pdsql Documentation

Release 1.0.4

Mike Kittridge

Apr 27, 2018

Modules

The pdsql package contains convenience functions for adding, manipulating, and changing data in SQL servers with a emphasis on Pandas DataFrames for the handling of data in Python.

At the moment, the only supported SQL system is MSSQL, but other SQL systems can/will be added in the future through the better implementation of Sqlalchemy. Priority will be given to PostgreSQL and SQLite/Spatialite.

CHAPTER 1

create_engine

The create_engine function is used to create an appropriate database engine through Sqlalchemy to interact with SQL databases.

1.1 MSSQL

The mssql module contains a variety of functions to interact with MSSQL databases through Python and Pandas.

1.1.1 Reading tables

pdsql.mssql.**rd_sql**(*server*, *database*, *table=None*, *col_names=None*, *where_col=None*, *where_val=None*, *where_op='AND'*, *geo_col=False*, *from_date=None*, *to_date=None*, *date_col=None*, *rename_cols=None*, *stmt=None*, *export_path=None*)

Function to import data from an MSSQL database.

Parameters

- **server** (*str*) – The server name. e.g.: 'SQL2012PROD03'
- **database** (*str*) – The specific database within the server. e.g.: 'LowFlows'
- **table** (*str*) – The specific table within the database. e.g.: 'LowFlowSiteRestrictionDaily'
- **col_names** (*list of str*) – The column names that should be retrieved. e.g.: ['SiteID', 'BandNo', 'RecordNo']
- **where_col** (*str or dict*) – Must be either a string with an associated where_val list or a dictionary of strings to lists.'. e.g.: 'SnapshotType' or {'SnapshotType': ['value1', 'value2']}
- **where_val** (*list*) – The WHERE query values for the where_col. e.g. ['value1', 'value2']

- **where_op** (*str*) – If where_col is a dictionary and there are more than one key, then the operator that connects the where statements must be either 'AND' or 'OR'.
- **geo_col** (*bool*) – Is there a geometry column in the table?.
- **from_date** (*str*) – The start date in the form '2010-01-01'.
- **to_date** (*str*) – The end date in the form '2010-01-01'.
- **date_col** (*str*) – The SQL table column that contains the dates.
- **rename_cols** (*list of str*) – List of strings to rename the resulting DataFrame column names.
- **stmt** (*str*) – Custom SQL statement to be directly passed to the database. This will ignore all prior arguments except server and database.
- **export_path** (*str*) – The export path for a csv file if desired. If None, then nothing is exported.

Returns

Return type DataFrame

pdsql.mssql.**rd_sql_ts**(*server*, *database*, *table*, *groupby_cols*, *date_col*, *values_cols*, *resample_code=None*, *period=1*, *fun='mean'*, *val_round=3*, *where_col=None*, *where_val=None*, *where_op='AND'*, *from_date=None*, *to_date=None*, *min_count=None*, *export_path=None*)

Function to specifically read and possibly aggregate time series data stored in MSSQL tables.

Parameters

- **server** (*str*) – The server name. e.g.: 'SQL2012PROD03'
- **database** (*str*) – The specific database within the server. e.g.: 'LowFlows'
- **table** (*str*) – The specific table within the database. e.g.: 'LowFlowSiteRestrictionDaily'
- **groupby_cols** (*str or list of str*) – The columns in the SQL table to grouped and returned with the time series data.
- **date_col** (*str*) – The date column in the table.
- **values_cols** (*str or list of str*) – The column(s) of the value(s) that should be resampled.
- **resample_code** (*str or None*) – The Pandas time series resampling code. e.g. 'D' for day, 'W' for week, 'M' for month, etc.
- **period** (*int*) – The number of resampling periods. e.g. period = 2 and resample = 'D' would be to resample the values over a 2 day period.
- **fun** (*str*) – The resampling function. i.e. mean, sum, count, min, or max. No median yet...
- **val_round** (*int*) – The number of decimals to round the values.
- **where_col** (*str or dict*) – Must be either a string with an associated where_val list or a dictionary of strings to lists.'. e.g.: 'SnapshotType' or {'SnapshotType': ['value1', 'value2']}
- **where_val** (*list*) – The WHERE query values for the where_col. e.g. ['value1', 'value2']

