List Management Documentation

Release 4.0

Ricky Schools

Sep 23, 2018

Contents:

CHAPTER 1

List Management (L.I.M.A)

A python based software that ingests 3rd-party tabular (excel, csv, etc.) files for conference and rep lists and aims to ensure that a Salesforce CRM is up-to-date.

At it's core, LIMA is a matching system that can push necessary updates, given a data input. Enabling LIMA are a number of great libraries:

- pandas
- salesforce-bulk
- selenium
- scikit-learn
- simple-salesforce
- us
- uszipcode

1.1 How lists are processed

1.2 Usage

1.2.1 Full process

To leverage the entire functionality of the list program (LIMA), you need only perform the below. New in version 4.0 is the ability to specify if you're manually running (ad-hoc) or if it's being controlled by a bot (cron'd/scheduled).

```
from ListManagement.list_processing import ListProcessing

# to run in manual/ad-hoc mode
lp = ListProcessing(mode='manual').main_contact_based_processing()

# to run in cron'd/auto mode
lp = ListProcessing(mode='auto').main_contact_based_processing()
```

1.2.2 Individual components

The modules within the search node of LIMA lend themselves well to ad-hoc/one-off searches. Using the tools like the below will parse your input file into one (or more) output files.

1.2.2.1 Finra

If you have a list of advisors where your data contains either a CRD number, or has the advisor's First, Last, and Company name, you can leverage the power of the Finra API. See the documentation for more details.

```
from ListManagement.search.finra import Finra
from PythonUtilities.LoggingUtility import Logging

log = Logging('Lima-Finra-AdHoc', 'LM_FAH').logger
fin = Finra(log=log)

path = '~/your/path/to/a/file.xlsx'

fin.scrape(path=path, parse_list=True, save=True)
```

1.2.2.2 Salesforce

Similar to the Finra scenario, if you have a file with CRD, Email, or or has the advisor's First, Last, and Company name, you can leverage the power of the Salesforce Search API. See the documentation for more details.

```
from ListManagement.search.salesforce import Search
from PythonUtilities.LoggingUtility import Logging

log = Logging('Lima-Searc-AdHoc', 'LM_SAH').logger
search = Search(log=log)

path = '~/your/path/to/a/file.xlsx'

search.perform_search_one(searching_list_path=path, list_type='Campaign')
```

1.2.2.3 Search chaining

Similar to how the LIMA uses these modules, you can manufacture (if necessary) a similar process to take a given file, search it against 1) Salesforce, 2) attempt to identify any not found via a BrokerCheck search (Finra) and 3) research against Salesforce.

```
from ListManagement.search import salesforce, finra
from PythonUtilities.LoggingUtility import Logging

log = Logging('Lima-Finra-AdHoc', 'LM_CAH').logger
sf_search = salesforce.Search(log)
fin_search = finra.Finra(log)
```

(continues on next page)

(continued from previous page)

```
extracted = {'File Path': '~/your/path/to/a/file.xlsx'}

extracted.update(sf_search.perform_search_one(searching_list_path=extracted['File Path
↪'], list_type='Campaign'))
extracted.update(fin_search.scrape(path=extracted['File Path'], scrape_type='crd',␣
↪parse_list=True))
extracted.update(sf_search.perform_search_two(extracted['FINRA_SEC Found'], extracted[
↪'Found Path'], list_type='Campaign'))
```

Written by Ricky Schools, with help from Max Charles, for FS Investments.

CHAPTER 2

ListManagement package

2.1 list_processing.py

The core module that orchestrates matching for 3rd party documents (excel, csv, et al) and updates the Salesforce CRM for FS Investments.

Handles Broker Dealer (Account) and Business Development Group (BizDev Group) updated representative lists (typically received monthly and quarterly, respectively).

Handles attendee lists for events and conferences (Campaigns) where FS Investments has committed money.

class ListManagement.list_processing.**ListProcessing**(*mode='manual'*)
Bases: object

Orchestrates all matching and updates for pending list requests.

account_processing(*_vars*)
Handles all processing for lists that are sourced from the account object.

1) Predicts headers and pre_process each file.
2) Searches against SalesForce for matches on: a) CRDNumber b) AMPFId c) Email d) LookupName.
3) If advisors are not found, scrape data from FINRA based on First, Last, and Account Name.
4) If advisors are scraped, re-search those advisors against SalesForce.
5) For all found advisors, scrape their licenses (and other metadata) from FINRA
6) Parse list into actionable pieces: a) upload to campaign b) create c) not found.
7) Update the 'Last List Upload' field on the account record.
8) Prepare data for upload into SalesForce (via source_channel function), do so for each file created above.
9) Extract stats from search processing, send notification email, and update SalesForce list record.
10) If applicable, drop files in the bulk_processing network drive to create or update SaleForce contacts.

