_job_by_ecls

`compile_adcomp_stipend_totals` *Compile total amount paid via adcomp and stipend payments per person per fy*

Description

determine adcomp and stipend amount group by job, person, dept or some other variable. parameters should be passed unquoted. All input parameters use quasiquotation.

Usage

```
compile_adcomp_stipend_totals(
  df,
  unique_id_field = PIDM,
  posn_field_name = POSN,
  suff_field_name = SUFF,
  wage_month_field_name = MONTHLY_RATE,
  months_field_name = MONTHS
)
```

Arguments

`df`	the dataframe containing job records with adcomp/stipend records
`unique_id_field`	the grouping of the adcomp and stipend amounts. Typically gid/pidm, posn, or some dept
`posn_field_name`	the name of the field used to store the position number. Complies with tidyr programming best-practice. Should be passed unquoted
`suff_field_name`	the name of the field used to store the suffix. Complies with tidyr programming best-practice. Should be passed unquoted

Value

a dataframe containing one row per distinct grouping variable which contains a non-zero adcomp or stipend sum.

`compile_osu_msu_pivot` *Compile the OSU-MSU salary report with appropriate formatting. Requires use of an excel template prior to publication. See OSU_MSU_post_19F.xlsx*

Description

Compile the OSU-MSU salary report with appropriate formatting. Requires use of an excel template prior to publication. See OSU_MSU_post_19F.xlsx

Usage

```
compile_osu_msu_pivot(
  raw_osu_df,
  msu_cips_only = F,
  fpathname_out = "./OSU_MSU_20F_post_all_cips"
)
```

Arguments

`raw_osu_df`	the dataframe of unformatted/unfiltered DATAFEED data supplied by 'import_raw_osu_data' function
`msu_cips_only`	a boolean to include if an additional dataset of CIPs related to MSU cips should be included
`fpathname_out`	the excel document name to export.

Value

a list containing the formatted OSU data for all cips, optionally the formatted OSU data for MSU related cips, and the raw unformatted OSU dataset

`compute_fiscal_year` *Compute the Fiscal Year given a date*

Description

Computes the fiscal year of a vector of dates based on the Montana State fiscal calendar (July 1 - June 30).

Usage

```
compute_fiscal_year(date)
```

Arguments

`date`	the vector of dates from which to compute fiscal year

Value

the vector of year integers

Author(s)

Ian C Johnson

Examples

```
dte_1 <- as.Date("2018-01-01")
dte_2 <- as.Date("2018-08-01")
compute_fiscal_year(dte_1)
compute_fiscal_year(dte_2)
```

---

compute_n_workdays *Estimate the number of workdays in a given date range.*

Description

Assumes that workdays is the standard Monday-Friday work week excluding holidays. See ?msuopa::msu_holidays for more information about the holidays used in computation.

Usage

```
compute_n_workdays(date_start, date_end, holidays = msuopa::msu_holidays)
```

Arguments

date_start	the start date as a character or date
date_end	the end date as a character or date
holidays	a vector containing msu designated holidays as Date objects.

Value

a single integer quantifying the number of workdays between date_start and date_end, excluding holidays and weekends.

Author(s)

Ian C Johynson

---

contact_sql_pull *Helper function for pull_contact_info.*

Description

Not to be exported.

Usage

```
contact_sql_pull(pidm_vec_in, bann_conn)
```

Arguments

pidm_vec_in	a vector of pidms not to exceed 1k items in length.
bann_conn	banner connection object

`convert_allee_txt_rds` *convert_allee_txt_rds*

Description

Convert a folder path containing .txt semi-colon delimited all employees report(s) from Report Web into RDS files.

Usage

```
convert_allee_txt_rds(folder_path, opt_output_path)
```

Arguments

`folder_path`	the folder containing .txt all employees reports

Value

called for it's side-effects, NULL return value.

`create_sheets` *create_sheets inserts a new sheet and data to the input workbook. Contains the code for formatting and styles.*

Description

create_sheets inserts a new sheet and data to the input workbook. Contains the code for formatting and styles.

Usage

```
create_sheets(df, wb_active, df_name, opt_header_row)
```

Arguments

`df`	the dataframe to be added
`wb_active`	the workbook into which it will be added
`df_name`	the name of the worksheet
`opt_header_row`	an optional header row, default = 2

Value

NULL as the activeworkbook is being operated on by reference

determine_ada_status *Determine ADA status from ATS applicant record*

Description

helper function for the exported ‘supplement_ats‘ function

Usage

```
determine_ada_status(f1, f2, f3, f4)
```

Arguments

f1	ada field one name
f2	ada field two name
f3	ada field three name
f4	ada field four name

Value

a character ’status’ string of ’Y’, ’N’, or ’Non-Response’.

determine_vet_status *Determine Vet status from ATS applicant record*

Description

helper function for the exported ‘supplement_ats‘ function

Usage

```
determine_vet_status(f1, f2)
```

Arguments

f1	?
f2	?

Value

a character ’status’ string of ’Y’, ’N’, or ’Non-Response’.

`dmhs_base_get_col_types` *Not to be exported*

Description

Not to be exported

Usage

```
dmhs_base_get_col_types()
```

`dmhs_jobs_get_col_types` *Not to be exported*

Description

Not to be exported

Usage

```
dmhs_jobs_get_col_types()
```

`drop_col` *Drop a column from a df by enquoted name if it exists.*

Description

If the column is not contained in the supplied dataframe, the dataframe is returned unmodified. Useful for removing potentially duplicated columns.

Usage

```
drop_col(df, col_name)
```

Arguments

`df`	the dataframe containing the column to drop
`col_name`	the unquoted name of the column to drop

Value

the original dataframe minus the specified column

Author(s)

Ian C Johnson

drop_unneeded_snapshot_cols

Drop inconsistently applied columns from the historical opa snapshots

Description

these are columns not systematically produced in every snapshot. Due to the inconsistent application/definition, they are dropped from the final dataframe. Failure to remove causes row bind issues.

