Fusetools

Release 1.0

FuseCloud

Feb 19, 2022

CONTENTS

1 Roadmap 3

2 Download and install 5

3 Minified Version 7

4 Licensing 9

Python Module Index 79

Index 81

FuseTools is a Python package to help you accomplish miscellaneous and business tasks. With FuseTools you can access a wide variety of applications and APIs for different use cases.

CHAPTER

ONE

ROADMAP

FuseTools has functionality to interact with many applications and services including the following:

CHAPTER

TWO

DOWNLOAD AND INSTALL

This package is in the Python Package Index, so `pip install fusetools` should be enough. You can also clone it on Github.

CHAPTER

THREE

MINIFIED VERSION

Don't need all of FuseTools? You can create your own variant with just the modules you want.

```
from fusetools.devops_tools import Local

Local.create_sub_pkg(
    src_pkg_dir=src_pkg_dir, #source folder of existing package
    src_pkg_name="fusetools", #existing package name
    tgt_pkg_dir=sav_dir, #new package folder
    tgt_pkg_name=tgt_pkg_name, #new package name
    folder_list=False, #folders to copy over
    file_list=False, #files to copy over
    # specify the modules you want to use
    module_list=["gsuite_tools.py",
                 "financial_tools.py"],
    install_pkg=False, #whether or not to install new package
    python_alias="python" #your OS's python alias
)

# create init.py file
myBat = open(sav_dir + f"{tgt_pkg_name}/__init__.py", 'w+')
myBat.write(str(""))
myBat.close()
```

CHAPTER

FOUR

LICENSING

Fusetools is distributed under the MIT License.

4.1 Introduction

Fusetools is a Python package with code to help you accomplish miscellaneous and business tasks. With Fusetools you can access a wide variety of applications and APIs for different use cases.

4.1.1 Roadmap

Fusetools has functionality to interact with many applications and services including the following:

4.1.2 Download and install

This package is in the Python Package Index, so `pip install fusetools` should be enough. You can also clone it on Github.

4.1.3 Licensing

Fusetools is distributed under the MIT License.

4.2 Examples

Here we have some example Jupyter notebooks for the following modules:

- cloud tools - AWS
- cloud tools - Firebase
- comm tools
- db etl tools
- gsuite tools

4.3 Changes

4.4 Links

fusetools

4.5 fusetools

fusetools.analytics_tools	DataFrame & SQL Tools for Analytics.
fusetools.comm_tools	Communication services.
fusetools.commerce_tools	Marketplace services.
fusetools.dashboard_tools	Dashboards and data visualization applications.
fusetools.date_tools	Functions for interacting with Python date objects.
fusetools.db_conn_tools	Database connections and engines.
fusetools.db_etl_tools	Database connections and engines.
fusetools.gsuite_tools	Google Suite Tools.
fusetools.home_tools	Home Automation Tools.
fusetools.logging_tools	Logging tasks.
fusetools.marketing_tools	Marketing, Advertising, Campaign & Analytics Tools.
fusetools.ml_tools	Functions for interacting with Machine Learning Tools.
fusetools.music_tools	Functions for interacting with Music Tools.
fusetools.pm_tools	Functions for interacting with Project Management Tools.
fusetools.research_tools	Functions for conduction various Research tasks.
fusetools.stat_tools	Functions for interacting with Machine Learning Tools.
fusetools.text_tools	Tools for performing text tasks.
fusetools.web_tools	

4.5.1 fusetools.analytics_tools

DataFrame & SQL Tools for Analytics.

Classes

Pandas	Functions for running analytical Pandas operations
SQL	Functions for running analytical SQL operations

fusetools.analytics_tools.Pandas

class fusetools.analytics_tools.**Pandas**

Bases: object

Functions for running analytical Pandas operations

Methods

append_window_agg	Joins a window function's aggregation to a Pandas DataFrame.
find_na_holder	Returns the combination of 2 de-duped columns in a Pandas DataFrame.
period_comp	Creates a snapshot comparison between two periods.
period_start_dt	Returns the first day of a year or month for a Pandas Series.
ptd_measure	Creates a 'Period to date' aggregation.
yoy_comp	Computes a YoY cumulative YTD comparison across for a given week.

classmethod append_window_agg(*df*, *dim*, *metric*, *metric_agg*, *comp_col=False*)

Joins a window function's aggregation to a Pandas DataFrame.

Parameters

- **df** – Pandas DataFrame.

- **dim** – Column for which to partition of data by.
- **metric** – Column to aggregate.
- **metric_agg** – Type of calculation to perform.
- **comp_col** – Flag of whether or not to create a comparison column.

Returns Pandas DataFrame with a window function's aggregation.

classmethod find_na_holder(*df*, *col*, *col_new*)
Returns the combination of 2 de-duped columns in a Pandas DataFrame.

Parameters

- **df** – Pandas DataFrame
- **col** – Original column.
- **col_new** – New column.

Returns Combination of 2 de-duped columns in a Pandas DataFrame.

classmethod period_comp(*df*, *period_field*, *val_fields*, *dim=False*, *val_field_suffix=False*, *hist=False*)
Creates a snapshot comparison between two periods.

Parameters

- **df** – Pandas DataFrame.
- **period_field** – Column with period to compare across
- **val_fields** – List of columns with numeric values to compare
- **dim** – Column with dimension to group across (optional)
- **val_field_suffix** – Suffix for value field to add to final dataset (optional)
- **hist** – Include history flag, returns all periods if True, otherwise just the most recent two periods

Returns Comparison Pandas DataFrame

classmethod period_start_dt(*df*)
Returns the first day of a year or month for a Pandas Series.

Parameters **df** – Pandas DataFrame.

Returns First day of year or month for Pandas Series.

classmethod ptd_measure(*df*, *period*, *val_fields*, *kpi*, *dim=False*)
Creates a 'Period to date' aggregation.

Parameters

- **df** – Pandas DataFrame
- **period** – Type of period (year, month)
- **val_fields** – Columns to aggregate.
- **kpi** – Type of aggregation to perform.
- **dim** – Dimension to group comparison by (Optional).

Returns Pandas DataFrame with PTD measure.

classmethod yoy_comp(*df*, *val_dict*, *dim=False*, *hist=False*)
Computes a YoY cumulative YTD comparison across for a given week.

Parameters

- **df** – Pandas DataFrame.
- **val_dict** – Column and aggregation type specification.
- **dim** – Dimension to group comparison by (Option).
- **hist** – Flag of whether to keep all historical date combinations.

Returns Pandas DataFrame with YoY cumulative YTD comparison.

fusetools.analytics_tools.SQL

class fusetools.analytics_tools.**SQL**
Bases: object

Functions for running analytical SQL operations

Methods

sql_rs_wow_comp	Performs a week over week comparison.
sql_rs_yoy_cum_comp	Performs a cumulative aggregation over the year as well as provided dimensional columns OR if agg_func param is not provided.

classmethod sql_rs_wow_comp(*tbl_name*, *time_col*, *fact_cols*, *dim_cols=False*, *date_join_tbl=False*, *date_join_col=False*, *date_join_time_start_col=False*, *date_join_time_end_col=False*)

Performs a week over week comparison.

Parameters

- **tbl_name** – Table with data to perform calculation on
- **time_col** – Column with sub-year time granularity to compare across years
- **fact_cols** – List of KPI columns to calculate
- **dim_cols** – List of dimensional columns to compare aggregation over (optional)
- **date_join_tbl** – Table with date columns to join to (optional)
- **date_join_col** – Column from date table to join on (optional)
- **date_join_time_start_col** – Column from date table with time granularity start date
- **date_join_time_end_col** – Column from date table with time granularity end date

Returns Analytical SQL query

classmethod sql_rs_yoy_cum_comp(*tbl_name*, *time_col*, *year_col*, *fact_cols*, *agg_func=False*, *dim_cols=False*, *date_join_tbl=False*, *date_join_col=False*, *date_join_time_start_col=False*, *date_join_time_end_col=False*)

Performs a cumulative aggregation over the year as well as provided dimensional columns OR if agg_func param is not provided.

Parameters

- **tbl_name** – Table with data to perform calculation on
- **time_col** – Column with sub-year time granularity to compare across years
- **year_col** – Column with year time granularity
- **fact_cols** – List of KPI columns to calculate
- **agg_func** – Type of aggregation to perform, if not provided will just do a snapshot of current week vs current week
- **dim_cols** – List of dimensional columns to compare aggregation over (optional)
- **date_join_tbl** – Table with date columns to join to (optional)
- **date_join_col** – Column from date table to join on (optional)
- **date_join_time_start_col** – Column from date table with time granularity start date
- **date_join_time_end_col** – Column from date table with time granularity end date

Returns Analytical SQL query

4.5.2 fusetools.comm_tools

Communication services.

Functions

build_html	**param template_folder** Folder containing email templates

fusetools.comm_tools.build_html

fusetools.comm_tools.**build_html**(*template_file*, *data*)

Parameters

- **template_folder** – Folder containing email templates
- **template_file** (str) – Email template to use
- **data** (dict) – Dictionary of key/value pairs for data to populate in template

Return type str

Returns HTML for email template

Classes

SendGrid	SendGrid's API infrastructure.
Twilio	Twilio's API infrastructure.
WhatsApp	Twilio's WhatsApp API infrastructure.

fusetools.comm_tools.SendGrid

class fusetools.comm_tools.**SendGrid**

Bases: object

SendGrid's API infrastructure.

Methods

send_email	Sends an email via SendGrid API.

classmethod send_email(*api_key*, *from_email*, *to_emails*, *subject*, *html_content*)

Sends an email via SendGrid API.

Parameters

- **api_key** – SendGrid API Key.
- **from_email** – Email to send the message from.
- **to_emails** – Email to send the message to.
- **subject** – Email subject line.
- **html_content** – HTML content to send in email.

Returns JSON API call response.

fusetools.comm_tools.Twilio

class fusetools.comm_tools.**Twilio**
Bases: object

Twilio's API infrastructure.

Methods

get_messages	Retrieves a list of message attributes for a given Twilio developer account.
send_message	Sends an SMS/MMS message.

classmethod get_messages(*account_sid*, *auth_token*, *date_sent_before=None*, *date_sent_after=None*, *date_sent=None*, *limit=None*, *from_number=None*, *to_number=None*)
Retrieves a list of message attributes for a given Twilio developer account.

Parameters

- **date_sent_after** –
- **to_number** –
- **from_number** –
- **limit** –
- **date_sent** –
- **date_sent_before** –
- **account_sid** – Twilio developer account ID.
- **auth_token** – Twilio developer authorization token.

Returns Pandas DataFrame of message attributes for a given Twilio developer account.

classmethod send_message(*body*, *from_number*, *to_number*, *account_sid*, *auth_token*, *media_url=False*)
Sends an SMS/MMS message.

Parameters

- **body** – Message body (text).
- **from_number** – Phone number to send the message from.
- **to_number** – Phone number to send the message to.
- **account_sid** – Twilio developer account ID.

- **auth_token** – Twilio developer authorization token.
- **media_url** – Media URL to pass for MMS messages.

Returns API message sent ID.

fusetools.comm_tools.WhatsApp

class fusetools.comm_tools.**WhatsApp**

Bases: object

Twilio's WhatsApp API infrastructure.

Methods

send_message	Sends a WhatsApp message.

classmethod send_message(*body*, *from_number*, *to_number*, *account_sid*, *auth_token*, *media_url=False*)

Sends a WhatsApp message.

Parameters

- **body** – Message body (text).
- **from_number** – Phone number to send the message from.
- **to_number** – Phone number to send the message to.
- **account_sid** – Twilio developer account ID.
- **auth_token** – Twilio developer authorization token.
- **media_url** – Media URL to pass for MMS messages.

Returns API message sent ID.

4.5.3 fusetools.commerce_tools

Marketplace services.

Classes

Discogs	Discogs' API infrastructure.

fusetools.commerce_tools.Discogs

class fusetools.commerce_tools.**Discogs**

Bases: object

Discogs' API infrastructure.

Methods

api_request	Function to perform generic Discogs API requests.
get_authentication_token	Creates a JSON authentication token to use for API calls.

classmethod api_request(*key*, *secret*, *token_path*, *request_type*, *inventory_bytes=False*, *search_cat=False*, *search_string=False*)

Function to perform generic Discogs API requests.

Parameters

- **key** – Discogs API key.
- **secret** – Discogs API secret.
- **token_path** – Authenticated API token.
- **request_type** – API request type.
- **inventory_bytes** – Encoded bytes array for inventory to list.
- **search_cat** – Search category.
- **search_string** – Search string.

Returns API JSON response.

classmethod get_authentication_token(*usr*, *pwd*, *sav_dir*, *chromedriver_path*, *key*, *secret*)

Creates a JSON authentication token to use for API calls.

Parameters

- **usr** – Discogs username.
- **pwd** – Discogs password.
- **sav_dir** – Local filepath to save auth token.
- **chromedriver_path** – Path to Chromedriver instance for Selenium.
- **key** – Discogs API key.
- **secret** – Discogs API secret.

Returns JSON authentication token.

4.5.4 fusetools.dashboard_tools

Dashboards and data visualization applications.

Classes

Tableau	Functions for interacting with the Tableau Server API.

fusetools.dashboard_tools.Tableau

class fusetools.dashboard_tools.**Tableau**
Bases: object

Functions for interacting with the Tableau Server API.