Parameters **_vars** – Python dictionary containing metadata regarding a list request.

Returns

Return type Nothing

bizdev_processing(*_vars*)

Handles all processing for lists that are sourced from the BizDev Group object.

1) Predicts headers and pre_process each file.
2) Searches against SalesForce for matches on: a) CRDNumber b) AMPFId c) Email d) LookupName.
3) If advisors are not found, scrape data from FINRA based on First, Last, and Account Name.
4) If advisors are scraped, re-search those advisors against SalesForce.
5) For all found advisors, scrape their licenses (and other metadata) from FINRA
6) Parse list into actionable pieces: a) upload to campaign b) create c) not found.
7) Update the 'Last List Upload' field on the bizdev group record.
8) Prepare data for upload into SalesForce (via source_channel function), do so for each file created above.
9) Extract stats from search processing, send notification email, and update SalesForce list record.
10) If applicable, drop files in the bulk_processing network drive to create or update SaleForce contacts.

Parameters **_vars** – Python dictionary containing metadata regarding a list request.

Returns

Return type Nothing

campaign_processing(*_vars*)

Handles all processing for lists that are sourced from the campaign object.

1) Predict headers and pre_process each file.
2) Searches against SalesForce for matches on: a) CRDNumber b) AMPFId c) Email d) LookupName.
3) If advisors are not found, scrape data from FINRA based on First, Last, and Account Name.
4) If advisors are scraped, re-search those advisors against SalesForce contact list.
5) Parse list into actionable pieces: a) upload to campaign b) create c) not found.
6) Prepare data for upload into SalesForce (via source_channel function), do so for each file created above.
7) Extract stats from search processing, send notification email, and update SalesForce list record.
8) Ff applicable, drop file in the bulk_processing network drive to create or update SaleForce contacts.

Parameters **_vars** – Python dictionary containing metadata regarding a list request.

Returns

Return type Nothing

create_log_record_of_current_list_data(*msg*)

Catches a 'fatal error' incurred by the list program. Halts all processing if called.

Parameters **msg** – Error message.

Returns

Return type Nothing

finra_search_and_search_two(*_vars*)

Handles all Finra and secondary searching (if necessary).

1) If advisors are not found and are missing CRDNumbers, scrape FINRA
2) If FINRA is scraped, search our SalesForce database to increase our likely match-rate.

Parameters **_vars** – Python dictionary containing metadata regarding a list request.

Returns

Return type Updated python dictionary containing metadata regarding a list request.

is_bad_extension(*_vars*)

Checks if the a list request has an extension that can be processed. If a list is unable to be processed, notify the requester of the situation.

Parameters **_vars** – Python dictionary containing metadata regarding a list request.

Returns

Return type Boolean (True or False)

main_contact_based_processing()

This method determines how to route a pending list request, based on the Salesforce object type.

An abstracted path looks like the following:

1) Check if there are pending lists. # TODO: Add check before looping through each list.
2) If lists are pending, begin looping through all available lists.
3) **Check that each pending list is of a workable extension.**

   See Also: ListProcessing().is_bad_extension() method for extensions that will stop the processing.
4) **If the list has an acceptable extension, enter a** routing statement to determine which rules to process the list request through.
5) **After completing processing, notify the requester, record stats, and** begin processing any additional lists.

Returns

Return type Nothing

CHAPTER 3

ListManagement.search package

3.1 finra.py

Provides an 'API' via Selenium for extracting metadata on registered financial advisors from Finra BrokerCheck.

class ListManagement.search.finra.**Finra**(*log=None*)

Bases: object

A web scraper that leverages FINRA to:

1) Find the CRD of a financial advisor given a 'Lookup Name'. (First, Last, Firm Name)
2) Find metadata (see self._x_paths keys below) on a financial advisor, given their CRD Number.

return_scraped_data(*df*, *on*)

Transforms a dictionary (containing scraped advisor metadata) into a pandas data frame. :param df: A pandas data frame containing the original information/advisors to scrape. :param on: A string; Represents what we should

Returns

Return type The original pandas data frame joined with the scraped data frame.

scrape(*path*, *alter_name=None*, *scrape_type=None*, *parse_list=False*, *save=False*)

User level method allowing them access to the Finra 'API'. Given a tabular file (Excel, CSV), a user can specify:

1) The scrape type ('all', see self._x_paths above, or 'crdnumber') they desire to perform.
2) If they want to save the results of the file.
3) **An option to parse the results of the scrape into 3 parts: (typically for scrape_type='crdnumber'**

a) Found

b) Not found

c) Ambiguous (due to multiple advisors having the same name)

Parameters

- **path** – A string; Represents a full file path.
- **alter_name** – A string; Represents a suffix to append to a full file path.
- **scrape_type** – A string; The scrape type ('all', see self._x_paths above, or 'crdnumber') they desire to perform.
- **parse_list** – Boolean; Used to enable the list parsing (see 3) above).
- **save** – Boolean; Determines whether scraped results are saved to a file path or returned to the user in a console.

Returns

- *Dependent on save parameter above.*
- *Scraped results are saved to a file path or returned to the user in a console.*

3.2 salesforce.py

Provides a set of actions to compare the Contact object of a Salesforce CRM to a 3rd party list (excel, csv).

class ListManagement.search.salesforce.**Search**(*log=None*)

Bases: object

Search object that orchestrates comparisons and parsing between source (3rd party, Finra, et al) data sets and target (Salesforce contact object) data sets.

perform_search_one(*searching_list_path*, *list_type*)

User method to implement the first comparison between a 3rd party list and Salesforce.

Parameters

- **searching_list_path** – String; Represents a full file path.
- **list_type** – String; Represents the name of a Salesforce object.

Returns

Return type A python dictionary containing next steps for list processing.

perform_search_two(*searching_list_path*, *found_path*, *list_type*)

User method to implement the second comparison between a 3rd party list and Salesforce.

All advisors found during the call of this method are moved from the searching_list_path to the found_path.

Parameters

- **searching_list_path** – String; Represents a full file path of the un-matched advisors.
- **found_path** – String; Represents a full file path of the matched advisors.
- **list_type** – String; Represents the name of a Salesforce object.

Returns

Return type A python dictionary containing next steps for list processing.

perform_sec_search(*searching_list_path*, *found_path*)

CHAPTER 4

ListManagement.search.ml package

ListManagement.search.ml.header_predictions.**pre_processing**(*df*, *obj*)

ListManagement.search.ml.header_predictions.**predict_headers_and_pre_processing**(*path*, *obj*, *log*, *mode*)

class ListManagement.search.ml.model.**HeaderPredictions**(*log=None*, *use_saved=False*, *run_diagnostics=False*)

Bases: object

create_training_features(*headers*, *t_type='train'*)

data_preprocessing()

diagnostics(*save=False*)

predict(*predict_path*, *obj*)

CHAPTER 5

ListManagement.sources package

5.1 accounts.py

Contains functions relative to processing lists requests sourced from the Account object in Salesforce.

`ListManagement.sources.accounts.`**`make_sc`**(*path*, *frame*, *record_name*, *obj_id*, *obj*)

Function to manufacture a 'source_channel' for contacts that need to be created.

A source channel gives FS Investments the ability to see from where and when a contact record was sourced from.

Parameters

- **`path`** – String; Represents a full file path.
- **`frame`** – Pandas data frame; Object containing relational data.
- **`record_name`** – String; Represents the name of a salesforce record.
- **`obj_id`** – String; Represents the id of a Salesforce record
- **`obj`** – String; Represents the name of a Salesforce object.

Returns

Return type Tuple; updated pandas dataframe, move to bulk (boolean), and to_create_path (string)

`ListManagement.sources.accounts.`**`parse`**(*path*, *frame*, *dict_elements*)

Function to help parse Account list requests into actionable Salesforce jobs (update, create).

Parameters

- **`path`** – String; Represents a full file path to the source list.
- **`frame`** – Pandas data frame
- **`dict_elements`** – Dictionary; Contains metadata generated during list processing.

Returns

Return type Tuple; updated dictionary and list of files created.

5.2 bdgs.py

Contains functions relative to processing lists requests sourced from the BizDev Group object in Salesforce.

ListManagement.sources.bdgs.**make_sc**(*path*, *frame*, *record_name*, *obj_id*, *obj*, *aid*)
Function to manufacture a 'source_channel' for contacts that need to be created.

A source channel gives FS Investments the ability to see from where and when a contact record was sourced from.

Parameters

- **path** – String; Represents a full file path.
- **frame** – Pandas data frame; Object containing relational data.
- **record_name** – String; Represents the name of a salesforce record.
- **obj_id** – String; Represents the id of a Salesforce record
- **obj** – String; Represents the name of a Salesforce object.
- **aid** – String; Represents the Id of an Account object in Salesforce.