- **where_op** (*str*) – If where_col is a dictionary and there are more than one key, then the operator that connects the where statements must be either 'AND' or 'OR'.
- **from_date** (*str*) – The start date in the form '2010-01-01'.
- **to_date** (*str*) – The end date in the form '2010-01-01'.
- **min_count** (*int*) – The minimum number of values required to return groupby_cols. Only works when groupby_cols and vlue_cols are str.
- **export_path** (*str*) – The export path for a csv file if desired. If None, then nothing is exported.

Returns Pandas DataFrame with MultiIndex of groupby_cols and date_col

Return type DataFrame

pdsql.mssql.**rd_sql_geo**(*server*, *database*, *table*, *col_stmt*, *where_lst=None*)

Function to extract the geometry and coordinate system from an SQL geometry field. Returns a shapely geometry object and a proj4 str.

Parameters

- **server** (*str*) – The server name. e.g.: 'SQL2012PROD03'
- **database** (*str*) – The specific database within the server. e.g.: 'LowFlows'
- **table** (*str*) – The specific table within the database. e.g.: 'LowFlowSiteRestrictionDaily'
- **where_lst** (*list*) – A list of where statements to be passed and added to the final SQL statement.

Returns

- *list of shapely geometry objects* – The main output is a list of shapely geometry objects for all queried rows of the SQL table.
- *str* – The second output is a proj4 str of the projection system.

1.1.2 Creating tables

pdsql.mssql.**create_mssql_table**(*server*, *database*, *table*, *dtype_dict*, *primary_keys=None*, *foreign_keys=None*, *foreign_table=None*, *drop_table=False*)

Function to create a table in an mssql database.

Parameters

- **server** (*str*) – The server name. e.g.: 'SQL2012PROD03'
- **database** (*str*) – The specific database within the server. e.g.: 'LowFlows'
- **table** (*str*) – The specific table within the database. e.g.: 'LowFlowSiteRestrictionDaily'
- **dtype_dict** (*dict of str*) – Dictionary of df columns to the associated sql data type. Examples below.
- **primary_keys** (*str or list of str*) – Index columns to define uniqueness in the data structure.
- **foreign_keys** (*str or list of str*) – Columns to link to another table in the same database.
- **foreign_table** (*str*) – The table in the same database with the identical foreign key(s).

- **drop_table** (*bool*) – If the table already exists, should it be dropped?

Returns

Return type None

1.1.3 Writing to tables

pdsql.mssql.**to_mssql**(*df*, *server*, *database*, *table*, *index=False*, *dtype=None*)

Function to append a DataFrame onto an existing mssql table.

Parameters

- **df** (*DataFrame*) – DataFrame to be saved. The DataFrame column/index names must match those on the mssql table exactly.
- **server** (*str*) – The server name. e.g.: 'SQL2012PROD03'
- **database** (*str*) – The specific database within the server. e.g.: 'LowFlows'
- **table** (*str*) – The specific table within the database. e.g.: 'LowFlowSiteRestrictionDaily'
- **index** (*bool*) – Should the index be added as a column?
- **dtype** (*dict of column name to SQL type, default None*) – Optional specifying the datatype for columns. The SQL type should be an SQLAlchemy type.

Returns

Return type None

1.1.4 Updating tables

pdsql.mssql.**update_mssql_table_rows**(*df*, *server*, *database*, *table*, *on*, *append=True*)

Function to selectively delete rows from an mssql table.