Usage

```
drop_unneeded_snapshot_cols(df)
```

Arguments

df	the historical opa_snapshot dataframe that may or may not contain the column.

Value

the dataframe minus the inconsistent columns

Author(s)

Ian C Johnson

filter_by_max_per_key

Filter a dataframe to only include rows holding a maximum value for each key-value.

Description

a utility function that filters to only the records for each key-value which contain the . This assumes that the key values are duplicated over rows with given numeric or date column differentiating their values.

Usage

```
filter_by_max_per_key(df, key_col_name, col_to_max_name)
```

Arguments

df	the dataframe to be filtered
key_col_name	the name of the column containing the keys to use as grouping variables. Uses quasi-quotation so supply the column name unquoted. See https://dplyr.tidyverse.org/articles/program
col_to_max_name	the name fo the column containing a numeric or date value. only the row containing the max value in this column will be kept. Uses quasi-quotation so supply the column name unquoted. See https://dplyr.tidyverse.org/articles/programming.html

Value

the original dataframe filtered to only contain the max-dated records. All records with this max date per grouping variable will be returned.

Author(s)

Ian C Johnson

---

format_allEE_dates *Format All EE Report Dates*

Description

Properly format dates using the ISO YYYY-MM-DD standard. All Employees report formats them as character type in the DD-MMM-YYYY format.

Usage

```
format_allEE_dates(df)
```

Arguments

df	dataframe containing the all employees data

Value

the input dataframe with revised date formats

---

format_org_df *Format the org hierarchy dataframe returned by build-org_hierarchy to be standardized with column one being the highest level of the org hierarchy, col 2 being the next lower, etc. Effectively reverses the org hierarchy as previously stored.*

Description

Format the org hierarchy dataframe returned by build-org_hierarchy to be standardized with column one being the highest level of the org hierarchy, col 2 being the next lower, etc. Effectively reverses the org hierarchy as previously stored.

Usage

```
format_org_df(df, include_names = T, ftvorgn_code_title)
```

Arguments

df	the dataframe supplied by build_org_hierarchy function
include_names	boolean to include names
ftvorgn_code_title	an orgcode-title lookup table as supplied by the get_code_title_lu function

Value

a dataframe with a seed org value followed by it's hierachy in the least to most granular order. One column per org level with 1 being highest and 8 being the lowest/most-specific

---

`format_sht_names` *format_sht_names ensures that the name to be used for an execl table or worksheet is properly formatted*

Description

Requries the length to be less than or equal to 31 (sheet max) and replaces all spaces with underscores (table requirement)

Usage

```
format_sht_names(name_vec)
```

Arguments

`name_vec`	a string to be used as a worksheet name and/or table name

Value

the properly formatted string. if no formatting is needed then return the input parameter string.

`fread_allee_csv` *Read All Employee Report from CSV source*

Description

given a csv file containing the all employees report data, load it into a dataframe. Uses data.table's `fread` function for performance reasons. Column types are specified using `all_ee_col_types` function.

Usage

```
fread_allee_csv(path, name)
```

Arguments

`path`	the full path the csv file
`name`	the full name of the csv file

Value

an unnamed dataframe containing the all employees data

See Also

allee_dates_from_fnames, all_ee_col_types, get_all_ee_report

ftvorgn_date_filter *Filter FTVORGN table by effective date.*

Description

Use to exclude future and historical data as specified by the as_of_date param.

Usage

```
ftvorgn_date_filter(as_of_date_in, ftvorgn_data)
```

Arguments

ftvorgn_data	the table of FTVORGN data as pulled by SELECT * FROM FTVORGN
as_of_date	the date used to determine historical, current, and future records

Value

a dataframe containing a single row for each unique org code. Additional rows would indicate that historical or future data was erroneously included.

get_access_conn *Get an Access database connection object*

Description

Get a connection to local or network stored Access db. The connection allows for basic operations to be made on the access database.

Usage

```
get_access_conn(db_file_path = "X:/Employees/EmployeesFY21.accdb")
```

Arguments

db_file_path	A full file path to the access database including the file name. Default value should be updated to reflect the most recent Employees annual snapshot file

Value

a connection object that can be used to exectute operations on the db

Author(s)

Ian C Johnson

get_account_lu *Pull Account Codes and associated Account Name*

Description

Make a lookup dataframe to associate account codes and account names as of a particular date. Pulls from the FTVACCT Banner table

Usage

```
get_account_lu(as_of_date, opt_bann_conn)
```

Arguments

as_of_date	the date which will be used to filter the table. Any records with effective dates after the as_of_date will be removed
opt_bann_conn	a banner connection object used to

Value

A table containing account codes and account names as-of the specified input parameter date. There will be a single row per unique account codes

get_address_data *Get the most recent Mailing and Campus Address Data for a set of PIDMs.*

Description

An unoptimized banner pull for campus and address data. Primarily pulled from SPRADDR table.

Usage

```
get_address_data(pidm_vec, opt_bann_conn)
```

Arguments

pidm_vec	a vector containing a set of pidms for which the addresses will be returned
opt_bann_conn	if a banner connection has already been made, supply it here. Otherwise, this function will prompt for logon credentials for a one time use connection.

Details

Depreciated. Use 'pull_contact_info' function instead.

Value

a dataframe containing one row per person per address type with corresponding address columns

`get_all_ee_report` *Get a dataframe containing one or more all employees reports*

Description

Load one or more All Employee reports from csv source. Optionally, save the compiled RDS file for faster loading in the future.