Returns

Return type Tuple; updated pandas dataframe, move to bulk (boolean), and to_create_path (string)

ListManagement.sources.bdgs.**parse**(*path*, *frame*, *dict_elements*)
Function to help parse BizDev Group list requests into actionable Salesforce jobs (update, create).

Parameters

- **path** – String; Represents a full file path to the source list.
- **frame** – Pandas data frame
- **dict_elements** – Dictionary; Contains metadata generated during list processing.

Returns

Return type Tuple; updated dictionary and list of files created.

5.3 campaigns.py

Contains functions relative to processing lists requests sourced from the Campaign object in Salesforce.

ListManagement.sources.campaigns.**make_sc**(*path*, *frame*, *record_name*, *obj_id*, *obj*)
Function to manufacture a 'source_channel' for contacts that need to be created.

A source channel gives FS Investments the ability to see from where and when a contact record was sourced from.

Parameters

- **path** – String; Represents a full file path.
- **frame** – Pandas data frame; Object containing relational data.
- **record_name** – String; Represents the name of a salesforce record.
- **obj_id** – String; Represents the id of a Salesforce record
- **obj** – String; Represents the name of a Salesforce object.

Returns

Return type Tuple; updated pandas dataframe, move to bulk (boolean), and to_create_path (string)

ListManagement.sources.campaigns.**parse**(*path*, *frame*, *dict_elements*, *event_timing*)

Function to help parse Campaign list requests into actionable Salesforce jobs (update, create).

Parameters

- **path** – String; Represents a full file path to the source list.
- **frame** – Pandas data frame
- **dict_elements** – Dictionary; Contains metadata generated during list processing.
- **event_timing** – String; Denotes whether a campaign has happened or not.

Returns

Return type Tuple; updated dictionary and list of files created.

CHAPTER 6

ListManagement.legacy package

ListManagement.legacy.get_sf_adv_list.**run**(*path_name*, *logger*)

ListManagement.legacy.sf_adv_formatting.**clean_dates**(*frame*, *headers*)

ListManagement.legacy.sf_adv_formatting.**date_for_move**(*loc_orig*, *loc_to*)

ListManagement.legacy.sf_adv_formatting.**make_lookup_name**(*df*, *jstr*)

ListManagement.legacy.sf_adv_formatting.**needs_update_flag**(*frame*, *headers*, *activity_range*, *sales_range*)

ListManagement.legacy.sf_adv_formatting.**shorten_data**(*df*, *col_name*, *x*)

CHAPTER 7

ListManagement.utility package

ListManagement.utility.email_helper.**attachment_reader**(*remove=False*, *raw=None*, *att=None*)

ListManagement.utility.email_helper.**body_parse**(*message*, *s_string*)

ListManagement.utility.email_helper.**close_mailbox_connection**(*m*)
closes the mailbox connection

Parameters **m** – mailbox object

Returns dictionary of mailbox information.

ListManagement.utility.email_helper.**craft_notification_email**(*items*)
creates the actual text values of the email.

Parameters **items** – dictionary items for stats processing

Returns body of email.

ListManagement.utility.email_helper.**determine_id_and_object_from_link**(*tmp*, *email_text*, *log*)

ListManagement.utility.email_helper.**email_parser**(*sender_name*, *look1*, *look2=None*)
parses the body text of an email_handler message

Parameters

- **sender_name** – text of an email_handler message (required)
- **look1** – start / end location of the text to parse (required)
- **look2** – optional - takes a secondary substring if finding text

Returns parsed substring

ListManagement.utility.email_helper.**get_decoded_email_body**(*message_body*)

ListManagement.utility.email_helper.**get_msg_part**(*msg_part*, *array*)
decodes the email_handler body from the email_handler data

Parameters

- **msg_part** – coded message string (required)
- **array** – items to parse

Returns decoded text of email_handler message

ListManagement.utility.email_helper.**handle_list_queue_requests**(*num*, *f_data*, *list_queue*)

ListManagement.utility.email_helper.**info_parser**(*body*, *look*, *look2=None*, *n=None*)
parses the body text of an email_handler message

Parameters

- **body** – text of an email_handler message (required)
- **look** – start / end location of the text to parse (required)
- **look2** – optional - takes a secondary substring if finding text
- **n** – length of where attachment link is

Returns parsed substring

ListManagement.utility.email_helper.**lists_in_queue**(*var_list*)
determines if there are any lists in the queue.