Methods

auth_tableau_server	Authenticates a connection to a Tableau Server domain.
get_all_sever_items	Retrieves all projects on an organization's Tableau server account.
get_tableau_server_obj_id	Returns the name and object id for a requested object name.
make_tableau_datasource_schema	Converts a Pandas DataFrame of column types & builds a Tableau schema list.
make_tableau_hyperfile	Creates a Tableau hyperfile (data source) to load to Tableau Server.
push_tableau_hyperfile	Pushes a Tableau hyperfile to Tableau Server to create a data source.

classmethod auth_tableau_server(*server_address*, *username*, *pwd*)
Authenticates a connection to a Tableau Server domain.

Parameters

- **server_address** – Tableau server domain address.
- **username** – Tableau server username.
- **pwd** – Tableau server password.

Returns Tableau authentication object & Tableau server object.

classmethod get_all_sever_items(*server*, *tableau_auth*)
Retrieves all projects on an organization's Tableau server account.

Parameters

- **server** – Tableau server object.
- **tableau_auth** – Tableau authentication object.

Returns Projects on an organization's Tableau server account.

classmethod get_tableau_server_obj_id(*server*, *tableau_auth*, *obj_name*)
Returns the name and object id for a requested object name.

Parameters

- **server** – Tableau server domain address.
- **tableau_auth** – Authenticated Tableau server object.
- **obj_name** – Name to search for.

Returns Name and object id for a requested object name.

classmethod make_tableau_datasource_schema(*df_schema*)
Converts a Pandas DataFrame of column types & builds a Tableau schema list.

Parameters **df_schema** – Pandas DataFrame of column types.

Returns List of Tableau sqlType objects.

classmethod make_tableau_hyperfile(*df*, *save_dir*, *hyperfile_name*, *tableau_schema_cols*)
Creates a Tableau hyperfile (data source) to load to Tableau Server.

Parameters

- **df** – Pandas DataFrame to be loaded to Tableau server.
- **save_dir** – Local filepath to save Pandas DataFrame as CSV.
- **hyperfile_name** – Filename for hyperfile.
- **tableau_schema_cols** – List of columns for the hyperfile with designated Tableau data types.

Returns Tableau hyperfile.

classmethod push_tableau_hyperfile(*hyperfile_path*, *server*, *tableau_auth*, *project_folder_id*)
Pushes a Tableau hyperfile to Tableau Server to create a data source.

Parameters

- **hyperfile_path** – Path to Tableau hyperfile.
- **server** – Tableau server object.
- **tableau_auth** – Tableau authentication object.
- **project_folder_id** – ID of Tableau server project folder.

Returns Data source object for loaded hyperfile.

4.5.5 fusetools.date_tools

Functions for interacting with Python date objects.

Functions

get_last_dow	Get the last date for a given day of the week (ex: Sunday, Monday)
get_rptg_mon	Get the reporting year/month combination for a given date.
get_rptg_qtr	Get the reporting year/quarter combination for a given date.
get_rptg_week	Get the reporting year/week combination for a given date.
get_rptg_yr	Get the reporting year for a given date.

fusetools.date_tools.get_last_dow

fusetools.date_tools.**get_last_dow**(*dow*, *ref_date=False*)

Get the last date for a given day of the week (ex: Sunday, Monday)

Parameters

- **dow** – Day of week to pull date for.
- **ref_date** – Date to provide a reference for (optional).

Returns Last date for a given day of the week.

fusetools.date_tools.get_rptg_mon

fusetools.date_tools.**get_rptg_mon**(*ref_date=False*)

Get the reporting year/month combination for a given date.

Parameters **ref_date** – Date to provide a reference for (optional).

Returns Reporting year/month combination.

fusetools.date_tools.get_rptg_qtr

fusetools.date_tools.**get_rptg_qtr**(*ref_date=False*)

Get the reporting year/quarter combination for a given date.

Parameters **ref_date** – Date to provide a reference for (optional).

Returns Reporting year/quarter combination.

fusetools.date_tools.get_rptg_week

fusetools.date_tools.**get_rptg_week**(*ref_date=False*)

Get the reporting year/week combination for a given date.

Parameters **ref_date** – Date to provide a reference for (optional).

Returns Reporting year/week combination for a given date.

fusetools.date_tools.get_rptg_yr

fusetools.date_tools.**get_rptg_yr**(*ref_date=False*)
Get the reporting year for a given date.

Parameters **ref_date** – Date to provide a reference for (optional).

Returns Reporting year for a given date.

4.5.6 fusetools.db_conn_tools

Database connections and engines.

Classes

MySQL	MySQL database connections.
Oracle	Oracle database connections.
Postgres	Postgres database connections.
Redshift	Redshift database connections.
TeraData	Teradata database connections.

fusetools.db_conn_tools.MySQL

class fusetools.db_conn_tools.**MySQL**
Bases: object

MySQL database connections.

Methods

eng_mysql	Create a MySQL database engine object via SqlAlchemy.

classmethod eng_mysql(*usr*, *pwd*, *host=None*, *db=None*)
Create a MySQL database engine object via SqlAlchemy.

Parameters

- **usr** – MySQL username.
- **pwd** – MySQL password.

Returns MySQL database engine object.

fusetools.db_conn_tools.Oracle

class fusetools.db_conn_tools.**Oracle**
Bases: object

Oracle database connections.

Methods

con_oracle	Create an Oracle connection object.
eng_oracle	Create an Oracle database engine object using SqlAlchemy.
eng_oracle_addr	Create an Oracle database engine object using JDBC.

classmethod con_oracle(*usr*, *pwd*, *dbname*)
Create an Oracle connection object.

Parameters

- **usr** – Oracle username.
- **pwd** – Oracle password.

Returns Oracle database connection object.

classmethod eng_oracle(*usr*, *pwd*, *dbname*)
Create an Oracle database engine object using SqlAlchemy.

Parameters

- **usr** – Oracle username.
- **pwd** – Oracle password.

Returns Oracle database engine object.

classmethod eng_oracle_addr(*usr*, *pwd*, *host*, *port*, *dbname*)
Create an Oracle database engine object using JDBC. Requires cx_oracle package.

Parameters

- **usr** – Oracle username.
- **pwd** – Oracle password.

Returns Oracle database engine object.

fusetools.db_conn_tools.Postgres

class fusetools.db_conn_tools.**Postgres**
Bases: object

Postgres database connections.

Methods

con_postgres	Create a Postgres database connection object.
eng_postgres	Create a Postgres database engine object.

classmethod con_postgres(*host*, *db*, *usr*, *pwd*, *port=None*)
Create a Postgres database connection object.

Parameters

- **host** – Hostname for Postgres database.
- **db** – Name of Postgres database.
- **usr** – Postgres username.
- **pwd** – Postgres password.

Returns A Postgres database connection object.

classmethod eng_postgres(*usr*, *pwd*, *port*, *db*, *ssl_str=''*, *host=None*)
Create a Postgres database engine object.

Parameters

- **usr** – A Postgres username.
- **pwd** – A Postgres password.
- **port** – Port for Postgres database.

Returns A Postgres database connection.

fusetools.db_conn_tools.Redshift

class fusetools.db_conn_tools.**Redshift**
Bases: object

Redshift database connections.

Methods

conn_rs_pg	Create a Redshift database connection object via psycopg2 package.
conn_rs_sa	Create a Redshift database connection object via SqlAlchemy.

classmethod conn_rs_pg(*db*, *host*, *port*, *user*, *pwd*)
Create a Redshift database connection object via psycopg2 package.

Parameters

- **db** – Redshift database name.
- **host** – Redshift database host address.
- **port** – Redshift database post.
- **user** – Redshift username.
- **pwd** – Redshift password.

Returns Redshift database connection object.

classmethod conn_rs_sa(*db*, *host*, *port*, *user*, *pwd*)
Create a Redshift database connection object via SqlAlchemy.

Parameters

- **db** – Redshift database name.
- **host** – Redshift database host address.

- **port** – Redshift database post.
- **user** – Redshift username.
- **pwd** – Redshift password.

Returns Redshift database connection object.

fusetools.db_conn_tools.TeraData

class fusetools.db_conn_tools.**TeraData**
Bases: object

Teradata database connections.

Methods

conn_pyodbc	Create a Teradata database connection object via ODBC.
conn_sa	Create a Teradata database connection object via SqlAlchemy.
conn_td	Create a Teradata database connection object via Teradata Python package.

classmethod conn_pyodbc(*drivername*, *host*, *usr*, *pwd*)
Create a Teradata database connection object via ODBC.

Parameters

- **drivername** – Teradata driver name for ODBC.
- **host** – Teradata hostname.
- **usr** – Teradata username.
- **pwd** – Teradata password.

Returns Teradata database connection object.

classmethod conn_sa(*drivername*, *host*, *usr*, *pwd*)
Create a Teradata database connection object via SqlAlchemy.

Parameters

- **drivername** – Teradata database driver name for ODBC.
- **host** – Teradata database hostname.
- **usr** – Teradata username.
- **pwd** – Teradata password.

Returns Teradata database connection object.

classmethod conn_td(*host*)
Create a Teradata database connection object via Teradata Python package.

Parameters **host** – Teradata hostname.

Returns Returns a Teradata database connection object.

4.5.7 fusetools.db_etl_tools

Database connections and engines.




Classes

Generic	Generic functions for SQL queries and ETL.
Oracle	Generic functions for Oracle SQL queries and ETL.
Postgres	Generic functions for Postgres SQL queries and ETL.
Redshift	Generic functions for Redshift SQL queries and ETL.
Teradata	Generic functions for Teradata SQL queries and ETL.

fusetools.db_etl_tools.Generic

class fusetools.db_etl_tools.**Generic**
Bases: object

Generic functions for SQL queries and ETL.

Methods

db_apply_schema	Converts Pandas DataFrame columns based on schema DataFrame provided.
make_db_cols	Returns a Pandas DataFrame column names that are converted for database standards.
make_db_schema	Creates a mapping of Pandas data types to SQL data types.
make_groupby	Creates a dynmaically generated GROUP BY clause for a given SQL statement.
run_query	Executes a SQL query.

classmethod db_apply_schema(*df*, *schema_df*)
Converts Pandas DataFrame columns based on schema DataFrame provided.

Parameters

- **df** – A Pandas DataFrame with column types to be converted.
- **schema_df** – A Pandas DataFrame of columns with corresponding SQL data types.

Returns Pandas DataFrame with columns converted to SQL schema.

classmethod make_db_cols(*df*)
Returns a Pandas DataFrame column names that are converted for database standards.

Parameters **df** – A Pandas DataFrame with columns to be transformed

Returns Pandas DataFrame column names that are converted for database standards.

classmethod make_db_schema(*df*)
Creates a mapping of Pandas data types to SQL data types.

Parameters **df** – A Pandas DataFrame with column types to be converted.

Returns A Pandas DataFrame of columns with corresponding SQL data types.

classmethod make_groupby(*sql*, *dim_fact_delim*)
Creates a dynmaically generated GROUP BY clause for a given SQL statement.

Parameters

- **sql** – SQL statement provided.
- **dim_fact_delim** – Delimiter between selected columns.

Returns A complete SQL statement with dynamically generated GROUP BY clause.

classmethod run_query(*engine*, *sql*)
Executes a SQL query.

Parameters

- **engine** – A database engine object.
- **sql** – A SQL statement to be executed.

Returns Time for execution of SQL query.

fusetools.db_etl_tools.Oracle

class fusetools.db_etl_tools.**Oracle**
Bases: object

Generic functions for Oracle SQL queries and ETL.

Methods

get_oracle_date	Converts a date to an Oracle date of format "DD-MMM-YYY"
get_orcl_date	Converts a date to an Oracle date of format "DD-MMM-YYY".
insert_exec	Executes a provided SQL statement.
insert_tbl	Executes an INSERT INTO statement for a given Pandas DataFrame.
make_tbl	Provides a CREATE TABLE SQL statement for a given Pandas DataFrame.
make_tbl_complete	Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.
make_tbl_complete_force	Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.
orcl_tbl_varchar_convert	Converts a set of columns to VARCHAR(300) for a given Oracle table.

classmethod get_oracle_date(*date*)
Converts a date to an Oracle date of format "DD-MMM-YYY"

Parameters **date** – A provided date.

Returns An Oracle database date.

classmethod get_orcl_date(*dat*)
Converts a date to an Oracle date of format "DD-MMM-YYY".

Parameters **dat** – A provided date column of a Pandas Series.

Returns An Oracle database date.

classmethod insert_exec(*sql*, *conn*, *df*)
Executes a provided SQL statement.

Parameters

- **sql** – A provided SQL query.
- **conn** – A database connection.
- **df** – A Pandas DataFrame.

Returns Nothing.

classmethod insert_tbl(*df*, *tbl_name*)
Executes an INSERT INTO statement for a given Pandas DataFrame.

Parameters

- **df** – A Pandas DataFrame with values to be inserted.
- **tbl_name** – An Oracle table for Pandas DataFrame to be inserted into.

Returns SQL for INSERT INTO statement.

classmethod make_tbl(*df*, *tbl_name*)
Provides a CREATE TABLE SQL statement for a given Pandas DataFrame.

Parameters

- **df** – A Pandas DataFrame to be added as an Oracle table.
- **tbl_name** – Oracle table name to be created.

Returns CREATE TABLE SQL statement.

classmethod make_tbl_complete(*df*, *tbl_name*, *eng*, *conn*, *subcols=False*, *chunks=False*, *chunks_delay=False*)
Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.

