pyDrill-dsl Documentation

Release 0.0.2

pyDrill-dsl

Mar 15, 2017

Contents

1 pyDrill-dsl **3**
- 1.1 Features . . . 3
- 1.2 Sample usage . . . 3

2 Installation **5**

3 Usage **7**

4 Contributing **9**
- 4.1 Types of Contributions . . . 9
- 4.2 Get Started! . . . 10
- 4.3 Pull Request Guidelines . . . 11
- 4.4 Tips . . . 11

5 Credits **13**
- 5.1 Development Lead . . . 13
- 5.2 Contributors . . . 13

6 History **15**
- 6.1 0.0.1 (2016-02-18) . . . 15

7 Indices and tables **17**

Contents:

CHAPTER 1

pyDrill-dsl

Pythonic DSL for Apache Drill.

Schema-free SQL Query Engine for Hadoop, NoSQL and Cloud Storage

- Free software: MIT license
- Documentation: https://pydrill_dsl.readthedocs.org.

Features

- Uses Peewee syntax
- Support for all storage plugins
- Support for drivers PyODBC and pyDrill

Sample usage

```
from pydrill_dsl.resource import Resource

class Employee(Resource):
    first_name = Field()
    salary = Field()
    position_id = Field()
    department_id = Field()

    class Meta:
        storage_plugin = 'cp'
        path = 'employee.json'


Employee.select().filter(salary__gte=17000)
```

```
Employee.select().paginate(page=1, paginate_by=5)


salary_gte_17K = (Employee.salary >= 17000)
salary_lte_25K = (Employee.salary <= 25000)
Employee.select().where(salary_gte_17K & salary_lte_25K)

Employee.select(
    fn.Min(Employee.salary).alias('salary_min'),
    fn.Max(Employee.salary).alias('salary_max')
).scalar(as_tuple=True)

# creation of resource can be done without creation of class:
employee = Resource(storage_plugin='cp', path='employee.json',
                    fields=('first_name', 'salary', 'position_id', 'department_id'))
```

CHAPTER 2

Installation

At the command line:

```
$ easy_install pydrill_dsl
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv pydrill_dsl
$ pip install pydrill_dsl
```

CHAPTER 3

Usage

To use pyDrill-dsl in a project:

```
import pydrill_dsl
```

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

Types of Contributions

Report Bugs

Report bugs at https://github.com/PythonicNinja/pydrill_dsl/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

Write Documentation

pyDrill-dsl could always use more documentation, whether as part of the official pyDrill-dsl docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/PythonicNinja/pydrill_dsl/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

Get Started!

Ready to contribute? Here's how to set up *pydrill_dsl* for local development.

1. Fork the *pydrill_dsl* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/pydrill_dsl.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv pydrill_dsl
$ cd pydrill_dsl/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 pydrill_dsl tests
$ python setup.py test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/PythonicNinja/pydrill_dsl/pull_requests and make sure that the tests pass for all supported Python versions.

Tips

To run a subset of tests:

```
$ python -m unittest tests.test_pydrill_dsl
```

CHAPTER 5

Credits

Development Lead

- pyDrill-dsl <mail@pythonic.ninja>

Contributors

None yet. Why not be the first?

CHAPTER 6

History

0.0.2 (2016-05-19)

- First release on PyPI.
- Uses Peewee syntax
- Support for all storage plugins
- Support for drivers PyODBC and pyDrill
- Builds are tested by docker container with Apache Drill running

0.0.1 (2016-02-18)

- Initial release on github.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search