Usage

```
get_all_ee_report(
  folderpath,
  most_recent_only = TRUE,
  opt_start_date,
  opt_end_date,
  supplement = TRUE,
  save_output = FALSE
)
```

Arguments

`folderpath`	the folderpath containing the csv files.
`most_recent_only`	Boolean. Simplest way to specify loading the most recent csv file only for performance reasons.
`opt_start_date`	An optional start date if only certain files should be loaded. If not specified and `most_recent_only == FALSE`, then will load all files found in folderpath.
`opt_end_date`	An optional end date if only certain files should be loaded. If not specified and `most_recent_only == FALSE`, then will load all files found in folderpath.
`supplement`	Boolean value indicating if the dataframes should have additional derived columns added. Examples of columns include EMR Job Type, Fiscal Year, and Longevity Bonus.
`save_output`	A logical parameter to determine if a RDS file should be saved to a user selected directory. Useful for large-data pulls that will have recurrent needs.

Value

a single dataframe containing one or more all employees reports. report data can be distinguished by the added 'date' column.

Author(s)

Ian C Johnson

get_banner_conn *Get a Banner database connection object*

Description

Connects to the MSU Production Oracle Banner Database. Requires valid Banner username and password credentials. Requires appropriately configured tnsnames.ora file. Depends on ROracle to create Oracle database driver (aka OraDriver).

Usage

```
get_banner_conn(opt_pword, verbose = T)
```

Arguments

opt_pword	WARNING !!! password stored as plaintext until completion of connection attempt.

Value

a connection object that can be used to exectute operations on the db

Author(s)

Ian C Johnson

get_banner_snapshot *Pull the Employee snapshot for a particular as-of date*

Description

The employee snapshot is a general dataset containing employee and job information.

Usage

```
get_banner_snapshot(
  date,
  include_leave = FALSE,
  banner_org_vec = "All",
  max_fund_only = TRUE,
  opt_bann_conn,
  remove_eclses,
  return_input_params = TRUE,
  opt_campus_filter = "BZ",
  opt_rank_df,
  opt_tenure_df
)
```

Arguments

date	an 'as-of' date primarily used to filter job-specific data.
include_leave	an optional boolean value indicating whether or not to include LWOP and LWOP/WB i.e. with Benefits Jobs in the snapshot.
banner_org_vec	an optional parameter allowing for filtering to only return jobs that have majority, or a portion, of job funding from the given vector of Org #s. Defaults to all Org #s.
max_fund_only	boolean value determining if all labor distribution splits will be returned, or only those with the majority funding in each job. Defaults to TRUE.
opt_bann_conn	an active banner connection object typically derived from 'get_banner_conn'. If not provided will create a temp connection
remove_eclses	a character vector containing eclasses which will be not be returned by sql query.
return_input_params	a boolean value indicating whether an addtional column containing the input parameters should be included in the ouput dataframe.
opt_campus_filter	a character vector containing one of more Campus codes to filter sql statement. Leave as NA to not filter by campus.
opt_rank_df	a dataframe containing all Rank records supplied by 'opa::get_rank_records()'. If not supplied, will pull data within this function. Useful for looping scripts that run this function multiple times (particularly for multiple dates, etc).
opt_tenure_df	a dataframe containing all Tenure records supplied by 'opa::get_tenure_status()'. If not supplied, will pull data within this function. Useful for looping scripts that run this function multiple times (particularly for multiple dates, etc).

Value

a list of snapshot query returns. The names of the list items specify the date on which query is set.

---

get_bls_salary *Load BLS salary data*

Description

loads BLS data for MSA, State, and National aggregations

Usage

```
get_bls_salary(year, add_year_key = F)
```

Arguments

year	the dataset's benchmark year typically published in the following year
add_year_key	a field combining the SOC code and the year of the dataset. Useful for joining to datasets when working with multiple years of data

Value

a wide dataframe containing median and mean aggregates on the msa, state, and national levels.

get_code_title_lu *Get a lookup table defining the relationship between org number and org title*

Description

Get a lookup table defining the relationship between org number and org title

Usage

```
get_code_title_lu(ftvorgn_data)
```

Arguments

ftvorgn_data	the ftvorgn data specific to a single as-of date.

Value

a dataframe containing FTVORGN_TITLE and FTVORGN_ORGN_CODE

get_cupa_salary *Load CUPA Salary data*

Description

Load CUPA salary benchmark data from Helene.

Usage

```
get_cupa_salary(ay_year = 2018)
```

Arguments

ay_year	the academic year for which salary data is pulled. 2016 refers to 2016-2017, etc.

Value

a single dataframe containing admin, professional, and staff cupa salaries for Land Grant institutions

get_db_table_col_names *Get the names of all columns in a given database table.*

Description

Given a database table which exists in said database, get the column names. This is particularly useful when exploring less familiar tables and databases.

Usage

```
get_db_table_col_names(tbl_name, db_conn)
```

Arguments

tbl_name	a character string containing the database table name. Case specific.
db_conn	an Access or Oracle database connection object.

Value

a character vector containing the names of the fields

Author(s)

Ian C Johnson

See Also

get_db_table_names

get_db_table_names *Get all table names from a given database*

Description

Currently built to return Access (DBI connector) and Oracle (ROracle) database table names.

Usage

```
get_db_table_names(db_conn)
```

Arguments

db_conn	the database connection object

Value

a text character vector of table names

Author(s)

Ian C Johnson

get_dmhs *Load Datamart history files*

Description

Load Datamart history files

Usage

```
get_dmhs(
  most_recent_only = TRUE,
  opt_year = NA,
  opt_month = NA,
  fpath = "X:/Employees/EMR Report (production)/DM csv files/"
)
```

Arguments

most_recent_only	boolean to determine whether to load all files, or only the most recent
opt_year	an optional 4-digit numeric year paramater
opt_month	an optional 1 or 2 digit numeric month parameter
fpath	an optional fpath to be used if the source file location changes or the hellene drive is not mapped to the X Drive

Value

a single dataframe containing the BASE table joined with the JOBS table for the requested months

get_ftvorgn_data *Get FTVORGN table data*

Description

ftvorgn data is used to calculate organization hierarchy via the logical structure of the organization codes. This validation table is maintained by the finance team and regularly updated with new organization codes and org. titles. For this reason, it is recommended to use this function to pull the most up-to-date ftvorgn table data directly from Banner. Requires Banner logon credentials.