Parameters **var_list** – dictionary of list variables.

Returns boolean TRUE / FALSE

class ListManagement.utility.email_reader.**MailBoxReader**(*log*)
Bases: object

associate_email_request_with_sf_object(*dict_data*, *att*, *sender_addr*)

close_mailbox()

determine_path_and_complete_processing(*num*, *dict_data*, *att*, *msg_body*, *sender_addr*)

extract_pending_lists(*mailbox*, *folder*)

handle_new_email_requests(*num*, *raw_email*)

iterative_processing(*msg_list*)

class ListManagement.utility.email_reader.**ReturnDict**(*item*, *email_var*)
Bases: object

7.1 general.py

Houses functions that are used throughout the list processing process.

ListManagement.utility.general.**auto_maintain**(*directory*, *destination=None*, *ndays=30*, *log=None*)
This function helps to maintain a given directory by deleting files older than ndays old.

Parameters

- **directory** – A string representing the directory (folder) to maintain.
- **destination** – Optional. A string representing the directory to move a file to.
- **ndays** – An integer representing a ceiling for the age of files to keep.
- **log** – A logger instance.

Returns

Return type Nothing

ListManagement.utility.general.**clean_date_values**(*d_value*)
Transforms a string into a datetime object. :param d_value: A string representing a date.

Returns

Return type A datetime object.

ListManagement.utility.general.**clean_phone_number**(*number*)
Aims to normalize the formatting of a phone number.

Parameters **number** – A string representation of a phone number.

Returns

Return type A normalized & updated string representation of a phone number.

ListManagement.utility.general.**cmdorgui**()
Helper method to itentify if the running version of python is command line or GUI.

Returns

Return type Nothing

ListManagement.utility.general.**convert_unicode_to_date**(*date_string*)
Transforms a string containing a date into a datetime object. Decides if the date is in the future or in the past.

Parameters **date_string** – A string representing a date.

Returns

Return type A tuple containing the 1) datetime object and 2) if the date is in the future or past.

ListManagement.utility.general.**create_dir_move_file**(*path*)
Creates and moves a given file from it's current directory to a child directory.

Parameters **path** – A string representing a full file path.

Returns

Return type A string representing the new full file path location.

ListManagement.utility.general.**create_path_name**(*path*, *new_name*)
Creates a new file name given a path and a new name. :param path: A string representing the original name of a file. :param new_name: A string representing a new suffix to append to the existing file name.

Returns

Return type An updated string representing the new file name.

ListManagement.utility.general.**date_parsing**(*str_date_value*)
Converts a string to a datetime object.

Parameters **str_date_value** – String representation of a date.

Returns

Return type Datetime object of given date.

ListManagement.utility.general.**date_to_string**(*d_value*)
Turns a datetime object into a string. :param d_value: A datetime object.

Returns

Return type The string representation of a date.

ListManagement.utility.general.**determine_ext**(*f_name*)
Determines the extension of a file given a file name.

Parameters **f_name** – Name of a file. Examples: ABC_1234.xlsx, my_file.csv

Returns

- *The extension of a file and it's length.*
- **Examples** (*(5, .xlsx), (4, .csv)*)

ListManagement.utility.general.**determine_move_to_bulk_processing**(*df*)
Determines of a list request contains the necessary columns (meta-data) to be passed to Business Solutions' Salesforce bulk processing tool.

Parameters **df** – A pandas data frame containing a list.

Returns

Return type Boolean (True or False)

ListManagement.utility.general.**drop_in_bulk_processing**(*path*, *log*)
Drops a file and passes it to the bulk tool Business Solution built. :param path: A string representing a full file name. :param log: A log instance.

Returns

Return type Nothing

ListManagement.utility.general.**drop_unneeded_columns**(*df, obj, ac=['CRDNumber', 'FirstName', 'LastName', 'AccountId', 'MailingStreet', 'MailingCity', 'MailingState', 'MailingPostalCode', 'SourceChannel', 'Email', 'Website', 'AUM', 'GDC', 'Fax', 'HomePhone', 'MobilePhone', 'Phone'], create=True, bdg=False*)
Removes all columns that are not needed by Business Solutions' Salesforce bulk uploading tool.

Parameters

- **df** – A pandas data frame containing a list.
- **obj** – A string representing a Salesforce object.
- **ac** – A list containing the columns accepted by the bulk tool.
- **create** – Boolean. Denotes if this call is coming from a 'to_create' contacts request or not.
- **bdg** – Boolean. Denotes if this call is coming from a BizDev Group request or not.

