
pygcat Documentation

Release 0.2.0

Rodolphe Quiédeville

November 24, 2015

1	Introduction	3
2	PygCatalog	5
3	Exceptions	9
4	Contributing	11
5	Indices and tables	13
	Python Module Index	15

Make access to postgresql schema informations easy, is the column **C** is present in an index, is the table **T** contains a primary key, PygCat can answers this.

Contents:

Introduction

PygCatalog simply access to schema informations, you can easily know if a column is present in an index or use of special type in a table.

The only one requirements is to have a psycopg2 connection open to a database, PygCatalog will explore all schemas

PygCatalog

Read information in Postgresql system catalog

exception `pygcat.ColumnDoesNotExist`

A column does not exists

Raised when a column is specifically requested as a parameter in a function

class `pygcat.PygCatalog` (*conn=None, default_schemas=['public']*)

Python library to read PostgreSQL system catalog

analyze (*table=None*)

Run an ANALYZE over the database or a table

biggest_table ()

Return the biggest table in term of total size

The size is compute all disk usage used by the table, it includes datas, indexes and TOAST data. Sizes are express in Bytes.

Example

```
>>> cat.biggest_table()
('foo', 163840L, 1000L)
```

biggest_tables (*max=1, **kwargs*)

Return the biggest table in term of total size

The size is compute all disk usage used by the table, it includes datas, indexes and TOAST data.

Example

```
>>> cat.biggest_table()
('foo', 163840L, 1000L)
```

get_indexes (*schema='public', **kwargs*)

Return all indexes in a schema

Return all indexes defined in the schemas, each indexex is associated with the table oid, it's own oid, the number of tuples present in it and the name of the columns.

Example

```
>>> cat.get_indexes()
{'foo_name_idx': {'table_oid': 121090,
                  'oid': 121093,
                  'columns': None,
                  'tuple': 1000L},
```

```
'foo_name_ratio_idx': {'table_oid': 121090,
                        'oid': 121094,
                        'columns': None,
                        'tuple': 1000L}
}
```

Returns dict that contains all indexes

Return type dict

get_operator_class (***kwargs*)

Return information on operator class

<http://www.postgresql.org/docs/current/static/catalog-pg-opclass.html>

get_table_columns (*table*, *schema='public'*)

Return all columns in a table

Return type list

get_tables (***kwargs*)

Return tables list

You may specify a single schema to look in by specifying the keyword argument *schema*

Example

```
>>> cat.get_tables(schema='public')
```

get_triggers (*tablename*, ***kwargs*)

Return information on triggers

<http://www.postgresql.org/docs/current/static/catalog-pg-trigger.html>

Example

```
>>> cat.get_triggers('foobar')
[{'name': 'car_insert_trigger', 'event': 'INSERT',
  'timing': 'BEFORE'},
 {'name': 'car_update_trigger', 'event': 'UPDATE',
  'timing': 'AFTER'}]
```

Returns all triggers on a table

Return type array

is_column_exists (*column_name*, *table_name*, *schema='public'*)

Check if a column exists in a table

Parameters

- **column_name** – the column's name to look for
- **table_name** – the table's name to look in

is_column_indexed (*column_name*, *table_name*, *schema='public'*)

Check if a column is indexed

Check if the column is present in at least one index.

Parameters

- Returns** The result of the addition

Return type boolean

Example

is_table_exists (*table_name*, *schema*='public')

Check if a table exists

Parameters `table_name` (*string*) – The table’s name to look for

Returns The result of the addition

Return type boolean

Example

```
pgversion()
    Run the version of PostgreSQL

reset_cache()
    Reset the cache

schemas()
    Return schemas
    Return the list of all schemas present in the database
```

Example

Return type list

set_default_schema (*schema*)
Define the default schema to work on

Parameters **schema** (*string*) – The schema's name to work on

Returns The result of the addition

Return type boolean

set_default_schemas (*schemas*)
Define as set of schemas to work on

Remove schemas set twice or more

table_tuples (*table*, ****kwargs**)
Return the table's number of tuples

exception `pygcat.TableDoesNotExist`

A table does not exists

Raised when a table is specificaly requested as a parameter in a function

Exceptions

class `pygcat.SchemaDoesNotExist`

A schema does not exist

Raised when a schema is specifically requested as a parameter in a function

class `pygcat.TableDoesNotExist`

A table does not exist

Raised when a table is specifically requested as a parameter in a function

class `pygcat.ColumnDoesNotExist`

A column does not exist

Raised when a column is specifically requested as a parameter in a function

Contributing

- [Source code](#)
- [Issues](#)

Indices and tables

- `genindex`
- `modindex`
- `search`

p

pygcat, [5](#)

A

analyze() (pygcat.PygCatalog method), 5

B

biggest_table() (pygcat.PygCatalog method), 5

biggest_tables() (pygcat.PygCatalog method), 5

C

ColumnDoesNotExist, 5

ColumnDoesNotExist (class in pygcat), 9

G

get_indexes() (pygcat.PygCatalog method), 5

get_operator_class() (pygcat.PygCatalog method), 6

get_table_columns() (pygcat.PygCatalog method), 6

get_tables() (pygcat.PygCatalog method), 6

get_triggers() (pygcat.PygCatalog method), 6

I

is_column_exists() (pygcat.PygCatalog method), 6

is_column_indexed() (pygcat.PygCatalog method), 6

is_table_exists() (pygcat.PygCatalog method), 7

P

pgversion() (pygcat.PygCatalog method), 7

pygcat (module), 5

PygCatalog (class in pygcat), 5

R

reset_cache() (pygcat.PygCatalog method), 7

S

SchemaDoesNotExist, 7

SchemaDoesNotExist (class in pygcat), 9

schemas() (pygcat.PygCatalog method), 7

set_default_schema() (pygcat.PygCatalog method), 7

set_default_schemas() (pygcat.PygCatalog method), 7

T

table_tuples() (pygcat.PygCatalog method), 7

TableDoesNotExist, 8

TableDoesNotExist (class in pygcat), 9