Parameters

- **df** – Pandas DataFrame to create a table from.
- **tbl_name** – Name of table to be created.
- **eng** – Oracle database engine object.
- **conn** – Oracle database connection object.
- **subcols** – A list of columns of the Pandas DataFrame to apply operations on.
- **chunks** – Number of chunks to split Pandas DataFrame into.
- **chunks_delay** – Delay between chunk's INSERT statement.

Returns Print statements outline sequential SQL statements executed.

classmethod make_tbl_complete_force(*df*, *tbl_name*, *eng*, *conn*, *attempt_n*, *subcols=False*, *chunks=False*, *chunks_delay=False*)
Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.

Parameters

- **df** – Pandas DataFrame to create a table from.
- **tbl_name** – Name of table to be created.
- **eng** – Oracle database engine object.
- **conn** – Oracle database connection object.
- **attempt_n** – Number of times to attempt to run INSERT statement.
- **subcols** – A list of columns of the Pandas DataFrame to apply operations on.
- **chunks** – Number of chunks to split Pandas DataFrame into.
- **chunks_delay** – Delay between chunk's INSERT statement.

Returns Print statements outline sequential SQL statements executed.

classmethod orcl_tbl_varchar_convert(*tbl_name*, *convert_cols*, *engine*)
Converts a set of columns to VARCHAR(300) for a given Oracle table.

Parameters

- **tbl_name** – Oracle table name.
- **convert_cols** – List of columns to convert.
- **engine** – Oracle database engine.

Returns Printed ALTER table statements for each column.

fusetools.db_etl_tools.Postgres

class fusetools.db_etl_tools.**Postgres**
Bases: object

Generic functions for Postgres SQL queries and ETL.

Methods

insert_df_pg	Executes an INSERT INTO statement for a given Pandas DataFrame into a Postgres table..
insert_val_pg	Creates SQL to run an INSERT operation of a given Postgres table.
make_df_tbl_pg	Creates SQL to run a CREATE TABLE statement based on a Pandas DataFrame.
make_tbl_complete_pg	Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.
run_query_pg	Executes a SQL statement with a Postgres database connection.
sequential_load_pg	**param override**
sequential_load_pg_wk	**param rptg_dates**
upsert_tbl_pg	Creates SQL to run an UPSERT (INSERT new records or UPDATE existing records) operation of a given Postgres table.

continues on next page

Table 25 – continued from previous page

upsert_val_pg	Creates SQL to run an UPSERT (INSERT new records or UPDATE existing records) operation of a given Postgres table.

classmethod insert_df_pg(*cursor*, *conn*, *df*, *tbl_name*, *return_statement=None*)
Executes an INSERT INTO statement for a given Pandas DataFrame into a Postgres table..

Parameters

- **cursor** – Postgres database cursor object.
- **conn** – Postgres database connection object.
- **df** – Pandas DataFrame to insert into a Postgres table.
- **tbl_name** – Postgres table name.

Returns Elapsed time to execute query.

classmethod insert_val_pg(*col_list*, *val_list*, *tbl_name*)
Creates SQL to run an INSERT operation of a given Postgres table.

Parameters

- **col_list** – List of columns to INSERT or UPDATE.
- **val_list** – List of values to INSERT or UPDATE.
- **tbl_name** – Name of Postgres table.

Returns SQL to run an INSERT statement.

classmethod make_df_tbl_pg(*tbl_name*, *df*)
Creates SQL to run a CREATE TABLE statement based on a Pandas DataFrame.

Parameters

- **tbl_name** – Postgres table name.
- **df** – Pandas DataFrame.

Returns CREATE TABLE SQL statement.

classmethod make_tbl_complete_pg(*df*, *tbl_name*, *conn*, *cursor*, *batch_size=False*)
Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.

Parameters

- **df** – Pandas DataFrame to create a table from.
- **tbl_name** – Name of table to be created.
- **conn** – Postgres database connection object.
- **cursor** – Postgres database cursor object.
- **batch_size** – Records to load per batch.

Returns Elapsed time to execute query.

classmethod run_query_pg(*conn*, *sql*)
Executes a SQL statement with a Postgres database connection.

Parameters

- **conn** – Postgres database connection object,

- **sql** – SQL Statement to execute.

Returns Elapsed time to execute query.

classmethod sequential_load_pg(*override*, *tgt_tbl*, *conn*, *dt_start*, *dt_end*, *saved_day_id_range_placeholder*, *dt1_interval*, *dt2_interval*, *sql_loop_fn*, *sql_loop_fn_type*, *filter_day_id_field1=False*, *sql_loop_fn_dt_placeholder1=False*, *filter_day_id_field2=False*, *filter_id_type2=False*, *sql_loop_fn_dt_placeholder2=False*, *filter_day_id_field3=False*, *filter_id_type3=False*, *sql_loop_fn_dt_placeholder3=False*, *loop_src1=False*, *loop_src2=False*, *loop_src3=False*, *log_dir=False*)

Parameters

- **override** –
- **tgt_tbl** –
- **conn** –
- **dt_start** –
- **dt_end** –
- **saved_day_id_range_placeholder** –
- **dt1_interval** –
- **dt2_interval** –
- **sql_loop_fn** –
- **sql_loop_fn_type** –
- **filter_day_id_field1** –
- **sql_loop_fn_dt_placeholder1** –
- **filter_day_id_field2** –
- **filter_id_type2** –
- **sql_loop_fn_dt_placeholder2** –
- **filter_day_id_field3** –
- **filter_id_type3** –
- **sql_loop_fn_dt_placeholder3** –
- **loop_src1** –
- **loop_src2** –
- **loop_src3** –
- **log_dir** –

Returns

classmethod sequential_load_pg_wk(*rptg_dates*, *override*, *tgt_tbl*, *conn*, *rptg_wk*, *rptg_wk_start*, *rptg_wk_end*, *sql_loop_fn*, *filter_dt_field1=False*, *filter_dt_type1=False*, *filter_dt_placeholder1=False*, *filter_dt_field2=False*, *filter_dt_type2=False*, *filter_dt_placeholder2=False*, *filter_dt_field3=False*, *filter_dt_type3=False*, *filter_dt_placeholder3=False*, *log_dir=False*)

Parameters

- **rptg_dates** –
- **override** –
- **tgt_tbl** –
- **conn** –
- **rptg_wk** –
- **rptg_wk_start** –
- **rptg_wk_end** –
- **sql_loop_fn** –
- **filter_dt_field1** –
- **filter_dt_type1** –
- **filter_dt_placeholder1** –
- **filter_dt_field2** –
- **filter_dt_type2** –
- **filter_dt_placeholder2** –
- **filter_dt_field3** –
- **filter_dt_type3** –
- **filter_dt_placeholder3** –
- **log_dir** –

Returns

classmethod upsert_tbl_pg(*src_tbl*, *tgt_tbl*, *src_join_cols*, *src_insert_cols*, *src_update_cols=False*, *update_compare_cols=False*)

Creates SQL to run an UPSERT (INSERT new records or UPDATE existing records) operation of a given Postgres table.

Parameters

- **src_tbl** – Postgres source table that contains data to be merged from.
- **tgt_tbl** – Postgres target table to receive UPSERT operation.
- **src_join_cols** – Columns to use to join source and target tables.
- **src_insert_cols** – Columns to be inserted from source table.
- **src_update_cols** – Columns to be updated from source table.
- **update_compare_cols** – Columns to use to compare values across source and target tables.

Returns A SQL Insert statement and a SQL Update statement.

classmethod upsert_val_pg(*col_list*, *val_list*, *tbl_name*, *constraint_col*)
Creates SQL to run an UPSERT (INSERT new records or UPDATE existing records) operation of a given Postgres table.

Parameters

- **col_list** – List of columns to INSERT or UPDATE.
- **val_list** – List of values to INSERT or UPDATE.
- **constraint_col** – Column/value logic to check against for INSERT or UPDATE.
- **tbl_name** – Name of Postgres table.

Returns SQL to run an UPSERT statement.

fusetools.db_etl_tools.Redshift

class fusetools.db_etl_tools.**Redshift**
Bases: object

Generic functions for Redshift SQL queries and ETL.

Methods

insert_df_rs	Executes an INSERT INTO statement for a given Pandas DataFrame into a Redshift table..
insert_val_rs	Creates SQL to run an INSERT operation of a given Redshift table.
make_df_tbl_rs	Creates SQL to run a CREATE TABLE statement based on a Pandas DataFrame.
make_tbl_complete_rs	Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.
run_query_rs	Executes a SQL statement with a Redshift database connection.
sequential_load_rs	**param override**
sequential_load_rs_wk	**param rptg_dates**
upsert_tbl_rs	Creates SQL to run an UPSERT (INSERT new records or UPDATE existing records) operation of a given Redshift table.

classmethod insert_df_rs(*cursor*, *conn*, *df*, *tbl_name*)

Executes an INSERT INTO statement for a given Pandas DataFrame into a Redshift table..

Parameters

- **cursor** – Redshift database cursor object.
- **conn** – Redshift database connection object.
- **df** – Pandas DataFrame to insert into a Redshift table.
- **tbl_name** – Redshift table name.

Returns Elapsed time to execute query.

classmethod insert_val_rs(*col_list*, *val_list*, *tbl_name*)

Creates SQL to run an INSERT operation of a given Redshift table.

Parameters

- **col_list** – List of columns to INSERT or UPDATE.
- **val_list** – List of values to INSERT or UPDATE.
- **tbl_name** – Name of Postgres table.

Returns SQL to run an INSERT statement.

classmethod make_df_tbl_rs(*tbl_name*, *df*)

Creates SQL to run a CREATE TABLE statement based on a Pandas DataFrame.

Parameters

- **tbl_name** – Redshift table name.
- **df** – Pandas DataFrame.

Returns CREATE TABLE SQL statement.

classmethod make_tbl_complete_rs(*df*, *tbl_name*, *conn*, *cursor*, *batch_size=False*)

Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.

Parameters

- **df** – Pandas DataFrame to create a table from.
- **tbl_name** – Name of table to be created.
- **conn** – Redshift database connection object.
- **cursor** – Redshift database cursor object.
- **batch_size** – Records to load per batch.

Returns Elapsed time to execute query.

classmethod run_query_rs(*conn*, *sql*)

Executes a SQL statement with a Redshift database connection.

Parameters

- **conn** – Redshift database connection object,
- **sql** – SQL Statement to execute.

Returns Elapsed time to execute query.

classmethod sequential_load_rs(*override*, *tgt_tbl*, *conn*, *dt_start*, *dt_end*, *saved_day_id_range_placeholder*, *dt1_interval*, *dt2_interval*, *sql_loop_fn*, *sql_loop_fn_type*, *filter_day_id_field1=False*, *sql_loop_fn_dt_placeholder1=False*, *filter_day_id_field2=False*, *filter_id_type2=False*, *sql_loop_fn_dt_placeholder2=False*, *filter_day_id_field3=False*, *filter_id_type3=False*, *sql_loop_fn_dt_placeholder3=False*, *loop_src1=False*, *loop_src2=False*, *loop_src3=False*, *log_dir=False*)

Parameters

- **override** –
- **tgt_tbl** –
- **conn** –
- **dt_start** –
- **dt_end** –
- **saved_day_id_range_placeholder** –
- **dt1_interval** –
- **dt2_interval** –
- **sql_loop_fn** –
- **sql_loop_fn_type** –
- **filter_day_id_field1** –
- **sql_loop_fn_dt_placeholder1** –
- **filter_day_id_field2** –
- **filter_id_type2** –
- **sql_loop_fn_dt_placeholder2** –
- **filter_day_id_field3** –
- **filter_id_type3** –
- **sql_loop_fn_dt_placeholder3** –
- **loop_src1** –
- **loop_src2** –
- **loop_src3** –
- **log_dir** –

Returns

classmethod sequential_load_rs_wk(*rptg_dates*, *override*, *tgt_tbl*, *conn*, *rptg_wk*, *rptg_wk_start*, *rptg_wk_end*, *sql_loop_fn*, *filter_dt_field1=False*, *filter_dt_type1=False*, *filter_dt_placeholder1=False*, *filter_dt_field2=False*, *filter_dt_type2=False*, *filter_dt_placeholder2=False*, *filter_dt_field3=False*, *filter_dt_type3=False*, *filter_dt_placeholder3=False*, *log_dir=False*)

Parameters

- **rptg_dates** –
- **override** –
- **tgt_tbl** –
- **conn** –
- **rptg_wk** –
- **rptg_wk_start** –
- **rptg_wk_end** –
- **sql_loop_fn** –
- **filter_dt_field1** –
- **filter_dt_type1** –
- **filter_dt_placeholder1** –
- **filter_dt_field2** –
- **filter_dt_type2** –
- **filter_dt_placeholder2** –
- **filter_dt_field3** –
- **filter_dt_type3** –
- **filter_dt_placeholder3** –
- **log_dir** –

Returns

classmethod upsert_tbl_rs(*src_tbl*, *tgt_tbl*, *src_join_cols*, *src_insert_cols*, *src_update_cols=False*, *update_compare_cols=False*)

Creates SQL to run an UPSERT (INSERT new records or UPDATE existing records) operation of a given Redshift table.

Parameters

- **src_tbl** – Redshift source table that contains data to be merged from.
- **tgt_tbl** – Redshift target table to receive UPSERT operation.
- **src_join_cols** – Columns to use to join source and target tables.
- **src_insert_cols** – Columns to be inserted from source table.
- **src_update_cols** – Columns to be updated from source table.
- **update_compare_cols** – Columns to use to compare values across source and target tables.