Usage

```
get_ftvorgn_data(opt_as_of_date, opt_bann_conn)
```

Arguments

opt_as_of_date	a POSIXct formated date to be used to select only records on or before a certain time. useful when pulling backdated banner snapshots
opt_bann_conn	an active banner connection object typically derived from `get_banner_conn()`. If not provided will create a temp connection

Value

a dataframe containing all FTVORGN table variables. Depreciated rows are removed leaving only the most recent record on or prior to the as_of_date.

`get_gender` *Pull Gender data*

Description

given a vector of pidms, pull the associated vector of genders from the SPBPERS Banner table.

Usage

```
get_gender(pidm_vec, opt_bann_conn)
```

Arguments

`pidm_vec`	a vector of pidms not to exceed 1000 entries
`opt_bann_conn`	an optional banner connection. one will be created if not supplied

Value

a dataframe containing the pidms and aassociated genders

`get_netid_pidm_gid` *Pull GID, PIDM, or NetID identifiers for a vector of employee ids*

Description

Three primary identifier values used in various datasets. GID is the banner ID typically used for outward facing implementations. PIDM is the internal ID used on banner database tables. NetID is the the external id used for single-sign-on and other third party applications.

Usage

```
get_netid_pidm_gid(ids_in, type_in, opt_bann_conn)
```

Arguments

`ids_in`	a vector of ids for which an alternative type will be pulled. Can handle leng
`type_in`	one of 'pidm', 'gid', or 'netid' which specifies the input id type
`type_out`	one of 'pidm', 'gid', or 'netid' which specifies the output id type

get_opa_access_snapshots *Pull historical cleaned OPA employee snapshots*

Description

Historically, the snapsht has been taken once a year in mid-October. These datasets are reviewed for accuracy and have driven much of the head count, job count, and FTE reporting. The primary snapshot table uses a name int he form of 'Employees19F' where 19 indicates the calendar year of the snapshot and F indicates the Fall Semester. Due to the working file nature of these documents, care should be taken to ensure that only the proper tables are loaded

Usage

```
get_opa_access_snapshots(
  opt_snapshot_fpath = "X:/Employees/EmployeesFY21.accdb",
  opt_bann_conn
)
```

Arguments

opt_snapshot_fpath	the filepath to an access database containing the most recent employee snapshot table.
opt_bann_conn	an optional banner connection object.

Value

a dataframe containing all appended reviewed-snapshots.

Author(s)

Ian C Johnson

get_osu_salary *Pull Year of OSU Faculty Salaries*

Description

OSU Faculty salaries provide annual faculty salaries based on 9/10 month contracts. If comparing to FY contract faculty such as Department Heads, be sure to convert to AY by multiplying Annual Salary by 9/11. Data aggregated by CIP code. Assumes that the Faculty type has been included in the Avg Sal and N columns in the format FacType - Average Salary aor FacType - N

Usage

```
get_osu_salary(year = 2019, pivot_long = T)
```

Arguments

year	The year corresponding to the salary benchmark
pivot_long	original datasource stores faculty Rank in column names. Use pivot longer to convert to a tidy dataset with Rank stored in an independent column

Value

a dataframe containing faculty AY salary benchmarks and number of surveyed jobs contributing to the benchmark.

---

get_payroll_data *Load Flat file 'Payroll Earnings and Labor Distribution by Employee' from file*

Description

load payroll datafiles into a single dataframe. Start and end date should cover first day of the month in question. Data originally sourced from the ReportWeb 'Payroll and Earnings Labor Distribution Report.'

Usage

```
get_payroll_data(
  opt_fpath = "X:/Employees/Payroll Earnings & Labor Distrubtion by Employee/",
  most_recent_only = T,
  opt_start_date,
  opt_end_date,
  filter_max_org = T,
  add_pidm = T,
  opt_bann_conn
)
```

Arguments

opt_fpath	an optional parameter if the payroll files are located in a non-default location
most_recent_only	pull only the most recent payroll saved to the fpath.Overrules opt_start_date.
opt_start_date	an optional POSIXct start date. Payrolls covering dates prior will be excluded from the returned dataframe.
opt_end_date,	an optional POSIXct end date. Payrolls covering dates after will be exclude form the returned dataframe.
filter_max_org	a boolean operator spcifying whether to return only the row corresponding to the highest funding organizations for each job, or to return the full labor distribution
add_pidm	a boolean specifying whether a pidm should be included in the output. the file currently only contains GID values
opt_bann_conn	a banner connection option required if pidms are being returned. can be dropped if pidms not included

Value

a single dataframe containing all payroll data

get_person_names_data *Pull most Employee Name data*

Description

return the full and most up-to-datre names of individuals specified by the input gid or pidm vector. No limit on length of input vectors

Usage

```
get_person_names_data(pidm_vec, gid_vec, opt_bann_conn)
```

Arguments

pidm_vec	a vector of pidms specifiying the individuals whose names will be returned
gid_vec	a vector of gids specifiying the individuals whose names will be returned
opt_bann_conn	an optional banner connection object

Value

a dataframe containing names and pidms

---

get_pidm_gid_lu *Pull a GID-PIDM lookup table from Banner #' @description get a dataset containing gids and their corresponding pidms. This is pulled directly from banner. If a banner connection is not possible, see OPA's employee snapshot files. This dataset is comprehensive for every student and employee that has ever worked on campus while banner has been implemented*

Description

Pull a GID-PIDM lookup table from Banner #' @description get a dataset containing gids and their corresponding pidms. This is pulled directly from banner. If a banner connection is not possible, see OPA's employee snapshot files. This dataset is comprehensive for every student and employee that has ever worked on campus while banner has been implemented

Usage

```
get_pidm_gid_lu(opt_banner_conn, opt_pidm_vec, opt_gid_vec)
```

Arguments

opt_banner_conn	if a banner connection has already been made, supply it here. Otherwise, this function will prompt for logon credentials for a one time use connection.
opt_pidm_vec	use this optional parameter to filter the underlying sql query. Useful for time-sensitive applications.
opt_gid_vec	use this optional parameter to filter the underlying sql query. Useful for time-sensitive applications.

