Radio ads PostgreSQL DB

Release 1.0

Marcin Borowski

Mar 16, 2024

CONTENTS:

1 cs50sql_fin **1**
- 1.1 populate module 1
- 1.2 test module 4
- 1.3 tools package 4

2 Indices and tables **5**

Python Module Index **7**

Index **9**

CHAPTER

ONE

CS50SQL_FIN

1.1 populate module

populate.**add_10_fields**(*data_set: dict[DataFrame, str, list[str]]*) → None

Function adding data into ad_time_details table, which consists of 10 columns.

Parameters
data_set – A dict contaning data to be added, table name, and field / column name.

Data is a Pandas DataFrame, table name and field name are both strings. :raise KeyError: If key name does not match the pattern :raise psycopg.DataError: If table or field names doesn't match those in the DB :return: None

populate.**add_1_field**(*data: list*, *table_name: str*, *field_name: str*) → None

Skeleton function for adding data to single column tables.

Parameters

- **data** – List of strings or integers representing table contents
- **table_name** – String reprexsenting name of the table into which data is going to be added

Raises
psycopg.DataError – If data type does not match table restrictions

Returns
None

populate.**add_3_fields**(*data_set: dict[DataFrame, str, list]*) → None

Function adding data into mediums table, which consists of 3 columns.

Parameters
data_set – A dict contaning data to be added, table name, and field / column name.

Data is a Pandas DataFrame, table name and field name are both strings. :raise KeyError: If key name does not match the pattern :raise psycopg.DataError: If table or field names doesn't match those in the DB :return: None

populate.**add_8_fields**(*data_set: dict[DataFrame, str, list[str]]*) → None

Function adding data into ad_time_details table, which consists of 8 columns.

Parameters
data_set – A dict contaning data to be added, table name, and field / column name.

Data is a Pandas DataFrame, table name and field name are both strings. :raise KeyError: If key name does not match the pattern :raise psycopg.DataError: If table or field names doesn't match those in the DB :return: None

populate.**check_for_data_1_field**(*data_: list*, *table_name: str*, *field_name: str*) → tuple[bool, list[str]]

Skeleton function for checking if there is data inside each of one column tables. Ads data if there are any new entries, skips if no new data was found. If DB is empty returns immediately.

Parameters

- **data** – List containing data to be checked and added. Data is of str or int types.
- **table_name** – String representing name of the table into which data is going to be added
- **field_name** – String representing name of the field/ column name

Raises
psycopg.DataError – If data type does not match table restrictions

Returns
A tuple containing bool for logic purposes, anbd the data set to be added

Return type
tuple[bool, list[str/int]]

populate.**check_for_data_3_fields**(*fields: list[str]*, *table_name: str*, *submediums: DataFrame*) → tuple[bool, DataFrame]

Returns a bool for logic purposes and data to be added into mediums table. If DB is empty returns original DF. During data update process returns the data not present in the DB or indicates there is nothing to be added.

Parameters

- **fields** – A list containing field / column names represented as a str
- **table_name** – Name of the table into which data is going to be added as a str
- **submediums** – Pandas DataFrame containing data to add.

Raises
psycopg.DataError – If data type does not match table restrictions

Returns
Tuple containing bool for logic purposes and a Pandas DataFrame

as data to be added into the DB during the update :rtype: tuple[bool, pd.DataFrame]

populate.**get_colum_names**(*table_name: str*) → list[str]

A function which returns the names of selected table from the DB.

Raises
psycopg.DatabaseError – If column names does not match DB contents

Returns
List containing all the column names present in selected table.

Return type
list[str]

populate.**get_id_for_ad_time**(*fields: list[str]*, *table_: str*) → tuple[bool, DataFrame]

Gets IDs from reference tables to ad_time_details table. Mainly connects time details of singular ad emission with other tables containing details via IDs. This function populates one of two core tables in this DB. Returns a bool for logic purposes and data to be added into mediums.

Parameters

- **fields** – A list containing field / column names represented as a str
- **table** – Name of the table out of which the data is going to be pulled,

represented as a str :raise psycopg.DataError: If data type does not match table restrictions :return: Tuple containing bool for logic purposes and a Pandas DataFrame as data to be added into the DB during the update or initial DB fill. :rtype: tuple[bool, pd.DataFrame]

populate.**get_id_for_ads_desc**(*fields: list[str]*, *table_: str*) → tuple[bool, DataFrame]

Gets IDs from reference tables to ads_desc table. Mainly connects other tables and data of singular ad emission via IDs with other tables. This function populates one of two core tables in this DB. Returns a bool for logic purposes and data to be added into mediums.

Parameters

- **fields** – A list containing field / column names represented as a str
- **table** – Name of the table out of which the data is going to be pulled,

represented as a str :raise psycopg.DataError: If data type does not match table restrictions :return: Tuple containing bool for logic purposes and a Pandas DataFrame as data to be added into the DB during the update or initial DB fill. :rtype: tuple[bool, pd.DataFrame]

populate.**get_id_for_submediums**(*fields: list[str]*, *table_: str*) → tuple[bool, DataFrame]

Gets IDs from reference tables to mediums table. Mainly connects submediums with broadcaster and reach tables. Returns a bool for logic purposes and data to be added into mediums.