Returns

- *An updated pandas data frame object containing only the*
- *bulk tool's accepted columns.*

ListManagement.utility.general.**duration**(*start*, *end*)
Creates a string representation of the time elapsed between start and end.

Parameters

- **start** – Timestamp/datetime object of when a process began.

- **end** – Timestamp/datetime object of when a process ended.

Returns

Return type A formatted time representing the elapsed time in hours, minutes, and seconds.

ListManagement.utility.general.**find_chrome_driver_location**(*filename='chromedriver'*)
Helper method to find the location of 'chromedriver'.

Parameters **filename** – The name of a file to find.

Returns

Return type Location where given file is found.

ListManagement.utility.general.**is_path**(*path*)
Checks if a given path is a file.

Parameters **path** – A string. Should be a path to a directory or file.

Returns

Return type Boolean (True or False)

ListManagement.utility.general.**last_list_uploaded_data**(*object_id*)
Builds a list containing the 'object_id' and the current time (in ISO-format).

Parameters **object_id** – A string representation of a Salesforce object id.

Returns

- *A list.*
- **Examples** (*[object_id, current time in ISO-format.]*)

ListManagement.utility.general.**lower_head_values**(*lname*)
Converts a string to lower-case. :param lname: A string.

Returns

Return type A lower-case string.

ListManagement.utility.general.**myprogressbar**(*batchsize*, *totalsize*, *barlength=25*, *message=''*, *char='#'*)
Helper method to display a progress bar.

Parameters

- **batchsize** – An integer representing the number of records processed.
- **totalsize** – An integer representing the size of a list.
- **barlength** – A customizable integer representing the length of a bar.
- **message** – A string customizing representing the name of the bar.
- **char** – A string. The character to populate the progress bar.

Returns

Return type Nothing

ListManagement.utility.general.**path_leaf**(*path*)
Dynamic function to return a file or folder name. :param path: A string representing a file name.

Returns

Return type A file name or a directory.

ListManagement.utility.general.**record_processing_stats**(*values*, *save=True*)
Records a backup of the metadata associated from processing a list in SQL and Excel.

Parameters

- **values** – A collection of values to record related to a lists processing.
- **save** – Boolean (True or False). Dictates whether to save or return a data frame.

Returns

- *A dictionary (if save=True) or pandas data frame representing*
- *the stats recorded during list processing (if save=False).*

ListManagement.utility.general.**remove_underscores**(*line*)
Replaces underscores with spaces.

Parameters **line** – A string representing a single cell of a pandas data frame.

Returns

Return type A updated string, without underscores.

ListManagement.utility.general.**save_conf_creation_meta**(*sc*, *objid*, *status*)

ListManagement.utility.general.**shorten_fname_to_95chars**(*f_name*)
Shortens a filename to less than 95 characters.

Parameters **f_name** – A string representing a file name.

Returns

Return type A string representing a file name (updated, if necessary)

ListManagement.utility.general.**split_by_uppers**(*line*)
Takes a string and if no spaces are present, splits them by the present upper case values.

Parameters **line** – A string value.

Returns

Return type An updated string value.

ListManagement.utility.general.**split_dir_name**(*full_path*)
Provides a file name given a full path.

Parameters **full_path** – A string representation of a full file path.

Returns

Return type A file name.

ListManagement.utility.general.**split_name**(*path*)
Splits a path into 1) the containing folder and 2) the file name.

Parameters **path** – A string representing a full file path.

Returns

Return type A string representing a file name.

ListManagement.utility.general.**strip_unicode_chars**(*row*)
Attempts to coerce all data to UTF-8.

Parameters **row** – A string value, representing a single cell (value)

Returns

Return type A coerced string value.

ListManagement.utility.general.**timedelta_to_processing_str**(*duration*)
Transforms a timestamp into a string representing the time in days, hours, minutes, & seconds. :param duration: A timestamp object.

Returns

Return type String representation of duration.

ListManagement.utility.pandas_helper.**concat_dfs**(*df_list*)

ListManagement.utility.pandas_helper.**determine_num_records**(*path*)

ListManagement.utility.pandas_helper.**is_null**(*x*)

ListManagement.utility.pandas_helper.**make_df**(*data=None*, *columns=None*)

ListManagement.utility.pandas_helper.**new_stat_line**(*value_dict*)
writes the new line of data to the stats dataframe

Parameters **value_dict** – values to add to stats

Returns dataframe of stats to record.

ListManagement.utility.pandas_helper.**read_df**(*path*)

ListManagement.utility.pandas_helper.**save_df**(*df*, *path*)

7.2 processes.py

Contains core processes that are used throughout the processing of a given list.