Value

a two column dataframe containing gids and corresponding pidms

Author(s)

Ian C Johnson

`get_race_data` *Pull Race/Ethnicity Data*

Description

Pull ipeds race data from bob's ipeds access db. The path is optional incase the access database location changes.

Usage

```
get_race_data(
  race_file_path = "X:/IRCommon/RACE/IPEDS_Race_2.accdb",
  re_tbl_name = "RE20200922",
  race_tbl_name = "Race20200922",
  opt_pidm_filter
)
```

Arguments

`race_file_path`	the access filepath and name containing bob's IPEDS race/ethnicity data
`re_tbl_name`	the access tbl_name from which to pull data
`race_tbl_name`	the access table name from which race descriptions will be pulled
`opt_pidm_filter`	an optional vector of pidms to filter the returned dataset

Value

the full ipeds race table as a dataframe keyed by PIDM

`get_rank_records` *Return Rank and Aggergated Rank records per person as-of a particular date.*

Description

These values are used to differentiate NTT and TT/T Faculty. They are pulled from the PERRANK and PTRRANK Banner tables.

Usage

```
get_rank_records(
  return_most_recent = TRUE,
  opt_as_of_date,
  opt_bann_conn,
  opt_rank_records
)
```

Arguments

return_most_recent	a boolean filter specifying if only the most recent records relative to the as_of_date should be returned. If no as_of_date parameter supplied, will default to the most recent to the Sys.Date()
opt_as_of_date	use to filter to only those rank records applicable to a certain date. will convert ISO formated character strings to POSIXct. If missing, will default to the current date.
opt_bann_conn	an optional banner connection to pull data from PERAPPT banner table
opt_rank_records	an optional dataframe consisting of joined ptrrank and perrank data. WARNING: No error checking is done on this input!

Value

a data frome consisting of pidm, rank, rank desc, aggregated rank, agg rank desc, begin date, and begin date month floor. Only most recent rank record (less than as_of_date) returned per per person

Author(s)

Ian C Johnson

See Also

get_tenure_status

---

get_stipend_comments *Get Stipend Comments for a given pidm from the PPRCCMT table*

Description

Get Stipend Comments for a given pidm from the PPRCCMT table

Usage

```
get_stipend_comments(pidms, opt_bann_conn)
```

Arguments

pidms	the vector of unquoted pidms, not to exceed length of 1000
opt_bann_conn	the optional banenr connection supplied by opa::get_banner_conn()

Value

a dataframe containing all Stipend comments for the given pidms with both PIDM and the fiscal year in which they were entered.

`get_tenure_status` *Get a dataframe containing tenure status records.*

Description

The tenure status dataframe can be filtered to * all records * only the most recent record for each individual employee * only the most recent record as of a specific date * all records prior to a specific as-of date

Usage

```
get_tenure_status(
  return_most_recent = TRUE,
  opt_as_of_date,
  opt_bann_conn,
  opt_tenure_records,
  opt_rename_columns = TRUE
)
```

Arguments

`return_most_recent`
: a boolean which specifies whether all tenure status records should be returned as of a specific date, or if only the most recent record prior or equal to as-of-date. Defaults to TRUE

`opt_as_of_date`
: an optional as-of date to filter the tenure status records. if supplied, will discard any records occuring after this date. Uses PERAPPT_APPT_EFF_DATE to determine cutoff.

`opt_bann_conn`
: an optional banner connection object to pull data from PERAPPT

`opt_tenure_records`
: tenure record table data which may be included as an input here to avoid re-pulling the data from banner. Useful when using in a loop.

`opt_rename_columns`
: choose whether to rename output columns to shorthand or use the default banner table column names.

Value

tenure status data from PERAPPT table specific to a given as-of date.

Author(s)

Ian C Johnson

`import_raw_osu_data` *Convert the raw OSU DATAFEED file into a semi-formatted dataframe with appropriate column names. Optionally, write this 'long-format' df to the FacSal2 access db.*

Description

Convert the raw OSU DATAFEED file into a semi-formatted dataframe with appropriate column names. Optionally, write this 'long-format' df to the FacSal2 access db.

Usage

```
import_raw_osu_data(fpath, write_to_access)
```

Arguments

`fpath`	the filepath including filename and extension pointing of the fixed-width DATAFEED file
`write_to_access`	a boolean specifying whether the dataframe should be written to the FacSal2 db.

Value

a dataframe containing the DATAFEED file with appropriate column names

`job_lbr_dist_qry` *job_lbr_dist_qry*

Description

internal function in the msuopa package. queries Banner given a set of pidm, posn, suffix job keys and returns labor distribution data from the NBRJLBD table.