Returns A SQL Insert statement and a SQL Update statement.

fusetools.db_etl_tools.Teradata

class fusetools.db_etl_tools.**Teradata**
Bases: object

Generic functions for Teradata SQL queries and ETL.

Methods

insert_td	Executes an INSERT INTO statement for a given Pandas DataFrame.
make_tbl_complete_td	Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.
make_tbl_td	Creates SQL to run a CREATE TABLE statement based on a Pandas DataFrame.
run_query_td	Executes a SQL statement with a Teradata database connection.

classmethod insert_td(*tbl*, *df*, *conn*, *batch_size=False*, *date_cols=False*)
Executes an INSERT INTO statement for a given Pandas DataFrame.

Parameters

- **tbl** – Teradata table name.
- **df** – Pandas DataFrame.
- **conn** – Teradata connection object.
- **batch_size** – Records to load per batch.
- **date_cols** – A list of date columns to convert to Pandas datetime.

Returns Printed SQL statements for each step.

classmethod make_tbl_complete_td(*df*, *tbl_name*, *conn*, *batch_size=False*)
Executes a series of SQL statements to CREATE and INSERT into a table from a Pandas DataFrame.

Parameters

- **df** – Pandas DataFrame to create a table from.
- **tbl_name** – Name of table to be created.
- **conn** – Teradata database connection object.
- **batch_size** – Records to load per batch.

Returns Elapsed time to execute query.

classmethod make_tbl_td(*df*, *tbl_name*)
Creates SQL to run a CREATE TABLE statement based on a Pandas DataFrame.

Parameters

- **df** – Pandas DataFrame.
- **tbl_name** – Teradata table name.

Returns CREATE TABLE SQL statement.

classmethod run_query_td(*conn*, *sql*)
Executes a SQL statement with a Teradata database connection.

Parameters

- **conn** – Teradata database connection object.
- **sql** – SQL statement to execute.

Returns Elapsed time to execute query.

4.5.8 fusetools.gsuite_tools

Google Suite Tools.

Classes

GDrive	Functions for interacting with Google Drive.
GMail	Functions for interacting with GMail.
GSheets	Functions for interacting with Google Sheets.

fusetools.gsuite_tools.GDrive

class fusetools.gsuite_tools.**GDrive**
Bases: object

Functions for interacting with Google Drive.

Methods

authorize_credentials	Creates an authorized credentials object for Google Drive.
create_service_serv_acct	Creates a GDrive authenticated credentials object.
create_upload_folder	Creates a folder in Google Drive.
download_file	Downloads a file from a Google Drive account.
get_all_google_items	Get all files in a Google Drive (or folder if specified).
get_file_revisions	**param file_id**
upload_files	Uploads files to a folder on Google Drive.

classmethod authorize_credentials(*cred_path*, *token_path*)
Creates an authorized credentials object for Google Drive.

Parameters

- **cred_path** – Local path to GSuite credentials object.
- **token_path** – Local path to GSuite authorization token.

Returns Authorized credentials object for GSuite.

classmethod create_service_serv_acct(*member_acct_email*, *token_path*)
Creates a GDrive authenticated credentials object.

Parameters

- **member_acct_email** – GDrive service acct email address.
- **token_path** – Path to GDrive authentication token.

Returns Return GDrive authenticated credentials object.

classmethod create_upload_folder(*folder_name*, *credentials*, *overwrite_folder=False*, *parent_id=None*)
Creates a folder in Google Drive.

Parameters

- **folder_name** – Name of folder to create.

- **credentials** – GSuite credentials object.
- **overwrite_folder** – Whether or not to delete and re-create the folder.
- **parent_id** – Id of parent folder (optional).

Returns Created folder information.

classmethod download_file(*file_id*, *file_name*, *credentials*, *save_local=False*)
Downloads a file from a Google Drive account.

Parameters

- **file_name** – Name of file to download from Google Drive.
- **save_local** – whether to return file in bytes or save locally (default is bytes).
- **file_id** – ID for Google Drive file.
- **credentials** – GSuite credentials object.

Returns Downloaded file from Google Drive.

classmethod get_all_google_items(*credentials*, *page_size=None*, *folder_id=None*)
Get all files in a Google Drive (or folder if specified).

Parameters

- **page_size** –
- **limit** –
- **credentials** – GSuite credentials object.
- **folder_id** – GDrive folder to search in (optional)

Returns Pandas DataFrame of files in a Google Drive.

classmethod get_file_revisions(*file_id*, *credentials*)

Parameters

- **file_id** –
- **credentials** –

Returns

classmethod upload_files(*folder_id*, *upload_filepaths*, *credentials*, *upload_filenames=False*, *upload_from_memory=False*)
Uploads files to a folder on Google Drive.

Parameters

- **folder_id** – ID of folder to upload files into.
- **credentials** – GSuite credentials object.
- **upload_filepaths** – Filepaths for files to upload.
- **upload_filenames** – Filenames for files, must match length of filepaths (optional). Otherwise uses filepath names.

Returns Confirmation of files being uploaded.

fusetools.gsuite_tools.GMail

class fusetools.gsuite_tools.**GMail**
Bases: object

Functions for interacting with GMail.

Methods

create_service	Creates an authenticated service object for GMail.
create_service_serv_acct	Creates a GMail authenticated credentials object.
download_email_attachment	Downloads email attachments for a given emails.
get_emails	Returns a Pandas DataFrame of received email details.
get_mailbox_labels	**param service**
send_email	Sends an email via GMail.

classmethod create_service(*cred_path*, *token_path=None*, *working_dir=None*)
Creates an authenticated service object for GMail.

Parameters

- **cred_path** – Path to Google credentials object.
- **token_path** – Path of authentication token.
- **working_dir** – Path of working directory to store token if token path not specified.

Returns Authenticated service object for GMail

classmethod create_service_serv_acct(*member_acct_email*, *token_path*)
Creates a GMail authenticated credentials object.

Parameters

- **member_acct_email** – GSuite service acct email address.
- **token_path** – Path to GSuite authentication token.

Returns Return GMail authenticated credentials object.

classmethod download_email_attachment(*service*, *msg_id*, *sav_dir=False*, *user_id='me'*, *save_memory=False*, *req_limit=1*)
Downloads email attachments for a given emails.

Parameters

- **service** – Authenticated service object for GMail.

- **msg_id** – ID of message to download attachments for.
- **sav_dir** – Directory to save an attachment into.
- **user_id** – User ID to pull emails for, default="me".

Returns Downloaded email attachments.

classmethod get_emails(*service*, *label_ids=False*, *custom_tree_branch_list=False*, *user_id='me'*, *req_limit=1*)

Returns a Pandas DataFrame of received email details.

Parameters

- **service** – Authenticated service object for GMail.
- **user_id** – User ID to pull emails for, default="me"
- **label_ids** – GMail label IDs to pull messages for, default="INBOX"
- **custom_tree_branch_list** – List of branch elements to navigate HTML body data (ex: [0,1,0])

Returns Pandas DataFrame of received email details.

classmethod get_mailbox_labels(*service*, *user_id='me'*, *req_limit=1*)

Parameters

- **service** –
- **user_id** –

Returns

classmethod send_email(*service*, *to*, *sender*, *subject*, *message_text*, *attachments=False*, *attachments_bytes=False*, *attachment_types=False*, *attachment_names=False*, *message_is_html=False*, *req_limit=1*)

Sends an email via GMail.

Parameters

- **service** – Authenticated service object for GMail.
- **to** – Target email address.
- **sender** – Sender email address.
- **subject** – Subject for email.
- **message_text** – Message body text.
- **attachments** – List of message attachments.
- **message_is_html** – Whether or not message_text is in HTML format.

Returns Sent message.

fusetools.gsuite_tools.GSheets

class fusetools.gsuite_tools.**GSheets**
Bases: object

Functions for interacting with Google Sheets.

Methods

add_google_sheet_comment	**param spreadsheet_id**
add_google_sheet_tab	Adds a tab to a Google Sheet.
bulk_add_google_sheet_comment	**param spreadsheet_id**
bulk_update_cell_background_color	**param spreadsheet_id**
clear_google_sheet_data	
create_service_serv_acct	Creates a GSheets authenticated credentials object.
delete_google_sheet_data	Deletes data from a Google Sheet.
delete_google_sheet_tab	Adds a tab to a Google Sheet.
drop_duplicates	**param spreadsheet_id**
freeze_rows_cols	Freezes the rows of a given Google Sheet.
get_google_sheet	Gets data from a Google Sheet.
get_google_sheet_metadata	**param include_grid_data**
get_google_sheet_tabs	Get the names and IDs of tabs for a given Google Sheet.
group_sheet_cols_rows	**param spreadsheet_id**
make_google_sheet	Creates a Google Sheet in one's GSuite account.
sort_google_sheet	
update_cell_background_color	**param spreadsheet_id**
update_google_sheet_df	Uploads a Pandas DataFrame to a Google Sheet.

continues on next page

Table 31 – continued from previous page

update_google_sheet_val	Updates a google spreadsheet cell with a value.

classmethod add_google_sheet_comment(*spreadsheet_id*, *tab_id*, *note_contents*, *start_row_idx*, *end_row_idx*, *start_col_idx*, *end_col_idx*, *credentials*, *req_limit=1*)

Parameters

- **spreadsheet_id** –
- **tab_id** –
- **credentials** –

Returns

classmethod add_google_sheet_tab(*spreadsheet_id*, *tab_name*, *credentials*, *req_limit=1*)

Adds a tab to a Google Sheet.

Parameters

- **spreadsheet_id** – Id of Google Sheet to add tab to.
- **tab_name** – Name of the tab to be added.
- **credentials** – GSuite credentials object.

Returns Result object for API call.

classmethod bulk_add_google_sheet_comment(*spreadsheet_id*, *request_list*, *credentials*, *req_limit=1*)

Parameters

- **spreadsheet_id** –
- **tab_id** –
- **credentials** –

Returns

classmethod bulk_update_cell_background_color(*spreadsheet_id*, *credentials*, *request_list*, *req_limit=1*)

Parameters

- **spreadsheet_id** –
- **sheet_id** –
- **row_idx_start** –
- **row_idx_end** –
- **col_idx_start** –
- **col_idx_end** –
- **credentials** –
- **dimension** –

Returns

classmethod create_service_serv_acct(*member_acct_email*, *token_path*)

Creates a GSheets authenticated credentials object.

Parameters

- **member_acct_email** – GSuite service acct email address.
- **token_path** – Path to GSuite authentication token.

Returns Return GSheets authenticated credentials object.

classmethod delete_google_sheet_data(*spreadsheet_id*, *sheet_id*, *idx_start*, *idx_end*, *credentials*, *dimension='ROWS'*, *req_limit=1*)

Deletes data from a Google Sheet.

Parameters

- **spreadsheet_id** – ID of spreadsheet to update.
- **idx_start** – Starting index of row/column for range to delete.
- **idx_end** – Ending index of row/column for range to delete.
- **credentials** –
- **dimension** –

Returns

classmethod delete_google_sheet_tab(*spreadsheet_id*, *tab_id*, *credentials*, *req_limit=1*)

Adds a tab to a Google Sheet.

Parameters

- **spreadsheet_id** – Id of Google Sheet to add tab to.
- **tab_name** – Name of the tab to be added.
- **credentials** – GSuite credentials object.

Returns Result object for API call.

classmethod drop_duplicates(*spreadsheet_id*, *sheet_id*, *credentials*, *dup_idx_start*, *dup_idx_end*, *row_idx_start*, *row_idx_end*, *col_idx_start*, *col_idx_end*, *rows_columns='COLUMNS'*, *req_limit=1*)

Parameters

- **spreadsheet_id** –
- **sheet_id** –
- **credentials** –
- **dup_idx_start** –
- **dup_idx_end** –
- **row_idx_start** –
- **row_idx_end** –
- **col_idx_start** –
- **col_idx_end** –
- **rows_columns** –

Returns

classmethod freeze_rows_cols(*spreadsheet_id*, *tab_id*, *freeze_idx*, *credentials*, *rows=True*, *req_limit=1*)

Freezes the rows of a given Google Sheet.

Parameters

- **rows** –
- **freeze_idx** –
- **spreadsheet_id** – Id of Google Sheet to retrieve.
- **tab_id** – Id of tab to modify.
- **freeze_row** – Spreadsheet row to freeze.
- **credentials** – GSuite credentials object.

Returns Result object for API call.

classmethod get_google_sheet(*spreadsheet_id*, *range_name*, *credentials*, *tab_name=False*, *req_limit=1*)

Gets data from a Google Sheet.

Parameters

- **spreadsheet_id** – Id of Google Sheet to retrieve.
- **range_name** – Row/Column of Sheet range to retrieve (ex: A1:A99).
- **tab_name** – Name of tab to pull data from (optional).
- **credentials** – GSuite credentials object.

Returns Pandas DataFrame of retrieved data.

classmethod get_google_sheet_metadata(*spreadsheet_id*, *credentials*, *include_grid_data=False*, *ranges=False*, *req_limit=1*)

Parameters

- **include_grid_data** –
- **spreadsheet_id** –
- **credentials** –
- **ranges** –

Returns

classmethod get_google_sheet_tabs(*spreadsheet_id*, *credentials*, *req_limit=1*)

Get the names and IDs of tabs for a given Google Sheet.

Parameters

- **spreadsheet_id** – ID of spreadsheet to update.
- **credentials** – GSuite credentials object.