ListManagement.utility.processes.**bdg_upload**(*session*, *data*, *obj_id*, *obj*, *col_num*, *df_path*, *remove_path=None*, *add_path=None*, *update_path=None*, *curr_memb=None*, *n_add=0*, *n_up=0*, *n_re=0*)
Adds/updates advisors in a Salesforce BizDev Group. :param session: An authenticated Salesforce REST API session. :param data: A list containing the data which will be uploaded to Salesforce. :param obj_id: An 18-char string; Represents an Id of a Salesforce object. :param obj: A string; Represents the name of a Salesforce object. :param col_num: An integer; The column to use for when parsing a list. :param df_path: A string; Represents the a full file path. :param remove_path: A string; Represents the name of the 'to_remove' file name. :param add_path: A string; Represents the name of the 'to_add' file name. :param update_path: A string; Represents the name of the 'to_update' file name. :param curr_memb: REMOVE; NOT USED. :param n_add: An integer; Represents the number of advisors adding to a campaign. :param n_up: An integer; Represents the number of advisors updating in a campaign. :param n_re: An integer; Represents the number of advisors removing from a campaign.

Returns

Return type A tuple lists containing [paths], and [num records]

ListManagement.utility.processes.**cmp_upload**(*session*, *data*, *obj_id*, *obj*, *n_re=0*, *n_added=0*, *n_uptd=0*)
Adds/updates campaign members (and their statuses) in a Salesforce campaign.

Parameters

- **session** – An authenticated Salesforce REST API session.
- **data** – A list containing the data which will be uploaded to Salesforce.
- **obj_id** – An 18-char string; Represents an Id of a Salesforce object.

- **obj** – A string; Represents the name of a Salesforce object.
- **n_re** – An integer; Represents the number of advisors removing from a campaign.
- **n_added** – An integer; Represents the number of advisors adding to a campaign.
- **n_uptd** – An integer; Represents the number of advisors updating in a campaign.

Returns

Return type A tuple lists containing [paths], and [num records]

ListManagement.utility.processes.**extract_dictionary_values**(*dict_data*, *log=None*)

This function leverages the dictionary, filled with meta data regarding a list and it's processing and extracts specific elements that will be used to:

1) Organize all list processing statistics to record and store. Ex. (Num names on list, match rate, et al).
2) Send a notification of process completion to the originator of a list request.

Parameters

- **dict_data** – A python dictionary containing meta data regarding a single list in the queue.
- **log** – A logger instance.

Returns

Return type A python dictionary containing next steps for the list program.

ListManagement.utility.processes.**id_preprocessing_needs**(*path*)

ListManagement.utility.processes.**parse_list_based_on_type**(*path*, *l_type=None*, *pre_or_post=None*, *log=None*, *to_create_path=None*)

Serves as a routing point to parse a list, dependent on the sources object.

Parameters

- **path** – A string representing a full file path.
- **l_type** – A string containing the source Salesforce object for the list request.
- **pre_or_post** – A string containing 'Pre' or 'Post'.
- **log** – A logger instance.
- **to_create_path** – The name of file ending with the '_to_create' suffix.

Returns

Return type An updated dictionary mapper object.

ListManagement.utility.processes.**sfdc_upload**(*path*, *obj*, *obj_id*, *session*, *log=None*)

A routing method which performs final preparation and orchestrates a Salesforce upload for a given file.

Parameters

- **path** – A string representing a full file path.
- **obj** – A string representing the name of a Salesforce object.
- **obj_id** – An 18-char string; The id of a Salesforce object.
- **session** – An authenticated Salesforce REST API session.
- **log** – A logger instance.

Returns

Return type A python dictionary containing metadata and next steps for the list program.

ListManagement.utility.processes.**source_channel**(*path*, *record_name*, *obj_id*, *obj*, *aid=None*, *log=None*)

Prepares a list for Salesforce upload and/or contact creation.

Parameters

- **path** – A string representing a full file path.
- **record_name** – A string; name of a Salesforce record.
- **obj_id** – An 18-char string; Id of a Salesforce Object.
- **obj** – A string; Name of a Salesforce Object.
- **aid** – An 18-char string; Optional. Parent Id (or Account Id) related to an Object in Salesforce.
- **log** – A logger instance.

Returns

Return type A python dictionary containing next steps for the list processing tool.

ListManagement.utility.processes.**upload**(*session*, *headers*, *data*, *obj_id*, *obj*, *col_num=None*, *df_path=None*)

Helps to perform an actual salesforce upload.

