pysolrq Documentation

Release 1.0.2

Diwanshu Shekhar

Feb 03, 2019

Contents:

CHAPTER 1

pysolrq package

class pysolrq.solr.**SolrClient**(*host*, *version=4.7*)

__dict__ = dict_proxy({'__module__': 'pysolrq.solr', 'get_control': <function get_co

__init__(*host*, *version=4.7*)
Constructor for SolrClient class

Parameters

- **host** (*str*) – Solr host Example:http://example.company.com:8983/solr/
- **version** (*float*) – Current version of Solr host, default=4.7

__module__ = 'pysolrq.solr'

__weakref__
list of weak references to the object (if defined)

get_collection(*collection*, *max_rows=50000*)
Factory method to return SolrCollection object

Parameters

- **collection** (*str*) – name of Solr collection
- **max_rows** (*int*) – maximum rows to fetch, default=50,000

Returns

Return type *SolrCollection*

get_control(*collection*)
Factory method to return SolrControl object

Parameters **collection** (*str*) – name of Solr collection

Returns

Return type *SolrControl*

class pysolrq.solr.**SolrCollection**(*host*, *collection*, *max_rows=50000*)
SolrCollection class

Should not be instantiated directly. Use get_collection method of SolrClient object to get SolrCollection object

__init__(*host*, *collection*, *max_rows=50000*)
Constructor for SolrCollection class

Parameters

- **host** (*str*) – Solr host Example:http://example.company.com:8983/solr/
- **collection** (*str*) – name of Solr collection
- **max_rows** (*int*) – maximum rows to fetch

__module__ = 'pysolrq.solr'

__repr__(*)* <==> *repr(x)*

__str__(*)* <==> *str(x)*

facet_range(*query*, *field_params*)
Get facet results using Solr Facets

Parameters

- **query** (*str*) – Query string Example: `'field1':'val1' AND 'field2':'val2'`
- **field_params** (*dict*) – Example: `{field_1:[start, end, gap, include], field_2:[start, end, gap, include]}`
- **bins** (*int*) –

Returns

Return type dict

fetch(*query*, *fields=None*, *num_rows=None*)
Fetches all rows from returned results from your Solr collection

Parameters

- **query** (*str*) – Query string Example: `'field1':'val1' AND 'field2':'val2'`
- **fields** (*list of str*) – comma separated list of field names Example: `['field1', 'field3']`
- **num_rows** (*int*) – number of rows to fetch

Returns

- *list* – a list of dicts
- *None* – if self.num_found exceeds self.max_rows

pre_fetch(*query*, *fields*)
Fetches the first 10 rows from returned results from your Solr collection

Parameters

- **query** (*str*) – Query string Example: `'field1':'val1' AND 'field2':'val2'`
- **fields** (*list of str*) – comma separated list of field names Example: `['field1', 'field3']`

Returns

Return type None

stats(*query, fields, metrics=['min', 'max', 'sum', 'count', 'missing', 'sumOfSquaresmean', 'stddev', 'percentiles', 'distinctValues', 'countDistinct', 'cardinality'], percentiles='25,50,75'*)

Gets basic statistics from Solr

Parameters

- **query** (*str*) –

  Query string:: Example: `'field1':'val1' AND 'field2':'val2'`

- **fields** (*list of str*) –

  comma separated list of field names:: Example: `['field1', 'field3']`

- **metrics** (*list of str*) – list of available metrics are: 'min', 'max', 'sum', 'count', 'missing', 'sumOfSquares', 'mean', 'stddev', 'percentiles', 'distinctValues', 'countDistinct', 'cardinality'

- **percentiles** (*str*) – A string where different percentile values are separated by commas Example: `"25,50,75"` Note: Uses t-digest approximation algorithm

Returns A dictionary with metrics as keys

Return type dict

class pysolrq.solr.**SolrControl**(*host*, *collection*)

SolrControl class can be used to make collections and perform indexing of your data.

The data can be in a delimited file such as CSV or a Solr acceptable xml format such as:

```
<add>
    <doc>
        <field name="id">001</field>
        <field name="food">milk</field>
        <field name="talk">meow</field>
    </doc>
    <doc>
        <field name="id">002</field>
        <field name="food">bone</field>
        <field name="talk">bark</field>
    </doc>
</add>
```

__init__(*host*, *collection*)

Constructor for SorControl class

Parameters

- **host** (*str*) – Solr host Example:http://example.company.com:8983/solr/
- **collection** (*str*) – name of Solr collection

__module__ = 'pysolrq.solr'

_clean(*values*)

Cleans the data in `values`

Parameters **values** (*list*) – list of some data

Returns A list of values in `values` with leading and trailing whitespaces removed

Return type list

_csv_iter (*filename*, *delimiter=', '*)
Returns a generator of the read delimited file

Parameters

- **filename** (*str*) – A delimited file
- **delimiter** (*str*) – Example: `","`

Yields *list* – The next list of values read in a row in the given delimited file

_data_iter (*file_path*, *delimiter=None*, *fields=None*, *unique_id=True*, *keep_row=False*)
Returns a generator of the read delimited file

Parameters

- **file_path** (*str*) – A delimited file
- **delimiter** (*str*) – Example: `","`
- **fields** (*tuple of str.*) – A list of field names to be used for indexing Example: `('field1', 'field3')`

Yields *str* – The next str is an xml formatted str with values read from a row in the `file_path` file. Example: if a delimited file contains a row as:

```
"cat", "milk", "meow"
```

this method will yield:

```
<add>
    <doc>
        <field name="id">3d144141'</field>
        <field name="food">Hi</field>
        <field name="talk">Hello</field>
    </doc>
</add>
```

assuming the given fields are `["food", "talk"]`

_get_data (*values*, *fields*, *unique_id=True*)
Given the values and fields, returns an str in Solr acceptable xml format

Parameters

- **values** (*list*) – list of some data
- **fields** (*list of str*) –

  A list of field names to be used for indexing:: Example: `['field1', 'field3']`

Returns

Return type str

_get_doc (*d*, *unique_id=True*)
Given a dictionary of fields and values, returns an str to be used by `_get_data` method

_post_to_collection (*data*)
Given the `data` in Solr acceptable xml format posts the data to the Solr Collection

_transform (*line*, *fields*)

_transform_partition (*partition*, *fields*)

_xmltostr(*file_path*)
Reads a solrxml file and converts it to a string

Parameters **file_path** (*str*) – An xml file

Returns

Return type str

make_collection(*num_shards*)
Makes a new collection This assumes that the user has already uploaded the collection's configuration to zookeeper

Parameters **num_shards** (*int*) – number of shards for the collection

Returns

Return type None

start_index(*file_path_or_spark_df*, *file_format='solrxml'*, *delimiter=None*, *fields=None*, *unique_id=True*, *batch_size=1*, *keep_row=False*, *cleaner_func=None*)
Indexes data to the collection

Parameters

- **file_path** (*str*) – Points to a file with data to be indexed
- **file_format** (*str*) – Available choices are 'solrxml' or 'csv'.
- **delimiter** (*str*) – Required when file_format='csv'. Example: `","`
- **fields** (*tuple of str.*) – A list of field names to be used for indexing Example: `('field1', 'field2')`
- **unique_id** (*bool*) – If True, autogenerates a field name id and a unique uuid value to the doc If False, modify the Solr config so that id is not a unique key

Returns

Return type None

Python Module Index

p

Index

Symbols

F

G

M

P

S