Returns Names and IDs of tabs for a given Google Sheet.

classmethod group_sheet_cols_rows(*spreadsheet_id*, *tab_id*, *start_idx*, *end_idx*, *credentials*, *rows_columns='ROWS'*, *req_limit=1*)

Parameters

- **spreadsheet_id** –
- **tab_id** –
- **start_idx** –

- **end_idx** –
- **credentials** –
- **rows_columns** –

Returns

classmethod make_google_sheet(*ss_name*, *credentials*, *req_limit=1*)
Creates a Google Sheet in one's GSuite account.

Parameters

- **ss_name** – Name of the Google Sheet to be created.
- **credentials** – GSuite credentials object.

Returns Id of newly created Google Sheet.

classmethod update_cell_background_color(*spreadsheet_id*, *sheet_id*, *row_idx_start*, *row_idx_end*, *col_idx_start*, *credentials*, *color_dict*, *cell_or_row='CELL'*, *col_idx_end=False*, *req_limit=1*)

Parameters

- **spreadsheet_id** –
- **sheet_id** –
- **row_idx_start** –
- **row_idx_end** –
- **col_idx_start** –
- **col_idx_end** –
- **credentials** –
- **dimension** –

Returns

classmethod update_google_sheet_df(*spreadsheet_id*, *df*, *data_range*, *credentials*, *header=False*, *req_limit=1*)
Uploads a Pandas DataFrame to a Google Sheet.

Parameters

- **spreadsheet_id** – ID of spreadsheet to update.
- **df** – Pandas DataFrame to update spreadsheet with.
- **data_range** – Spreadsheet range to insert DataFrame into.
- **credentials** – GSuite credentials object.

Returns API response.

classmethod update_google_sheet_val(*spreadsheet_id*, *tab_id*, *val*, *row*, *col*, *credentials*, *req_limit=1*)
Updates a google spreadsheet cell with a value.

Parameters

- **spreadsheet_id** – Id of spreadsheet to update.
- **tab_id** – Id of spreadsheet tab to update.

- **val** – Value to update spreadsheet cell with.
- **row** – Row of cell to update.
- **col** – Column of cell to update.
- **credentials** – GSuite credentials object.

Returns API response.

4.5.9 fusetools.home_tools

Home Automation Tools.

Classes

Nest

fusetools.home_tools.Nest

class fusetools.home_tools.**Nest**
Bases: object

Methods

execute_device_command
get_access_refresh_tokens
get_auth_code
get_device
get_devices
get_structures
refresh_access_token

4.5.10 fusetools.logging_tools

Logging tasks.

Functions

log_all	Log all stdout to a local logfile.
log_all_thread	Logs all specified data on a 'main' thread or a 'sub' thread.
log_setup	Creates a logger object with specified parameters.
log_tbl_df	Creates a Pandas DataFrame of logging details.
make_script_function	Execute a Python file as a function.

fusetools.logging_tools.log_all

fusetools.logging_tools.**log_all**(*filename=False*)

Log all stdout to a local logfile.

Parameters **filename** – Filename to write stdout content to.

Returns Logging contents to a local file.

fusetools.logging_tools.log_all_thread

fusetools.logging_tools.**log_all_thread**(*filename*, *level='INFO'*, *thread_type='main'*, *main_log=False*)

Logs all specified data on a 'main' thread or a 'sub' thread. Main thread excluded logging from other threads. Sub thread logs data for only that threadId.

Parameters

- **filename** – Filename to log the thread contents to.
- **level** – Level of logging for the thread (ie: INFO, DEBUG, WARNING, etc)
- **thread_type** – 'Main' thread or 'Sub' thread.
- **main_log** – Main log to specify if using a sub thread.

Returns A logfile with thread's output.

fusetools.logging_tools.log_setup

fusetools.logging_tools.**log_setup**(*name*, *filename*, *warning_type*)

Creates a logger object with specified parameters.

Parameters

- **name** – Name of logger object.
- **filename** – Filename of logging contents.
- **warning_type** – Level of warnings to provide.

Returns Logger object with specified parameters.

fusetools.logging_tools.log_tbl_df

fusetools.logging_tools.**log_tbl_df**(*proc_owner=None*, *proc_cat=None*, *proc_name=None*, *run_datetime=None*, *elapsed_run_time=None*, *max_completed_step=None*, *outcome=None*, *notes1_key=None*, *notes1_note=None*, *notes1_val=None*, *notes2_key=None*, *notes2_note=None*, *notes2_val=None*, *notes3_key=None*, *notes3_note=None*, *notes3_val=None*, *notes4_key=None*, *notes4_note=None*, *notes4_val=None*, *notes5_key=None*, *notes5_note=None*, *notes5_val=None*, *notes6_key=None*, *notes6_note=None*, *notes6_val=None*, *notes7_key=None*, *notes7_note=None*, *notes7_val=None*, *notes8_key=None*, *notes8_note=None*, *notes8_val=None*, *notes9_key=None*, *notes9_note=None*, *notes9_val=None*, *notes10_key=None*, *notes10_note=None*, *notes10_val=None*)

Creates a Pandas DataFrame of logging details.

Parameters

- **proc_owner** (Optional[list]) –
- **proc_cat** (Optional[list]) –
- **proc_name** (Optional[list]) –
- **rundate** –
- **proc_runtime** –
- **max_completed_step** (Optional[list]) –
- **outcome** (Optional[list]) –
- **notes1_key** (Optional[list]) –
- **notes1_note** (Optional[list]) –
- **notes1_val** (Optional[list]) –
- **notes2_key** (Optional[list]) –
- **notes2_note** (Optional[list]) –
- **notes2_val** (Optional[list]) –
- **notes3_key** (Optional[list]) –
- **notes3_note** (Optional[list]) –

- **notes3_val** (Optional[list]) –
- **notes4_key** (Optional[list]) –
- **notes4_note** (Optional[list]) –
- **notes4_val** (Optional[list]) –
- **notes5_key** (Optional[list]) –
- **notes5_note** (Optional[list]) –
- **notes5_val** (Optional[list]) –
- **notes6_key** (Optional[list]) –
- **notes6_note** (Optional[list]) –
- **notes6_val** (Optional[list]) –
- **notes7_key** (Optional[list]) –
- **notes7_note** (Optional[list]) –
- **notes7_val** (Optional[list]) –
- **notes8_key** (Optional[list]) –
- **notes8_note** (Optional[list]) –
- **notes8_val** (Optional[list]) –
- **notes9_key** (Optional[list]) –
- **notes9_note** (Optional[list]) –
- **notes9_val** (Optional[list]) –
- **notes10_key** (Optional[list]) –
- **notes10_note** (Optional[list]) –
- **notes10_val** (Optional[list]) –

Returns

fusetools.logging_tools.make_script_function

fusetools.logging_tools.**make_script_function**(*module*, *path*)
Execute a Python file as a function.

Parameters

- **module** – Arbitrary name for file to run.
- **path** – Filepath for file to run.

Returns Executed run of a Python file.

Classes

IgnoreThreadsFilter	Only accepts log records that originated from the main thread
ThreadFilter	Only accept log records from a specific thread or thread name

fusetools.logging_tools.IgnoreThreadsFilter

class fusetools.logging_tools.**IgnoreThreadsFilter**
Bases: logging.Filter

Only accepts log records that originated from the main thread

Initialize a filter.

Initialize with the name of the logger which, together with its children, will have its events allowed through the filter. If no name is specified, allow every event.

Methods

filter	Determine if the specified record is to be logged.

filter(*record*)
Determine if the specified record is to be logged.

Is the specified record to be logged? Returns 0 for no, nonzero for yes. If deemed appropriate, the record may be modified in-place.

fusetools.logging_tools.ThreadFilter

class fusetools.logging_tools.**ThreadFilter**(*threadid=None*, *threadname=None*)
Bases: logging.Filter

Only accept log records from a specific thread or thread name

Initialize a filter.

Initialize with the name of the logger which, together with its children, will have its events allowed through the filter. If no name is specified, allow every event.

Methods

filter	Determine if the specified record is to be logged.

filter(*record*)
Determine if the specified record is to be logged.

Is the specified record to be logged? Returns 0 for no, nonzero for yes. If deemed appropriate, the record may be modified in-place.

4.5.11 fusetools.marketing_tools

Marketing, Advertising, Campaign & Analytics Tools.

Classes

AppAnnie	Functions for interacting with App Annie.
GoogleAnalytics	Functions for interacting with Google Analytics.

fusetools.marketing_tools.AppAnnie

class fusetools.marketing_tools.**AppAnnie**
Bases: object

Functions for interacting with App Annie.

Methods

pull_metrics	Pull reporting metrics from App Annie.

classmethod pull_metrics(*api_keys*, *start_date*, *end_date*)
Pull reporting metrics from App Annie.

Parameters

- **api_keys** – App Annie API Keys.
- **start_date** – Date to pull reporting from.
- **end_date** – Date to pull reporting until.

Returns Pandas DataFrame of App Annie results.

fusetools.marketing_tools.GoogleAnalytics

class fusetools.marketing_tools.**GoogleAnalytics**
Bases: object

Functions for interacting with Google Analytics.

Methods

authorize_credentials	Initializes an Analytics Reporting API V4 service object.
ga_pull_metrics	Generates a Google Analytics report.

static authorize_credentials(*filename*)
Initializes an Analytics Reporting API V4 service object.

Parameters **filename** – Filepath to authentication JSON token.

Returns An authorized Analytics Reporting API V4 service object.

classmethod ga_pull_metrics(*creds*, *view_id*, *start_date*, *end_date*, *metrics*, *dimensions=False*, *filter_dimensions=False*, *filter_operators=False*, *filter_expressions=False*, *filter_excludes=False*)
Generates a Google Analytics report.

Parameters

- **creds** – Authentication object.
- **view_id** – Id for organization's Google Analytics account.
- **start_date** – Reporting start date.
- **end_date** – Reporting end date.
- **metrics** – List of KPIs to compute. Docs: https://ga-dev-tools.appspot.com/dimensions-metrics-explorer/.
- **dimensions** – List of Dimensions to aggregate results by. Docs: https://ga-dev-tools.appspot.com/dimensions-metrics-explorer/.
- **filter_dimensions** – List of Dimensions to filter results by.
- **filter_operators** – List of arithmetical operators to constraint dimensions by.
- **filter_expressions** – List of expressions to constraint dimensions by.
- **filter_excludes** – Whether or not use dimension filter to exclude results.

Returns JSON response with report results.

4.5.12 fusetools.ml_tools

Functions for interacting with Machine Learning Tools.

Classes

Prep	Functions for preparing data for machine learning tasks.
Train	Functions for training machine learning models.
Viz	Functions for visualizing results from machine learning tasks.

fusetools.ml_tools.Prep

class fusetools.ml_tools.**Prep**

Bases: object

Functions for preparing data for machine learning tasks.

Methods

label_encode_df	Assigns an incremental number for each string value and then converts the new number to a float so we can add the missing value (NaNs) back in for later imputation.
make_model_feature_df	Creates Pandas DataFrame with cumulatively trained combined feature names, coefficients, absolute coefficients in order of trained together.

classmethod label_encode_df(*df*, *col*, *col_new*)

Assigns an incremental number for each string value and then converts the new number to a float so we can add the missing value (NaNs) back in for later imputation.

Parameters

- **df** – Pandas DataFrame with atleast one feature to encode.
- **col** – Name of column to encode.
- **col_new** – Name of new, encoded column.

Returns Pandas DataFrame with new, encoded column.

classmethod make_model_feature_df (*df*, *cat*, *model_list*)
Creates Pandas DataFrame with cumulatively trained combined feature names, coefficients, absolute coefficients in order of trained together.

Parameters

- **df** – Pandas DataFrame of a fitted SckitLearn estimator's features, coefficients absolute coefficients.
- **cat** – Name of ScikitLearn estimator or self defines estimator type.
- **model_list** – List of ScikitLearn estimators.

Returns Pandas DataFrame with cumulatively trained combined feature names, coefficients, absolute coefficients and performance placeholders in order of trained together.

fusetools.ml_tools.Train

class fusetools.ml_tools.**Train**
Bases: object

Functions for training machine learning models.

Methods

train_model	Takes a ScikitLearn estimator instance and train and test dataframes returns: an estimator as well as the predictions and probabilities for the test set.

classmethod train_model (*estimator*, *X_train*, *X_test*, *y_train*, *predict_method*)
Takes a ScikitLearn estimator instance and train and test dataframes returns: an estimator as well as the predictions and probabilities for the test set.

Parameters

- **estimator** – ScikitLearn estimator instance
- **X_train** – Input training dataset
- **X_test** – Input test dataset
- **y_train** – Target training dataset
- **predict_method** – Type of prediction to make.

Returns Fitted SckitLearn estimator, test predictions and test scores for each prediction.

fusetools.ml_tools.Viz

class fusetools.ml_tools.**Viz**
Bases: object

Functions for visualizing results from machine learning tasks.

Methods

show_clf_perf	Prints a confusion matrix of an estimator's binary classification performance.
show_clf_perf_features	Creates a plot of given ScikitLearn estimator Accuracies (Y Axis) by number of features in model (X Axis).
show_model_results	Prints the Accuracy, AUC and Metrics Classification report for an estimator.

classmethod show_clf_perf(*width*, *height*, *y_test*, *y_pred*, *y_score*)
Prints a confusion matrix of an estimator's binary classification performance.