Usage

```
job_lbr_dist_qry(job_key_vec, bann_conn, opt_as_of_date)
```

Arguments

`bann_conn`	the banner connection object derived from msuopa::get_banner_conn()
`pidm_vec`	a vector containing pidms to be used in the job key
`posn_veca`	a vector containing position numbers to be used in the job key
`suff_vec`	a vector containing suffixes to be used in the job key
`as_of_date`	a POXIXct date specifying the effective date used for the labor distribution
`most_recent_only`	a boolean determining if only the most recently effective dated records should be returned
`majority_percent_only`	a boolean value determining if only the majority funding rows should be returned for each job's effective date

Value

a dataframe containing pidm, posn, suff, job_key, account, program, index, fund, organization number

---

`load_sql_qry` *Load and format a SQL query script from text/file source.*

Description

load a properly formatted sql query from a plain text file to be sent to an oracle db via ROracle::dbSendQuery or an access db via DBI::dbSendQuery

Usage

```
load_sql_qry(file_path, file_name)
```

Arguments

`file_path`	the full folder path containing the file
`file_name`	the full name of the file with extension

Value

a string containing the sql query

Author(s)

Ian C Johnson

See Also

'DBI::dbSendQuery()' and 'ROracle::dbSendQuery()'

`name_query` *Pull most recent name values from SPRIDEN table*

Description

helper function. Fails with more than 1000k pidm values. Use 'get_person_names_data' function instead.

Usage

```
name_query(pidm_vec, bann_conn)
```

Arguments

`pidm_vec`	a vector containing pidm values of the employees for which names will be pulled
`bann_conn`	a banner connection object supplied by 'opa::get_banner_conn()'

Value

a dataframe containing pidm and name.

`pad_gid` *Pad Gid Values for Dropped preceeding Zeros*

Description

GID values take the character form '-xxxxxxxx' where 'x' is a numeric character. By default, most applications erroneously assing these values to numeric data types causing the preceding zeros to be droped. There may be issues if only some of the values are proper lengths

Usage

```
pad_gid(gid_vec)
```

Arguments

`gid_vec`	the vector containing gid vlue

Value

a character value where each GID volue is the proper 9-character length with preceeding zeros included.

`pay_factor_month_conversion`
Determine approximate month length of a contract given number of payrolls over which it is paid.

Description

this is only approximate because bi-weekly payrolls do not align with the calendar year. Pay factors are a static value found in the NBRJOBS_FACTORS Banner field.

Usage

```
pay_factor_month_conversion(pay_factors, campus_code)
```

Arguments

`pay_factors`	the numeric payfactors from the NBRJOBS_FACTORS field. This is explicitley the default number of payrolls over which the total job compensation is paid.
`campus_code`	

Details

Depreciated as of bi-weekly payroll conversion. Use 'pull_paynos' to link payroll number/factors and dates/time-spans

Value

the month duration of a contract depending on the number of payrolls in which it was active in a given fy, and the campus on which the job resides

---

`process_nbrjlbd_data` *Pull or process Job Labor Distriubtion as of a specific date, or most recent data available.*

Description

Pull or process job labor distribution for each unique job defined by pidm, posn, suff, get the fund, orgn, account, program and index information. If the job has split labor distribution, each split will use a different row with associated percent (0-100) values. Use input parameters to control which labor distribution are returned (filtered by job and/or date). Requires unaltered NBRJLBD column names

Usage

```
process_nbrjlbd_data(
  opt_job_lbr_dist_df,
  opt_as_of_date = NA,
  opt_job_keys,
  most_recent_only = T,
  majority_percent_only = F,
  opt_bann_conn
)
```

Arguments

`opt_job_lbr_dist_df`
: dataframe containing NBRJLBD recrods with unaltered column names. See `pull_job_lbr_dist`.

`opt_as_of_date`
: a posixCT date used to specify the specific date on which the labor distribution shoudl be used. Defaults to Sys.Date()

`most_recent_only`
: a boolean determining if the function will return only the most recent labor distribution data, or all labor distribution data

`majority_percent_only`
: a boolean value

`opt_bann_conn`
: an optional banner database connection object

Value

a dataframe containing each job's fund, organization code, account code, index, program and percent

Author(s)

Ian C Johnson

See Also

get_job_lbr_dist job_lbr_dist_qry

`pull_contact_info` *Pull Up-to-date and Preferred contact information*

Description

Pulls data from the V_S_CURRENT_CONTACT_INFO View in Banner. This view replaces Bob's old Auto Address Access script.

Usage

```
pull_contact_info(pidm_vec, opt_bann_conn)
```

Arguments

`pidm_vec`	a vector of pidms to filter the returned dataset. Can handle vectors greater than 1k items long.
`opt_bann_conn`	an optional banner connection object

Value

a dataframe containing preferred/up-to-date email, phone, and address information.

`pull_degree_history` *Pull Degree History Data for all Employees*

Description

Pull Degree History Data for all Employees

Usage

```
pull_degree_history(opt_bann_conn)
```

Arguments

`opt_bann_conn`	an optional banner connection object

pull_deleted_modified_cips *pull_deleted_modified_cips*

Description

get a datafrmae containing moved or deleted cip codes from the 2010 to 2020 update

Usage

```
pull_deleted_modified_cips()
```

Value

a dataframe specifying cip codes that have been deleted, had cip codes merged into them, or been merged into a new cip

pull_job_labor_dist *Pull Job Labor Distribution Records for specific job-keys and an as-of date.*

Description

pull NBRJLBD table records from banner for specified pidm-posn-suff job-keys, as-of a specific date. Any modifications to labor distribution which occur after this date are not included. Likewise a boolean input parameter controls whether all previous labor distribution records are included, or merely the most recent that occurred prior to the as-of date.

Usage

```
pull_job_labor_dist(
  job_keys_in,
  as_of_date,
  most_recent_only = T,
  max_fund_percent_only = T,
  opt_bann_conn
)
```

Arguments

job_keys_in	pidm-posn-suff job-key vector
as_of_date	the string or POSIXct date used to filter future-dated records
most_recent_only	a boolean indicating whether to include all historical records previous to as-of date or merely the most recent prior to the as-of date
opt_bann_conn	an optional banner connnection object

Value

a data frame containing the critical columsn to aggregate fund sources. These include, job-key, index, orgn, account and distribution percent.