Parameters

- **df** (*DataFrame*) – DataFrame with data to be overwritten in SQL table.
- **server** (*str*) – The server name. e.g.: 'SQL2012PROD03'
- **database** (*str*) – The specific database within the server. e.g.: 'LowFlows'
- **table** (*str*) – The specific table within the database. e.g.: 'LowFlowSiteRestrictionDaily'
- **on** (*str or list*) – The columns for the df and sql table to join to to make the update.
- **stmt** (*str*) – SQL delete statement. Will override everything except server and database.

Returns

Return type None

1.1.5 Deleting rows in tables

pdsql.mssql.**del_mssql_table_rows**(*server*, *database*, *table=None*, *pk_df=None*, *stmt=None*, ***kwargs*)

Function to selectively delete rows from an mssql table.

Parameters

- **server** (*str*) – The server name. e.g.: 'SQL2012PROD03'
- **database** (*str*) – The specific database within the server. e.g.: 'LowFlows'
- **table** (*str or None if stmt is a str*) – The specific table within the database. e.g.: 'LowFlowSiteRestrictionDaily'
- **pk_df** (*DataFrame*) – A DataFrame of the primary keys of the table for the rows that should be removed. Will override anything in the kwargs.
- **stmt** (*str*) – SQL delete statement. Will override everything except server and database.
- ****kwargs** – Any kwargs that can be passed to sql_where_stmts.

Returns

Return type None

Notes

Using the pk_df is the only way to ensure that specific rows will be deleted from composite keys. The column data types and names of pk_df must match the equivelant columns in the SQL table. The procedure creates a temporary table from the pk_df then deletes the rows in the target table based on the temp table. Then finally deletes the temp table.

1.1.6 Helper functions

pdsql.mssql.**sql_where_stmts**(*where_col=None*, *where_val=None*, *where_op='AND'*, *from_date=None*, *to_date=None*, *date_col=None*)

Function to take various input parameters and convert them to a list of where statements for SQL.

Parameters

- **where_col** (*str or dict*) – Either a str with an associated where_val list or a dictionary of string keys to list values. If a str, it should represent the table column associated with the 'where' condition.
- **where_val** (*list or None*) – If where_col is a str, then where_val must be a list of associated condition values.
- **where_op** (*str of either 'AND' or 'OR'*) – The binding operator for the where conditions.
- **from_date** (*str or None*) – The start date in the form '2010-01-01'.
- **to_date** (*str or None*) – The end date in the form '2010-01-01'.
- **date_col** (*str or None*) – The SQL table column that contains the dates.

Returns Returns a list of str where conditions to be passed to an SQL execution function. The function needs to bind it with " where " + " and ".join(where_lst)

Return type list of str or None

pdsql.mssql.**sql_ts_agg_stmt**(*table*, *groupby_cols*, *date_col*, *values_cols*, *resample_code*, *period=1*, *fun='mean'*, *val_round=3*, *where_lst=None*)

Function to create an SQL statement to pass to an SQL driver to resample a time series table.

Parameters

- **table** (*str*) – The SQL table name.
- **groupby_cols** (*str or list of str*) – The columns in the SQL table to grouped and returned with the time series data.
- **date_col** (*str*) – The date column in the table.
- **values_cols** (*str or list of str*) – The column(s) of the value(s) that should be resampled.
- **resample_code** (*str*) – The Pandas time series resampling code. e.g. 'D' for day, 'W' for week, 'M' for month, etc.
- **period** (*int*) – The number of resampling periods. e.g. period = 2 and resample = 'D' would be to resample the values over a 2 day period.
- **fun** (*str*) – The resampling function. i.e. mean, sum, count, min, or max. No median yet...
- **val_round** (*int*) – The number of decimals to round the values.
- **where_lst** (*list or None*) – A list of where statements to be passed and added to the final SQL statement.

Returns A full SQL statement that can be passed directly to an SQL connection driver like pymssql through pandas read_sql function.

Return type str

1.1.7 API Pages

—

Index