Parameters

- **fields** – A list containing field / column names represented as a str
- **table** – Name of the table out of which the data is going to be pulled,

represented as a str :raise psycopg.DataError: If data type does not match table restrictions :return: Tuple containing bool for logic purposes and a Pandas DataFrame as data to be added into the DB during the update or initial DB fill. :rtype: tuple[bool, pd.DataFrame]

populate.**get_index_val**(*table_name: str*) → int

Function gets max index value from the selected table and returns it as an integer increased by one. When the table is empty, returns 1

Parameters
table_name – Name of the table out of which the data is going to be pulled,

represented as a str :raise psycopg.DataError: If data type does not match table restrictions :return: number representiung max index value of selected table icreased by 1 :rtype: int

populate.**get_min_max_date**(*fields: list[str]*, *table_: str*, *dataframe: DataFrame*) → tuple[bool, DataFrame]

Gets max and min dates from selected table. Then checks if dates present in the DF passed as a parameter are outside of dates range. If so, allows data insertion into the DB, if not it informs the user, and proceedes with the rest of the code.

Parameters

- **fields** – A list containing field / column names represented as a str
- **table** – Name of the table out of which the data is going to be pulled,

represented as a str :param dataframe: Pandas DataFrame with the new data to be checked if not present in selected table :raise psycopg.DatabaseError: If column names does not match DB contents :return: Tuple containing bool for logic purposes and a Pandas DataFrame as data to be added into the DB during the update or initial DB fill. :rtype: tuple[bool, pd.DataFrame]

populate.**iter_over_inputs**(*data_set: list[dict[list, str, str]]*) → None

Main loop for iteration over one column tables.

Parameters
data_set – List containing dicts with data, table name and field/column name.

List contains strings or integers representing the data to be added into selected tables. :param table_name: String representing name of the table into which data is going to be added :raise KeyError: If key name does not match the pattern :return: None

1.2 test module

test.**add_10_fields**(*data_set: dict[DataFrame, str, list[str]]*) → None

test.**add_1_field**(*data*, *table_name*, *field_name*)

test.**add_3_fields**(*data_set*)

test.**add_8_fields**(*data_set: dict[DataFrame, str, list[str]]*) → None

test.**check_for_data_1_field**(*data_*, *table_*, *field_*)

test.**check_for_data_3_fields**(*fields: list[str]*, *table_: str*, *submediums: DataFrame*) → DataFrame

test.**get_colum_names**(*table_name*)

test.**get_id_for_ad_time**(*fields: list[str]*, *table_: str*) → tuple[bool, DataFrame]

test.**get_id_for_ads_desc**(*fields: list[str]*, *table_: str*) → tuple[bool, DataFrame]

test.**get_id_for_submediums**(*fields*, *table*)

test.**get_index_val**(*table_: str*) → int

test.**get_min_max_date**(*fields: list[str]*, *table_: str*, *dataframe: DataFrame*) → tuple[bool, DataFrame]

test.**iter_over_inputs**(*data_set*)

1.3 tools package

1.3.1 Submodules

1.3.2 tools.conf module

1.3.3 Module contents

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

populate, 1

t

test, 4
tools, 4
tools.conf, 4

INDEX

A

add_10_fields() (*in module populate*), 1
add_10_fields() (*in module test*), 4
add_1_field() (*in module populate*), 1
add_1_field() (*in module test*), 4
add_3_fields() (*in module populate*), 1
add_3_fields() (*in module test*), 4
add_8_fields() (*in module populate*), 1
add_8_fields() (*in module test*), 4

C

check_for_data_1_field() (*in module populate*), 1
check_for_data_1_field() (*in module test*), 4
check_for_data_3_fields() (*in module populate*), 2
check_for_data_3_fields() (*in module test*), 4

G

get_colum_names() (*in module populate*), 2
get_colum_names() (*in module test*), 4
get_id_for_ad_time() (*in module populate*), 2
get_id_for_ad_time() (*in module test*), 4
get_id_for_ads_desc() (*in module populate*), 3
get_id_for_ads_desc() (*in module test*), 4
get_id_for_submediums() (*in module populate*), 3
get_id_for_submediums() (*in module test*), 4
get_index_val() (*in module populate*), 3
get_index_val() (*in module test*), 4
get_min_max_date() (*in module populate*), 3
get_min_max_date() (*in module test*), 4

I

iter_over_inputs() (*in module populate*), 3
iter_over_inputs() (*in module test*), 4

M

module
    populate, 1
    test, 4
    tools, 4
    tools.conf, 4

P

populate
    module, 1

T

test
    module, 4
tools
    module, 4
tools.conf
    module, 4