Parameters

- **width** – Plot width.
- **height** – Plot height.
- **y_test** – Output labels for the test dataset.
- **y_pred** – Output predictions labels for the test dataset.
- **y_score** – Probabilities of certainty for output prediction labels.

Returns Confusion matrix of an estimator's binary classification performance.

classmethod show_clf_perf_features(*width*, *height*, *df*, *title*, *model_list*, *df_max_stats*)
Creates a plot of given ScikitLearn estimator Accuracies (Y Axis) by number of features in model (X Axis).

Parameters

- **width** – Plot width.
- **height** – Plot height.
- **df** – Pandas DataFrame with cumulatively trained combined feature names, coefficients, absolute coefficients in order of trained together.
- **title** – Plot title.
- **model_list** – List of ScikitLearn estimators to iterate through and plot cumulative performance for.
- **df_max_stats** – Pandas DataFrame of estimator names, max accuracy achieved and number of features for fitted estimator. Used for plot annotation.

Returns Plot of given ScikitLearn estimator Accuracies (Y Axis) by number of features in model (X Axis).

classmethod show_model_results(*estimator*, *y_test*, *y_pred*, *y_score*)
Prints the Accuracy, AUC and Metrics Classification report for an estimator.

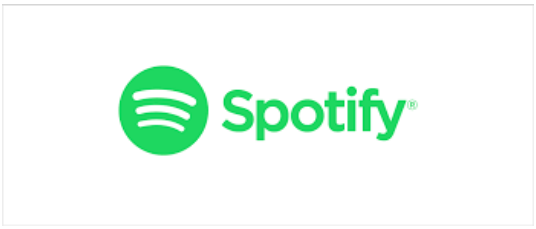
Parameters

- **estimator** – A trained estimator.
- **y_test** – Output labels for the test dataset.
- **y_pred** – Output predictions labels for the test dataset.
- **y_score** – Probabilities of certainty for output prediction labels.

Returns Accuracy, AUC and Metrics Classification report for an estimator.

4.5.13 fusetools.music_tools

Functions for interacting with Music Tools.



Classes

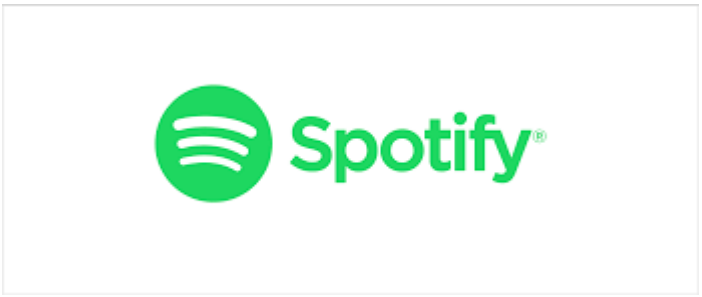
Spotify	Functions for interacting with Spotify.

fusetools.music_tools.Spotify

class fusetools.music_tools.**Spotify**

Bases: object

Functions for interacting with Spotify.



Methods

auth	Creates a Spotify API authentication object.
get_playlist_tracks	Retrieve tracks for a given Spotify playlist.
get_track_audio_features	Retrieve audio feature tracks for a given track.
get_user_playlists	Retrieve tracks for a given Spotify playlist.
search_track	

classmethod auth(*clientid*, *clientsecret*)

Creates a Spotify API authentication object.

Parameters

- **clientid** – Spotify developer client Id.
- **clientsecret** – Spotify developer client secret.

Returns Spotify API authentication object.

classmethod get_playlist_tracks(*sp*, *username*, *playlist_id*)
Retrieve tracks for a given Spotify playlist.

Parameters

- **sp** – Spotify API authentication object.
- **username** – Spotify username.
- **playlist_id** – Spotify playlist Id.

Returns List of tracks on a Spotify playlist.

classmethod get_track_audio_features(*sp*, *track_id*)
Retrieve audio feature tracks for a given track.

Parameters

- **sp** – Spotify API authentication object.
- **track_id** – Spotify track Id.

Returns Audio feature tracks for a given track.

classmethod get_user_playlists(*sp*, *username*)
Retrieve tracks for a given Spotify playlist.

Parameters

- **sp** – Spotify API authentication object.
- **username** – Spotify username.
- **playlist_id** – Spotify playlist Id.

Returns List of tracks on a Spotify playlist.

4.5.14 fusetools.pm_tools

Functions for interacting with Project Management Tools.

Classes

Asana	Functions for interacting with Asana.
Workfront	Functions for interacting with Workfront.

fusetools.pm_tools.Asana

class fusetools.pm_tools.**Asana**
Bases: object

Functions for interacting with Asana.

Methods

create_task	Creates an Asana task.
delete_task	
get_project_tasks	Retrieve tasks on an Asana project.
get_task_detail	Retrieve Asana task details.
pull_tasks_for_project	Retrieve tasks on an Asana project.

classmethod create_task(*asana_token*, *project*, *taskName*, *taskDue*, *assignee*, *taskNotes=False*)
Creates an Asana task.

Parameters

- **asana_token** – Asana API token.
- **project** – Asana project id.
- **taskName** – Name of task.
- **taskDue** – Due date for task.
- **assignee** – Assigned person for task.
- **taskNotes** – Notes on task.

Returns API call response.

classmethod get_project_tasks(*asana_token*, *project*)
Retrieve tasks on an Asana project.

Parameters

- **asana_token** – Asana API token.
- **project** – Asana project Id.

Returns Project tasks.

classmethod get_task_detail(*asana_token*, *task_id*)
Retrieve Asana task details.

Parameters

- **asana_token** – Asana API token.

- **task_id** – Asana task Id.

Returns Project task details.

classmethod pull_tasks_for_project(*asana_token*, *project*)
Retrieve tasks on an Asana project.

Parameters

- **asana_token** – Asana API token.
- **project** – Asana project Id.

Returns Project tasks.

fusetools.pm_tools.Workfront

class fusetools.pm_tools.**Workfront**
Bases: object

Functions for interacting with Workfront.

Methods

wf_login	Authenticates a user for a given Workfront login/password.
wf_pull_issues_custom_field	Retrieves Workfront issues with a given custom field.
wf_pull_projects_custom_field	Retrieves Workfront projects with a given custom field.
wf_pull_projects_status	Retrieves Workfront projects in a given project status.
wf_pull_tasks_name	Retrieves Workfront tasks with a name string.
wf_pull_tasks_status	Retrieves Workfront tasks with a given status.
wf_upload_file	Uploads a file to Workfront.

classmethod wf_login(*wf_name*, *wf_pwd*, *proxies=None*)
Authenticates a user for a given Workfront login/password.

Parameters

- **wf_name** – Workfront username.
- **wf_pwd** – Workfront password.
- **proxies** – Proxies to include on request (Optional).

Returns SessionId for authenticated user.

classmethod wf_pull_issues_custom_field(*wf_api_key*, *proxies=None*)

Retrieves Workfront issues with a given custom field.

Parameters

- **proxies** – Proxies to include on request (Optional).
- **wf_api_key** – Workfront API key.

Returns JSON response for API call.

classmethod wf_pull_projects_custom_field(*wf_api_key*, *proxies=None*)
Retrieves Workfront projects with a given custom field.

Parameters

- **wf_api_key** – Workfront API key.
- **proxies** – Proxies to include on request (Optional).

Returns JSON response for API call.

classmethod wf_pull_projects_status(*status*, *wf_api_key*, *proxies=None*)
Retrieves Workfront projects in a given project status.

Parameters

- **status** – Project status to search for projects.
- **wf_api_key** – Workfront API key.
- **proxies** – Proxies to include on request (Optional).

Returns JSON response for API call.

classmethod wf_pull_tasks_name(*name_string*, *wf_api_key*, *proxies=None*, *assignee=False*)
Retrieves Workfront tasks with a name string.

Parameters

- **name_string** – String to search tasks for.
- **wf_api_key** – Workfront API Key.
- **proxies** – Proxies to include on request (Optional).
- **assignee** – Additional filter on task assignee name.

Returns JSON response for API call.

classmethod wf_pull_tasks_status(*status*, *wf_api_key*, *proxies=None*, *assignee=False*)
Retrieves Workfront tasks with a given status.

Parameters

- **status** – Workfront status category.
- **proxies** – Proxies to include on request (Optional).
- **wf_api_key** – Workfront API Key.
- **assignee** – Additional filter on task assignee name.

Returns JSON response for API call.

classmethod wf_upload_file(*wf_api_key*, *file*, *file_path*, *obj_id*, *obj_type*, *proxies=None*)
Uploads a file to Workfront.

Parameters

- **wf_api_key** – Workfront API Key

- **file** – Name of file to upload.
- **file_path** – Name of filepath for file to upload.
- **obj_id** – Id of Workfront object to upload file to.
- **obj_type** – Type of object to upload on Workfront.
- **proxies** – Proxies to include on request (Optional).

Returns JSON response for API call.

4.5.15 fusetools.research_tools

Functions for conduction various Research tasks.

Classes

Economics	Functions for dealing with Economic data sources.
Geography	Functions for dealing with Geographical tasks.

fusetools.research_tools.Economics

class fusetools.research_tools.**Economics**

Bases: object

Functions for dealing with Economic data sources.

Methods

bea_gdp	Performs a query against the BEA API.
bls_query	Retrieves Consumer Price Index figures from the Bureau of Labor Statistics API.
bls_query_lookup	Executes a series of queries against the BLS database using a DataFrame of lookups.

classmethod bea_gdp(*api_key*, *tbl_name='SQGDP1'*)

Performs a query against the BEA API.

Parameters

- **api_key** – BEA API.
- **tbl_name** – Name of data table to query from BEA database.

Returns Pandas DataFrame of responses.

classmethod bls_query(*series_id*, *start_year*, *end_year*, *api_key*)

Retrieves Consumer Price Index figures from the Bureau of Labor Statistics API.

Parameters

- **series_id** – Unique ID for a geography and an economic measure.
- **start_year** – Starting year for query.

- **end_year** – Ending year for query.
- **api_key** – API key for BLS.

Returns JSON response for API call.

classmethod bls_query_lookup(*lookup_df*, *lookup_area_type*, *start_year*, *end_year*, *api_keys*)

Executes a series of queries against the BLS database using a DataFrame of lookups.

Parameters

- **lookup_df** – Pandas DataFrame of BLS lookup codes.
- **lookup_area_type** – Type of area to lookup data for.
- **start_year** – Starting year for query.
- **end_year** – Ending year for query.
- **api_keys** – API key for BLS.

Returns Pandas DataFrame of responses for all queried lookup codes.

fusetools.research_tools.Geography

class fusetools.research_tools.**Geography**

Bases: object

Functions for dealing with Geographical tasks.

Methods

calculate_distance	Calculates the distance between two latitude/longitude coordinate pairs.
get_city_lat_lon	Calculates the latitude and longitude for a city.
walk_bike_transit_score	**param addr**

classmethod calculate_distance(*lat_from*, *lon_from*, *lat_to*, *lon_to*)

Calculates the distance between two latitude/longitude coordinate pairs.

Parameters

- **lat_from** – Latitude of the point being compared from.
- **lon_from** – Longitude of the point being compared from.
- **lat_to** – Latitude of the point being compared to.
- **lon_to** – Longitude of the point being compared to.

Returns Calculated distance.

classmethod get_city_lat_lon(*city*)

Calculates the latitude and longitude for a city.

Parameters **city** – Name of city.

Returns Location (latitude and longitude).

classmethod walk_bike_transit_score(*addr*, *lat*, *lon*, *api_key*)

Parameters

- **addr** –
- **lat** –
- **lon** –
- **api_key** –

Returns

4.5.16 fusetools.stat_tools

Functions for interacting with Machine Learning Tools.

Classes

Desc	Functions for helping with Descriptive statistical tasks.
Test	Functions for implementing Statistical tests.
Viz	Functions for visualizing distributions.

fusetools.stat_tools.Desc

class fusetools.stat_tools.**Desc**

Bases: object

Functions for helping with Descriptive statistical tasks.

Methods

group_stats	Creates a Pandas DataFrame with specified aggregations over specified dimensions.
make_tbl	Creates a plot of a data table.

classmethod group_stats(*df*, *dim_cols*, *agg_dict*)

Creates a Pandas DataFrame with specified aggregations over specified dimensions.

Parameters

- **df** – Pandas DataFrame.
- **dim_cols** – List of columns to group by.

- **agg_dict** – Dictionary of columns and calculations to perform.

Returns Pandas DataFrame of calculated results.

classmethod make_tbl(*width*, *height*, *df*, *title*, *font_size*)
Creates a plot of a data table.

Parameters

- **width** – Width of table.
- **height** – Height of table.
- **df** – Pandas DataFrame.
- **title** – Title of plot.
- **font_size** – Font size.

Returns Plot of a data table.

fusetools.stat_tools.Test

class fusetools.stat_tools.**Test**
Bases: object

Functions for implementing Statistical tests.

Methods

chi_squared_result	Calculates correlation between 2 proportions using a Chi-Squared test..
correlation	Performs a Pearson test of correlation between two data samples.
cramers_corrected_stat	Calculates correlation between 2 categorical variables using Cramer's method.
pe	Calculates the Price Elasticity of Demand.
poisson	Performs a Poisson test which tests statistical difference between groups comparing counts over a period of time.
sample_size1	Calculates sample size needed for desired measuring effect size.
survival_result	Performs a survival test which tells if statistical difference in times until an outcome between two samples.
ttest	Performs a t-test between two groups split by a flag.
ttest_result	Performs a T-Test between two groups of data.

classmethod chi_squared_result(*sample1_successes*, *sample1_trials*, *sample2_successes*, *sample2_trials*)
Calculates correlation between 2 proportions using a Chi-Squared test..