Parameters

- **session** – An authenticated Salesforce REST API session.
- **headers** – A list of strings representing column names in a Salesforce table. Should consider removing, as it doesn't seem used.
- **data** – A list containing the data which will be uploaded to Salesforce.
- **obj_id** – An 18-char string; Represents an Id of a Salesforce object.
- **obj** – A string; Represents the name of a Salesforce object.
- **col_num** – TBD
- **df_path** – A string representing the a full file path.

Returns

Return type TBD

7.3 queue.py

Extracts necessary metadata from Salesforce for pending list requests.

ListManagement.utility.queue.**build_queue**(*sfdc*, *log=None*)

Queries Salesforce to extract any pending lists (and necessary metadata).

Parameters

- **sfdc** – Authenticated Salesforce REST API session.
- **log** – log object.

Returns

Return type Dictionary of pending lists in the queue and necessary metadata.

7.4 sf_helper.py

Contains helper functions that are useful when preparing data to upload to Salesforce.

ListManagement.utility.sf_helper.**cmp_mbr_id_for_contact_id**(*update_list*, *obj_list*)

ListManagement.utility.sf_helper.**get_user_id**(*sf*, *obj_id*, *att*, *user_email*)
Given an email address, returns a list of metadata to use when uploading to salesforce. :param sf: An authenticated Salesforce REST API session. :param obj_id: An 18-char string; Represents an Id of a Salesforce object. :param att: A string; Represents the name of an attachment. :param user_email: A string; Represents a given user's email address.

Returns

Return type A list of metadata to use when uploading to salesforce

ListManagement.utility.sf_helper.**headers_clean_up**(*headers*, *to_remove='ContactID'*)
Removes an element (or group of elements) from a list.

Parameters

- **headers** – A list of column names.
- **to_remove** – An element (or group of elements) to remove from a list.

Returns

Return type A sliced list.

ListManagement.utility.sf_helper.**remove_duplicates**(*mbr_list*)
Helper function to 'uniqify' a list. :param mbr_list: A list of lists containing duplicates.

Returns

Return type A unique set of lists.

ListManagement.utility.sf_helper.**split_list**(*id_in_obj*, *ids_from_search*, *obj_id*, *obj*, *col=None*, *remove=None*, *remove_unique=None*)
Helper function to take two lists and parse them into 2 to 3 different lists.

1) If any 'ids_from_search' aren't present in 'id_in_obj', add to insert.
2) If any 'ids_from_search' are present in 'id_in_obj', add to update.
3) **If any 'id_in_obj' aren't present in 'ids_from_search', add to remove.** (Only happens for BizDev Group Lists)

Parameters

- **id_in_obj** – A list of unique identifiers present in a Salesforce object.
- **ids_from_search** – A list of unique identifiers present in a 3rd party list.
- **obj_id** – An 18-char string; Represents an Id of a Salesforce object.
- **obj** – A string; Represents the name of a Salesforce object.
- **col_num** – An integer; The column to use for when parsing a list.
- **remove** – An empty list.
- **remove_unique** – REMOVE AS NOT USED.

Returns

Return type A tuple of lists (insert, update, remove)

CHAPTER 8

ListManagement.cron_jobs package

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Python Module Index

I

ListManagement, **??**
ListManagement.cron_jobs, **??**
ListManagement.cron_jobs.created_contacts_to_campaign,
??
ListManagement.cron_jobs.ml_diagnostics_update,
??
ListManagement.cron_jobs.requested_email_job,
??
ListManagement.legacy, **??**
ListManagement.legacy.get_sf_adv_list,
??
ListManagement.legacy.sf_adv_formatting,
??
ListManagement.legacy.sf_adv_query, **??**
ListManagement.list_processing, **??**
ListManagement.search, **??**
ListManagement.search.finra, **??**
ListManagement.search.ml, **??**
ListManagement.search.ml.header_predictions,
??
ListManagement.search.ml.model, **??**
ListManagement.search.salesforce, **??**
ListManagement.sources, **??**
ListManagement.sources.accounts, **??**
ListManagement.sources.bdgs, **??**
ListManagement.sources.campaigns, **??**
ListManagement.utility, **??**
ListManagement.utility.email_helper, **??**
ListManagement.utility.email_reader, **??**
ListManagement.utility.general, **??**
ListManagement.utility.pandas_helper,
??
ListManagement.utility.processes, **??**
ListManagement.utility.queue, **??**
ListManagement.utility.sf_helper, **??**