`pull_paynos` *Return Payroll numbers and dates covering a particular time interval and optionally, campus.*

Description

Pulls from PTRCALN

Usage

```
pull_paynos(start_date, end_date, opt_campus_code, opt_bann_conn)
```

Arguments

`start_date`	the start date of the time range to pull payroll numbers. Will include the payroll containing the date specified.
`end_date`	the end date of the time range to pull payroll numbers. Will include the payroll containing the date specified.
`opt_campus_code`	an optional campus code. One of 'BZ', 'BL', 'HV', 'GF'
`opt_bann_conn`	an optional banner connection object supplied by 'opa::get_banner_conn()'

Value

a dataframe containing payroll years, numbers, start and end dates, and a date interval object for each payroll

`pull_perehis_data` *Pull PEREHIS employee data for a set of pidms*

Description

Pull PEREHIS employee data for a set of pidms

Usage

```
pull_perehis_data(pidm_vec_in, opt_as_of_date, opt_bann_conn)
```

Arguments

`pidm_vec_in`	a vector containing pidms for which perehis data will be returned
`opt_as_of_date`	only return records occuring prior to this optional date. all records returned if not specified.
`opt_bann_conn`	an optional banner connection object

Value

PEREHIS banner table records for a given set of employees

`pull_pr_data` *Pull historic payroll data by job key, campus, year, and payno.*

Description

Pull all PHREARN records given a particular ‘pr_key‘ comprised of the concatenation of ‘Campus‘, ‘Year‘, and ‘Payno‘ OR given a particular set of ‘job_keys‘ comprised of ‘pidm‘, ‘posn‘, ‘suffix‘. Must contain at least one of job_key or pr_key or both.

Usage

```
pull_pr_data(
  job_keys,
  pr_keys,
  remove_leave_no_pay_rows = T,
  opt_bann_conn,
  opt_start_year,
  opt_start_pr,
  opt_end_year,
  opt_end_pr
)
```

Arguments

`job_keys`	the job keys whose payroll records will be returned
`pr_keys`	the payroll keys (campus, year, payno) whose records will be returned
`remove_leave_no_pay_rows`	a boolean flag that removes LNO and LNP earn code rows from teh dataset. These rows cause inflated payroll amount sums due to being populated with the pay amount if active but not actually being paid. Default TRUE.
`opt_bann_conn`	an optional banner connection object
`opt_start_year`	an optional fiscal year to filter to only view records in and after the give year
`opt_start_pr`	an optional payroll number to filter to only view records in the or in a greater than payroll number. Not adapted to handle change from monthly to bi-weekly payroll
`opt_end_year`	an optional fiscal year to filter to only view records in and before the give year
`opt_end_pr`	an optional payroll number to filter to only view records in the or in a less than than payroll number. Not adapted to handle change from monthly to bi-weekly payroll

Value

the phrearn rows specified by job key and/or payroll key

pull_rank_tables *Pull PERRANK and PTRRANK tables from Banner.*

Description

these tables contain the necessary information to determine an individual's rank status on any given date.

Usage

```
pull_rank_tables(bann_conn)
```

Arguments

bann_conn	a banner connection object supplied by the ROracle package or 'opa::get_banner_conn()' function.

Value

joined PERRANK, PTRRANK tables with renamed columns

pull_tenure_table *Pull entire PERAPPT table from Banner.*

Description

PERAPPT used to store tenure/appointment specific data. Necessary for determining tenure status for a given employee and date.

Usage

```
pull_tenure_table(bann_conn)
```

Arguments

bann_conn	a Banner connection object typically derived from 'opa::get_banner_conn()' or the ROracle package

Value

an unmodified dataframe containing all unmodified records from PERAPPT

`remove_na_cols` *remove_na_cols*

Description

remove columns from dataframe if they contain *only* NA values

Usage

```
remove_na_cols(df)
```

Arguments

`df`	the dataframe or datatable from which to remove columns

Value

a datatable with NA columns removed

`remove_na_rows` *remove_na_rows*

Description

remove columns from dataframe if they contain *only* NA values

Usage

```
remove_na_rows(df)
```

Arguments

`df`	the dataframe or datatable from which to remove columns

Value

a datatable with NA columns removed

rename_col *Rename a column in a dataframe*

Description

Quickly rename a column based on it's current name rather than location. This is helpful in certain instances when a name of a column cannot be determined in advance.

Usage

```
rename_col(df, old_name, new_name)
```

Arguments

df	the dataframe containing columns to be renamed
old_name	a string containing the name of the column to be renamed
new_name	a string containing the new name.

Value

the same dataset with a renamed column

Author(s)

Ian C Johnson

return_fy_payrolls *Return single FY's payroll data*

Description

Payroll data can be difficult to compile due to misalignment of the calendar year (from which payno is defined) and the fiscal year, and differing payroll calendars between campuses. This function returns all payroll records for a given fiscal year and campus. An additional field, 'pr_percent_in_fy' is included to indicate the percent of days in the payroll that fell into the requested FY in cases where the payroll spans multiple fiscal years.

Usage

```
return_fy_payrolls(
  fy,
  campus_pict = "ALL",
  simplify = TRUE,
  opt_ptrcaln_data,
  opt_bann_conn
)
```

Arguments

fy	the fiscal year for which to return ptrcaln payno and year
campus_pict	the campus pict code to filter by campus. Must be one of "ALL", "4M", "3B", "6B", "7M".
opt_ptrcaln_data	an optional unmodified PTRCALN dataset. supply to avoid pulling data from Banner.
opt_bann_conn	an optional banner connection object. Only needed if opt_ptrcaln_data not supplied

Value

a dataframe containing pict code and fy inputs, and corresponding PTRCALN_YEAR and PTRCALN_PAYNO. A percent of total indicates the percent of the payroll (# payroll working days in payroll in fy / # payroll working days in payroll)

Author(s)

Ian C Johnson

---

reverse_org_rows *reverse_org_rows*

Description

used to reverse the org hierarchy in built by the get_org_hierarchy function.