Parameters

- **sample1_successes** – Sample 1's successes.
- **sample1_trials** – Sample 1's trials.

- **sample2_successes** – Sample 2's successes.
- **sample2_trials** – Sample 2's successes.

Returns Chi-Squared p-value.

classmethod correlation(*sample1_dat*, *sample2_dat*)
Performs a Pearson test of correlation between two data samples.

Parameters

- **sample1_dat** – Sample 1 data array/list.
- **sample2_dat** – Sample 2 data array/list.

Returns Pearson correlation result.

classmethod cramers_corrected_stat(*cat_col1*, *cat_col2*)
Calculates correlation between 2 categorical variables using Cramer's method.

Parameters

- **cat_col1** – Categorical column 1.
- **cat_col2** – Categorical column 2.

Returns Correlation between 2 categorical variables using Cramer's method.

classmethod pe(*type*, *original_quantity=False*, *new_quantity=False*, *original_price=False*, *new_price=False*, *pe_prices=False*, *pe_quantities=False*)
Calculates the Price Elasticity of Demand.

Parameters

- **type** – Classification of whether data is in array/list data format or a scalar format (sample or other).
- **original_quantity** – Starting quantity demanded if data is scalar values.
- **new_quantity** – Ending quantity demanded if data is scalar values.
- **original_price** – Starting price if data is scalar values.
- **new_price** – Ending price if data is scalar values.
- **pe_prices** – Array/list of prices paid for quantities demanded.
- **pe_quantities** – Array/lust of quantities demanded.

Returns Price elasticity of demand (float).

classmethod poisson(*sample1_events*, *sample1_days*, *sample2_events*, *sample2_days*)
Performs a Poisson test which tests statistical difference between groups comparing counts over a period of time.

Parameters

- **sample1_events** – Count of sample 1 events.
- **sample1_days** – Count of sample 1 days.
- **sample2_events** – Count of sample 2 events.
- **sample2_days** – Count of sample 2 days.

Returns P-value for a Poisson statistical test.

classmethod sample_size1(*baseline_input*, *effect_size_input*, *significance_level_input*, *statistical_power_input*)
Calculates sample size needed for desired measuring effect size.

Parameters

- **baseline_input** – Baseline rate to measure effect against against.
- **effect_size_input** – Desired effect size to measure.
- **significance_level_input** – Desired level of statistical significance.
- **statistical_power_input** – Desired level of statistical power.

Returns Calculated sample size.

classmethod survival_result(*data_type*, *sample1_dat_survival=False*, *sample2_dat_survival=False*, *survival_confidence_level=False*, *sample1_dat_survival_mean=False*, *sample1_dat_survival_size=False*, *sample2_dat_survival_mean=False*, *sample2_dat_survival_size=False*)
Performs a survival test which tells if statistical difference in times until an outcome between two samples.

Parameters

- **data_type** – Classification of whether data is in array/list data format or a scalar format (sample or other).
- **sample1_dat_survival** – Sample 1 data if array/list.
- **sample2_dat_survival** – Sample 1 data if array/list.
- **survival_confidence_level** – Confidence interval to assess measure test.
- **sample1_dat_survival_mean** – Sample 1 mean if scalar value.
- **sample1_dat_survival_size** – Sample 1 size if scalar value.
- **sample2_dat_survival_mean** – Sample 2 mean if scalar value.
- **sample2_dat_survival_size** – Sample 2 size if scalar value.

Returns P-value for statistical significance in difference in times until an outcomes between two samples.

classmethod ttest(*df*, *grp_col*, *grp_1_flag*, *grp_2_flag*, *target_kpi*)
Performs a t-test between two groups split by a flag.

Parameters

- **df** – Pandas DataFrame containing data.
- **grp_col** – Column used to group the data.
- **grp_1_flag** – Value used to distinguish group 1.
- **grp_2_flag** – Value used to distinguish group 2.
- **target_kpi** – Column for the target metric to compare test across groups.

Returns T-Test p-value.

classmethod ttest_result(*sample1_dat_ttest*, *sample2_dat_ttest*)
Performs a T-Test between two groups of data.

Parameters

- **sample1_dat_ttest** – Sample 1 dataset.
- **sample2_dat_ttest** – Sample 2 dataset.

Returns T-Test p-value.

fusetools.stat_tools.Viz

class fusetools.stat_tools.**Viz**
Bases: object

Functions for visualizing distributions.

Methods

dist_plot	Creates a histogram of data.
make_plot_tbl	Creates a visualization of a data table next to a plot of the data.
make_plotting_tbl	Create a visualization of a data table + a bar graph.

classmethod dist_plot(*df*, *col*, *sav_dir=''*)
Creates a histogram of data.

Parameters

- **df** – Pandas DataFrame of data to plot.
- **col** – Column to plot on y-axis (bars).
- **sav_dir** – Directory to save plot in.

Returns Saved plot.

classmethod make_plot_tbl(*width*, *height*, *plot_size*, *tbl_size*, *df*, *col*, *tgt_col*, *title*, *xlabel*, *ylabel*, *agg_df*, *plot_type*, *yaxis_fmt*, *xaxis_fmt*, *stat*, *font_size*)
Creates a visualization of a data table next to a plot of the data. Intended for use in Jupyter Notebook.

Parameters

- **width** – Width of plot.
- **height** – Height of plot.
- **plot_size** – Size of overall plot.
- **tbl_size** – Size of data table.
- **df** – Pandas DataFrame of Data to plot.
- **col** – Dimension column for plot.
- **tgt_col** – KPI column for plot.
- **title** – Title for plot.
- **xlabel** – Xlabel for plot.
- **ylabel** – YLabel for plot.
- **agg_df** – Pandas DataFrame for data table.
- **plot_type** – Type of visualization to plot (box, box_h, scatter, dist, agg_dist)

- **yaxis_fmt** – Format for yaxis.
- **xaxis_fmt** – Format for xaxis.
- **stat** – Type of statistic to add to the plot if box plot (currently only T-Test P-value supported).
- **font_size** – Size of font for table.

Returns Visualization of a data table next to a plot of the data.

classmethod make_plotting_tbl(*width*, *height*, *plot_size*, *tbl_size*, *df_plot*, *plot_col_x*, *plot_col_y*, *plot_col_hue*, *plot_title*, *df_tbl*, *font_size*)

Create a visualization of a data table + a bar graph.

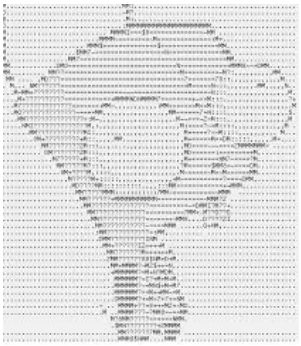
Parameters

- **width** – Width of plot.
- **height** – Height of plot.
- **plot_size** – Size of overall plot.
- **tbl_size** – Size of data table.
- **df_plot** – Pandas DataFrame of data to plot.
- **plot_col_x** – Column name to plot on X axis.
- **plot_col_y** – Column name to plot on Y axis (bars).
- **plot_col_hue** – Color for column on Y axis (bars).
- **plot_title** – Title for plot.
- **df_tbl** – Pandas DataFrame of data to show in datatable.
- **font_size** – Font size for data table.

Returns Visualization of a data table next to a bar plot of the data.

4.5.17 fusetools.text_tools

Tools for performing text tasks.



Classes

Blob	Functions for dealing with blobs of text.
Export	Functions for exporting Python text objects.

fusetools.text_tools.Blob

class fusetools.text_tools.**Blob**
Bases: object

Functions for dealing with blobs of text.

Methods

text_parse	Takes a blob of values input into 'TextArea' box widget and returns a numpy array Text blobs can be comma, space or line delimited

classmethod text_parse(*blob1*, *blob2*)
Takes a blob of values input into 'TextArea' box widget and returns a numpy array Text blobs can be comma, space or line delimited

Parameters

- **blob1** –
- **blob2** –

Returns

fusetools.text_tools.Export

class fusetools.text_tools.**Export**
Bases: object

Functions for exporting Python text objects.

Methods

concat_text_files	Concatenates contents of specified files into a target file and saves it.
dump_json	Exports a Python dictionary object to a JSON file.
dump_sql	Exports a Python string object to a SQL file.
find_replace_text	Finds and replaces a string in all text files in a target folder.

classmethod concat_text_files(*input_files*, *output_file*)
Concatenates contents of specified files into a target file and saves it.

Parameters

- **input_files** – List of text files to concatenate.

- **output_file** – Target output file.

Returns Concatenated target output file.

classmethod dump_json(*obj*, *filepath*)
Exports a Python dictionary object to a JSON file.

Parameters

- **obj** – Python dictionary object.
- **filepath** – Filepath to save object to.

Returns Exported JSON file.

classmethod dump_sql(*obj*, *filepath*)
Exports a Python string object to a SQL file.

Parameters

- **obj** – Python string object.
- **filepath** – Filepath to save object to.

Returns Exported SQL file.

classmethod find_replace_text(*directory*, *find*, *replace*, *file_pattern*, *specific_file=False*)
Finds and replaces a string in all text files in a target folder.

Parameters

- **directory** – Target folder containing files.
- **find** – String to search for.
- **replace** – New string to replace with.
- **file_pattern** – Pattern of text files to scan.

Returns Saved files with replaced text.

4.5.18 fusetools.web_tools

Functions

bitly_url_shortener
get_chrome_history

fusetools.web_tools.bitly_url_shortener

fusetools.web_tools.**bitly_url_shortener**(*long_url*, *api_token*, *domain='bit.ly'*)

fusetools.web_tools.get_chrome_history

fusetools.web_tools.**get_chrome_history**(*history_path=''*, *sql='select url from urls'*)

PYTHON MODULE INDEX

f

fusetools, 11
fusetools.analytics_tools, 11
fusetools.comm_tools, 15
fusetools.commerce_tools, 19
fusetools.dashboard_tools, 20
fusetools.date_tools, 22
fusetools.db_conn_tools, 24
fusetools.db_etl_tools, 29
fusetools.gsuite_tools, 42
fusetools.home_tools, 52
fusetools.logging_tools, 53
fusetools.marketing_tools, 57
fusetools.ml_tools, 59
fusetools.music_tools, 62
fusetools.pm_tools, 63
fusetools.research_tools, 67
fusetools.stat_tools, 69
fusetools.text_tools, 74
fusetools.web_tools, 76

INDEX

A

add_google_sheet_comment() (*fusetools.gsuite_tools.GSheets class method*), 48
add_google_sheet_tab() (*fusetools.gsuite_tools.GSheets class method*), 48
api_request() (*fusetools.commerce_tools.Discogs class method*), 20
AppAnnie (*class in fusetools.marketing_tools*), 57
append_window_agg() (*fusetools.analytics_tools.Pandas class method*), 12
Asana (*class in fusetools.pm_tools*), 64
auth() (*fusetools.music_tools.Spotify class method*), 62
auth_tableau_server() (*fusetools.dashboard_tools.Tableau class method*), 21
authorize_credentials() (*fusetools.gsuite_tools.GDrive class method*), 43
authorize_credentials() (*fusetools.marketing_tools.GoogleAnalytics static method*), 58

B

bea_gdp() (*fusetools.research_tools.Economics class method*), 67
bitly_url_shortener() (*in module fusetools.web_tools*), 76
Blob (*class in fusetools.text_tools*), 75
bls_query() (*fusetools.research_tools.Economics class method*), 67
bls_query_lookup() (*fusetools.research_tools.Economics class method*), 68
build_html() (*in module fusetools.comm_tools*), 15
bulk_add_google_sheet_comment() (*fusetools.gsuite_tools.GSheets class method*), 48
bulk_update_cell_background_color() (*fusetools.gsuite_tools.GSheets class method*), 48

C

calculate_distance() (*fusetools.research_tools.Geography class method*), 68
chi_squared_result() (*fusetools.stat_tools.Test class method*), 70
con_oracle() (*fusetools.db_conn_tools.Oracle class method*), 25
con_postgres() (*fusetools.db_conn_tools.Postgres class method*), 26
concat_text_files() (*fusetools.text_tools.Export class method*), 75
conn_pyodbc() (*fusetools.db_conn_tools.TeraData class method*), 28
conn_rs_pg() (*fusetools.db_conn_tools.Redshift class method*), 27
conn_rs_sa() (*fusetools.db_conn_tools.Redshift class method*), 27
conn_sa() (*fusetools.db_conn_tools.TeraData class method*), 28
conn_td() (*fusetools.db_conn_tools.TeraData class method*), 29
correlation() (*fusetools.stat_tools.Test class method*), 71
cramers_corrected_stat() (*fusetools.stat_tools.Test class method*), 71
create_service() (*fusetools.gsuite_tools.GMail class method*), 45
create_service_serv_acct() (*fusetools.gsuite_tools.GDrive class method*), 43
create_service_serv_acct() (*fusetools.gsuite_tools.GMail class method*), 45
create_service_serv_acct() (*fusetools.gsuite_tools.GSheets class method*), 48
create_task() (*fusetools.pm_tools.Asana class method*), 64
create_upload_folder() (*fuse-*

tools.gsuite_tools.GDrive class method), 43

D

db_apply_schema() (*fusetools.db_etl_tools.Generic class method*), 29
delete_google_sheet_data() (*fusetools.gsuite_tools.GSheets class method*), 49
delete_google_sheet_tab() (*fusetools.gsuite_tools.GSheets class method*), 49
Desc (*class in fusetools.stat_tools*), 69
Discogs (*class in fusetools.commerce_tools*), 19
dist_plot() (*fusetools.stat_tools.Viz class method*), 73
download_email_attachment() (*fusetools.gsuite_tools.GMail class method*), 45
download_file() (*fusetools.gsuite_tools.GDrive class method*), 44
drop_duplicates() (*fusetools.gsuite_tools.GSheets class method*), 49
dump_json() (*fusetools.text_tools.Export class method*), 76
dump_sql() (*fusetools.text_tools.Export class method*), 76

E

Economics (*class in fusetools.research_tools*), 67
eng_mysql() (*fusetools.db_conn_tools.MySQL class method*), 25
eng_oracle() (*fusetools.db_conn_tools.Oracle class method*), 25
eng_oracle_addr() (*fusetools.db_conn_tools.Oracle class method*), 26
eng_postgres() (*fusetools.db_conn_tools.Postgres class method*), 26
Export (*class in fusetools.text_tools*), 75

F

filter() (*fusetools.logging_tools.IgnoreThreadsFilter method*), 56
filter() (*fusetools.logging_tools.ThreadFilter method*), 56
find_na_holder() (*fusetools.analytics_tools.Pandas class method*), 13
find_replace_text() (*fusetools.text_tools.Export class method*), 76
freeze_rows_cols() (*fusetools.gsuite_tools.GSheets class method*), 49
fusetools
    module, 11
fusetools.analytics_tools
    module, 11
fusetools.comm_tools
    module, 15
fusetools.commerce_tools
    module, 19
fusetools.dashboard_tools
    module, 20
fusetools.date_tools
    module, 22
fusetools.db_conn_tools
    module, 24
fusetools.db_etl_tools
    module, 29
fusetools.gsuite_tools
    module, 42
fusetools.home_tools
    module, 52
fusetools.logging_tools
    module, 53
fusetools.marketing_tools
    module, 57
fusetools.ml_tools
    module, 59
fusetools.music_tools
    module, 62
fusetools.pm_tools
    module, 63
fusetools.research_tools
    module, 67
fusetools.stat_tools
    module, 69
fusetools.text_tools
    module, 74
fusetools.web_tools
    module, 76

G

ga_pull_metrics() (*fusetools.marketing_tools.GoogleAnalytics class method*), 58
GDrive (*class in fusetools.gsuite_tools*), 43
Generic (*class in fusetools.db_etl_tools*), 29
Geography (*class in fusetools.research_tools*), 68
get_all_google_items() (*fusetools.gsuite_tools.GDrive class method*), 44
get_all_sever_items() (*fusetools.dashboard_tools.Tableau class method*), 21
get_authentication_token() (*fusetools.commerce_tools.Discogs class method*),

20
get_chrome_history() (*in module fusetools.web_tools*), 77
get_city_lat_lon() (*fusetools.research_tools.Geography class method*), 68
get_emails() (*fusetools.gsuite_tools.GMail class method*), 46
get_file_revisions() (*fusetools.gsuite_tools.GDrive class method*), 44
get_google_sheet() (*fusetools.gsuite_tools.GSheets class method*), 50
get_google_sheet_metadata() (*fusetools.gsuite_tools.GSheets class method*), 50
get_google_sheet_tabs() (*fusetools.gsuite_tools.GSheets class method*), 50
get_last_dow() (*in module fusetools.date_tools*), 23
get_mailbox_labels() (*fusetools.gsuite_tools.GMail class method*), 46
get_messages() (*fusetools.comm_tools.Twilio class method*), 17
get_oracle_date() (*fusetools.db_etl_tools.Oracle class method*), 31
get_orcl_date() (*fusetools.db_etl_tools.Oracle class method*), 31
get_playlist_tracks() (*fusetools.music_tools.Spotify class method*), 63
get_project_tasks() (*fusetools.pm_tools.Asana class method*), 64
get_rptg_mon() (*in module fusetools.date_tools*), 23
get_rptg_qtr() (*in module fusetools.date_tools*), 23
get_rptg_week() (*in module fusetools.date_tools*), 23
get_rptg_yr() (*in module fusetools.date_tools*), 24
get_tableau_server_obj_id() (*fusetools.dashboard_tools.Tableau class method*), 22
get_task_detail() (*fusetools.pm_tools.Asana class method*), 64
get_track_audio_features() (*fusetools.music_tools.Spotify class method*), 63
get_user_playlists() (*fusetools.music_tools.Spotify class method*), 63
GMail (*class in fusetools.gsuite_tools*), 45
GoogleAnalytics (*class in fusetools.marketing_tools*), 58
group_sheet_cols_rows() (*fusetools.gsuite_tools.GSheets class method*), 50
group_stats() (*fusetools.stat_tools.Desc class method*), 69
GSheets (*class in fusetools.gsuite_tools*), 47

I

IgnoreThreadsFilter (*class in fusetools.logging_tools*), 56
insert_df_pg() (*fusetools.db_etl_tools.Postgres class method*), 34
insert_df_rs() (*fusetools.db_etl_tools.Redshift class method*), 37
insert_exec() (*fusetools.db_etl_tools.Oracle class method*), 31
insert_tbl() (*fusetools.db_etl_tools.Oracle class method*), 32
insert_td() (*fusetools.db_etl_tools.Teradata class method*), 41
insert_val_pg() (*fusetools.db_etl_tools.Postgres class method*), 34
insert_val_rs() (*fusetools.db_etl_tools.Redshift class method*), 38

L

label_encode_df() (*fusetools.ml_tools.Prep class method*), 59
log_all() (*in module fusetools.logging_tools*), 53
log_all_thread() (*in module fusetools.logging_tools*), 53
log_setup() (*in module fusetools.logging_tools*), 54
log_tbl_df() (*in module fusetools.logging_tools*), 54

M

make_db_cols() (*fusetools.db_etl_tools.Generic class method*), 30
make_db_schema() (*fusetools.db_etl_tools.Generic class method*), 30
make_df_tbl_pg() (*fusetools.db_etl_tools.Postgres class method*), 34
make_df_tbl_rs() (*fusetools.db_etl_tools.Redshift class method*), 38
make_google_sheet() (*fusetools.gsuite_tools.GSheets class method*), 51
make_groupby() (*fusetools.db_etl_tools.Generic class method*), 30
make_model_feature_df() (*fusetools.ml_tools.Prep class method*), 59
make_plot_tbl() (*fusetools.stat_tools.Viz class method*), 73
make_plotting_tbl() (*fusetools.stat_tools.Viz class method*), 74

make_script_function() (*in module fusetools.logging_tools*), 55
make_tableau_datasource_schema() (*fusetools.dashboard_tools.Tableau class method*), 22
make_tableau_hyperfile() (*fusetools.dashboard_tools.Tableau class method*), 22
make_tbl() (*fusetools.db_etl_tools.Oracle class method*), 32
make_tbl() (*fusetools.stat_tools.Desc class method*), 70
make_tbl_complete() (*fusetools.db_etl_tools.Oracle class method*), 32
make_tbl_complete_force() (*fusetools.db_etl_tools.Oracle class method*), 32
make_tbl_complete_pg() (*fusetools.db_etl_tools.Postgres class method*), 34
make_tbl_complete_rs() (*fusetools.db_etl_tools.Redshift class method*), 38
make_tbl_complete_td() (*fusetools.db_etl_tools.Teradata class method*), 41
make_tbl_td() (*fusetools.db_etl_tools.Teradata class method*), 41
module
    fusetools, 11
    fusetools.analytics_tools, 11
    fusetools.comm_tools, 15
    fusetools.commerce_tools, 19
    fusetools.dashboard_tools, 20
    fusetools.date_tools, 22
    fusetools.db_conn_tools, 24
    fusetools.db_etl_tools, 29
    fusetools.gsuite_tools, 42
    fusetools.home_tools, 52
    fusetools.logging_tools, 53
    fusetools.marketing_tools, 57
    fusetools.ml_tools, 59
    fusetools.music_tools, 62
    fusetools.pm_tools, 63
    fusetools.research_tools, 67
    fusetools.stat_tools, 69
    fusetools.text_tools, 74
    fusetools.web_tools, 76
MySQL (*class in fusetools.db_conn_tools*), 24

N

Nest (*class in fusetools.home_tools*), 52

O

Oracle (*class in fusetools.db_conn_tools*), 25
Oracle (*class in fusetools.db_etl_tools*), 30
orcl_tbl_varchar_convert() (*fusetools.db_etl_tools.Oracle class method*), 32

P

Pandas (*class in fusetools.analytics_tools*), 12
pe() (*fusetools.stat_tools.Test class method*), 71
period_comp() (*fusetools.analytics_tools.Pandas class method*), 13
period_start_dt() (*fusetools.analytics_tools.Pandas class method*), 13
poisson() (*fusetools.stat_tools.Test class method*), 71
Postgres (*class in fusetools.db_conn_tools*), 26
Postgres (*class in fusetools.db_etl_tools*), 33
Prep (*class in fusetools.ml_tools*), 59
ptd_measure() (*fusetools.analytics_tools.Pandas class method*), 13
pull_metrics() (*fusetools.marketing_tools.AppAnnie class method*), 57
pull_tasks_for_project() (*fusetools.pm_tools.Asana class method*), 65
push_tableau_hyperfile() (*fusetools.dashboard_tools.Tableau class method*), 22

R

Redshift (*class in fusetools.db_conn_tools*), 27
Redshift (*class in fusetools.db_etl_tools*), 37
run_query() (*fusetools.db_etl_tools.Generic class method*), 30
run_query_pg() (*fusetools.db_etl_tools.Postgres class method*), 34
run_query_rs() (*fusetools.db_etl_tools.Redshift class method*), 38
run_query_td() (*fusetools.db_etl_tools.Teradata class method*), 42

S

sample_size1() (*fusetools.stat_tools.Test class method*), 71
send_email() (*fusetools.comm_tools.SendGrid class method*), 16
send_email() (*fusetools.gsuite_tools.GMail class method*), 46
send_message() (*fusetools.comm_tools.Twilio class method*), 17
send_message() (*fusetools.comm_tools.WhatsApp class method*), 19
SendGrid (*class in fusetools.comm_tools*), 16

sequential_load_pg() (fusetools.db_etl_tools.Postgres class method), 35
sequential_load_pg_wk() (fusetools.db_etl_tools.Postgres class method), 35
sequential_load_rs() (fusetools.db_etl_tools.Redshift class method), 38
sequential_load_rs_wk() (fusetools.db_etl_tools.Redshift class method), 39
show_clf_perf() (fusetools.ml_tools.Viz class method), 61
show_clf_perf_features() (fusetools.ml_tools.Viz class method), 61
show_model_results() (fusetools.ml_tools.Viz class method), 61
Spotify (class in fusetools.music_tools), 62
SQL (class in fusetools.analytics_tools), 14
sql_rs_wow_comp() (fusetools.analytics_tools.SQL class method), 14
sql_rs_yoy_cum_comp() (fusetools.analytics_tools.SQL class method), 14
survival_result() (fusetools.stat_tools.Test class method), 72

T

Tableau (class in fusetools.dashboard_tools), 21
TeraData (class in fusetools.db_conn_tools), 28
Teradata (class in fusetools.db_etl_tools), 41
Test (class in fusetools.stat_tools), 70
text_parse() (fusetools.text_tools.Blob class method), 75
ThreadFilter (class in fusetools.logging_tools), 56
Train (class in fusetools.ml_tools), 60
train_model() (fusetools.ml_tools.Train class method), 60
ttest() (fusetools.stat_tools.Test class method), 72
ttest_result() (fusetools.stat_tools.Test class method), 72
Twilio (class in fusetools.comm_tools), 17

U

update_cell_background_color() (fusetools.gsuite_tools.GSheets class method), 51
update_google_sheet_df() (fusetools.gsuite_tools.GSheets class method), 51
update_google_sheet_val() (fusetools.gsuite_tools.GSheets class method), 51
upload_files() (fusetools.gsuite_tools.GDrive class method), 44
upsert_tbl_pg() (fusetools.db_etl_tools.Postgres class method), 36
upsert_tbl_rs() (fusetools.db_etl_tools.Redshift class method), 40
upsert_val_pg() (fusetools.db_etl_tools.Postgres class method), 36

V

Viz (class in fusetools.ml_tools), 60
Viz (class in fusetools.stat_tools), 73

W

walk_bike_transit_score() (fusetools.research_tools.Geography class method), 68
wf_login() (fusetools.pm_tools.Workfront class method), 65
wf_pull_issues_custom_field() (fusetools.pm_tools.Workfront class method), 65
wf_pull_projects_custom_field() (fusetools.pm_tools.Workfront class method), 66
wf_pull_projects_status() (fusetools.pm_tools.Workfront class method), 66
wf_pull_tasks_name() (fusetools.pm_tools.Workfront class method), 66
wf_pull_tasks_status() (fusetools.pm_tools.Workfront class method), 66
wf_upload_file() (fusetools.pm_tools.Workfront class method), 66
WhatsApp (class in fusetools.comm_tools), 18
Workfront (class in fusetools.pm_tools), 65

Y

yoy_comp() (fusetools.analytics_tools.Pandas class method), 13