Usage

```
reverse_org_rows(df)
```

Arguments

df	the dataframe produced by get_org_hierarchy's loop. Requires a max_org_depth column to know how to place the

Details

TODO: make into a loop

sort_cols_by_name *Reorder columns by alphabetical order*

Description

Reorder columns by alphabetical order

Usage

```
sort_cols_by_name(df)
```

Arguments

df	the dataframe whose columns will be reordered

Value

a dataframe containing columns ordered alphabetically descending order

source_folder_files *Source all R Files contained in a given folder*

Description

Source all files in a given folder for the current R session. Any file ending with *.R will be sourced. implemented in Base R

Usage

```
source_folder_files(folder_path = "./R/")
```

Arguments

folder_path	The folder path containing the R files. By default, uses the ./R/ folder contained in the working directory

Value

Success message will be printed to terminal

Author(s)

Ian C Johnson

split_vec_for_sql *Split Vector into list of vectors each containing a maximum number of values*

Description

Take a vector of values and split it into a list of vectors each containing, at most, the number of items specified in max_size. Useful for constructing plsql queries against Banner using the ‘

Usage

```
split_vec_for_sql(vector, max_size = 1000, all_distinct = T)
```

Arguments

vector	the vector of values to be split
max_size	the maximum items to place in each new vector. defaults to the plsql 1000 item limit

Value

a list containing the split vectors

Author(s)

Ian C Johnson

summarize_stipend_fy *Summarize total Stipend payment per individual over a fiscal year.*

Description

Determines stipend recipients and payments from payroll data. Links to home department and non-stipend job titles using Banner snapshots taken on a monthly basis covering entire fiscal year. Summarizes all non-stipend payments and stipend payments as percent of non-stipend payments.

Usage

```
summarize_stipend_fy(
  fy_in,
  write_to_file = F,
  opt_snapshot_df,
  opt_fpathname,
  opt_bann_conn
)
```

Arguments

write_to_file	a boolean indicating if the output should be written to file
opt_snapshot_df	an optional bound set of snapshots from the fiscal year in question
opt_fpathname	if write_to_file is TRUE, set the file path and name here
opt_bann_conn	an optional banner connection supplied from opa::get_banner_conn()
fy	the numeric fiscal year in question

---

supplement_all_ee *Add supplmental data to All Employees Report*

Description

A wrapper for several functions that add additional columns to the all employees report. Adds Job Type, Longevity Bonus, Full Name, Job Key, Job Date Key, and fiscal year.

Usage

```
supplement_all_ee(df)
```

Arguments

df	the all employees report with unaltered column header names.

Value

the original input dataframe with the additional columns

Author(s)

Ian C Johnson

See Also

add_emr_job_type, add_emr_orgs, add_longevity_bonus

supplement_ats *Supplement ATS EEO-6 Dataset*

Description

determine veteran and disability status for each applicant from the ATS EEO-6 report. Does not error check for missing columns. This is necessary because, historically, different fields have been used to store teh same attributes. Furthermore, the field names do not indicate which is 'correct' or even in current use.

Usage

```
supplement_ats(ats_data)
```

Arguments

ats_data	the dataset from the ATS reporting system. Custom report titled 'EEO Applicant Details Report'.

Value

a dataframe containing input ats dataset with modified column names to standardize modifications to ATS (ada_1,ada_2... instead of full sentences)

transpose_tidyr *Transpose a dataframe using tidyr functions*

Description

Transpose a dataframe using pivot_longer and pivot_wider. The names_to_str specifies the column names into which the previous column names will be added. names_from is the current column of values which will be transposed into the new columns. If a prefix to the column name values is needed, it can be supplied via names_out_prefix_str.

Usage

```
transpose_tidyr(df, names_to_str, names_from, names_out_prefix_str)
```

Arguments

df	the dataframe to be transposed
names_to_str	the name of the new column containing the values previously stored in the column names
names_from	the unquoted column whose values will be used as new column names
names_out_prefix_str	an optional prefix to be used in the new column names

Value

a transposed dataframe from the original df

trim_ws_from_df *Time Whitespace from all character columns in dataframe*

Description

remove whitespace surrounding values stored in character type dataframe columns. Commonly needed when pulling data from Access

Usage

```
trim_ws_from_df(df)
```

Arguments

df	the dataframe to clean of whitespace

Value

the original dataframe with whitespace removed from character column values

write_list_report *write_list_report*

Description

Export a list of dataframes to an excel workbook. Each sheet contains a dataframe in the list. The list should be named. If not, will be given defaults df_1, df_2, ... df_n.

Usage

```
write_list_report(df_list, output_name_path)
```

Arguments

df_list	the list of dataframes
output_name_path	the full name of the folder and file name to be exported

write_report *write_report*

Description

Export a dataframe to an excel and/or csv file. Typically used to share aggregated datasets.

Usage

```
write_report(df, fpath, fname, sheetname, include_xlsx = TRUE)
```

Arguments

df	the dataframe to be output
fpath	the full name of the path to which the file will be written
fname	the file name to be used for the output. Do not include .csv or .xlsx or any other filetype specifier
include_xlsx	an optional parameter which defaults to TRUE. Set to FALSE to only output an a flat csv file. Ideal when writing a large number of individual files
sheetName	the name of the sheet and table

Value

A success message will be returned

Index