datakit-data plugin Documentation

Release 0.2.0

Serdar Tumgoren

Jan 19, 2018

Contents

Contents:

CHAPTER 1

Quickstart

Install:

```
$ sudo pip install datakit-data
$ aws configure
```

Initialize project for use with S3:

```
$ cd /path/to/my-project
$ datakit data init
```

Edit *config/datakit-data.json*:

```
{
  "aws_user_profile": "default",
  "s3_bucket": "",
  "s3_path": "my-project"
}
```

Drop data files in project data directory:

```
$ touch data/foo.csv
```

Push/pull data files between local machine and S3:

```
$ datakit data push
$ datakit data pull
```

Note: Don't forget to check out *Usage* for more details and advanced configuration and usage options.

CHAPTER 2

Overview

The datakit-data package is a plugin for the datakit command-line tool. The plugin helps archive and share data assets on Amazon S3 by providing a few simple commands for pushing and pulling data.

At the Associated Press, members of the Data Team use the plugin to simplify collaboration on data projects.

2.1 Highlights

- Human-friendly commands to push files to S3 and to pull files down
- Customizable default configs to help rapidly bootstrap new projects
- Access to advanced features of underlying AWS command-line utility for power users

2.2 Plugin description

At root, *datakit-data* lets you link a local directory – called *data/* by convention – with a remote location in an S3 bucket, and to transfer data between those locations[1].

It's common to link numerous local *data/* directories – typically one per project – with project-specific paths on a remote S3 bucket.

Here's a preview of how the tool is used on the command line once a project has been integrated with S3 (see *Initialize*):

```
$ cd /path/to/data-project

# Stash data on S3
$ datakit data push

# Fetch data from S3
$ datakit data pull
```

[1] Buckets must be created prior to pushing data to S3. The plugin does not automatically create new buckets.

Please see *Usage* for more details on integrating a project with an S3 data store.

2.3 Other projects

The plugin is designed to be fairly generic to enable others to use and contribute to the code base.

However, it may be worth checking out the below projects before settling on a workflow, in case one of these better matches your needs:

- dat - A "package manager" for data.
- others...

CHAPTER 3

Installation

3.1 Install the plugin

In order to use this plugin with a system-wide install of datakit:

```
$ sudo pip install datakit-data
```

3.2 Configure AWS

After installing datakit-data, you must configure secret keys for reading from and writing to an AWS S3 bucket.

The easiest way to do this is to run the aws configure command and enter the appropriate information when prompted:

```
$ aws configure
```

Note: The above command creates the *~/.aws* directory and related configuration files, which can be updated manually if needed.

CHAPTER 4

Usage

This plugin is intended to help data analysts more easily share and archive project data, using AWS S3 as a centralized data store.

4.1 Setup

Integrating a project with S3 involves a few steps:

- Initialize the project
- Check S3 to ensure a new project won't overwrite a pre-existing project on S3[1]
- Update AWS configurations, as needed
- Exclude the project's *data/* directory from version control (see *Version control and data*).

4.1.1 Initialize

To initialize:

```
$ cd /path/to/my-project
$ datakit data init
```

The *data init* command creates:

- *data/* - a directory where data files should be placed. This directory will be synced to the S3 bucket and path specified in the project configuration file (see below).
- *config/datakit-data.json* - a JSON file with the below settings:

[1] datakit-data does not currently guard against overwrites of pre-existing projects of the same name.

```
{
  "aws_user_profile": "default",
  "s3_bucket": "",
  "s3_path": "my-project"
}
```

Note: datakit-data does not currently provide safeguards against accidental overwrites of previously created projects (on S3) with an identical name. Users should always double-check the target S3 bucket to ensure that a project path has not already been used.

4.1.2 Configure

Project-level settings for S3 integration must be updated before data can be pushed to S3.

These configurations can be found in *config/datakit-data.json*:

aws_user_profile The user profile configured in *~/.aws/credentials*. The *default* profile is assumed by *datakit-data*, but this value can be modified if you have multiple profiles.

s3_bucket Name of S3 bucket where project data should be stored. By default this is an empty string.

s3_path The S3 bucket path to which the local *data/* directory should be mapped. By default, *datakit-data* maps the local *data/* directory to a folder named after the project's root folder.

4.2 Default configurations

As a convenience, *datakit-data* provides the ability to pre-configure default settings for AWS integration. This feature helps speed up S3 integration for new projects.

Default values for the *aws_user_profile* and *s3_bucket* settings mentioned in *Configure* can be placed in **~/.datakit/plugins/datakit-data/config.json**. These configurations will then be applied to all projects when *datakit data init* is run.

4.2.1 Example

Below is an example showing pre-configured values for the S3 bucket name and an alternative aws user profile:

```
# ~/.datakit/plugins/datakit-data/config.json
{
  "aws_user_profile": "other_profile",
  "s3_bucket": "my-data-projects-bucket"
}
```

4.2.2 Custom S3 paths

datakit-data provides two additional settings, only available at the global config level, to help customize the generation of the S3 path across projects.

These settings are only applied during S3 initialization. They can be overriden manually at any point by editing *config/datakit-data.json* for a given project.

s3_path_prefix one or more directory levels to be **prepended** to a project config's S3 path

s3_path_suffix one or more directory levels to be **appended** to a project config's S3 path

The prefix/suffix settings are useful when project data must be stored somewhere other than a project directory at the root of an S3 bucket.

For example, to store data in an S3 bucket at the following path:

```
projects/2017/my-project
```

..you would set **s3_path_prefix** to *projects/2017*. This path would then be prepended to the project's name in the *s3_path* configuration whenever a new project is initialized.

Similarly, you can segregate data assets inside of a project directory on S3 by using the **s3_path_suffix**. For example, to store data at the below path:

```
my-project/data
```

. . . you would set **s3_path_suffix** to *data/*.

And of course, you can use both of these settings in tandem:

```
projects/2017/my-project/data
```

4.3 Data push/pull

Note: The below commands must be run from a directory initialized and configured for use with S3 (see *Initialize* for details).

Pushing and pulling data between your local machine and the S3 data store requires two commands:

```
$ datakit data push
$ datakit data pull
```

The above commands provide a human-friendly interface to the AWS S3 sync commmand line utility.

The sync utility writes all files in a project's local *data/* directory (and its subdirectories) to the S3 bucket and path specified in *config/datakit-data.json*, or vice versa.

By default, this command does not delete previously written files in a target location if they have been removed in the source location.

This functionality is available, however, via the *–delete* flag of the underlying AWS S3 sync utility. *datakit-data* provides access to the *–delete* flag and a limited set of other options provided by the *sync* command (see *Extra flags*).

4.3.1 Extra flags

While *datakit-data* is intended to simplify and standardize working with S3 as a data store, it also exposes a subset of more advanced options for the underlying AWS S3 sync utility.

Users can pass any **boolean** flag supported by *S3 sync* to the plugin's *push* or *pull* commands.

Boolean flags are those that do not accept values (i.e. simply calling them toggles a behavior on or off).

The flags must be passed to *datakit* as additional paramaters **without leading dashes**[2]

For example, to delete files on S3 that are *not* present locally:

```
$ datakit data push delete
```

To view which files will be affected before pushing data to S3:

```
$ datakit data push dryrun

or

$ datakit data push delete dryrun
```

Please refer to the AWS S3 sync documentation for details on other boolean flags.

4.4 Version control and data

This plugin expects data files associated with a project to live in a *data/* directory at the root of a project folder. This is typically the root of a code repository.

While code to acquire, clean and analyze data should be placed under version control, the *data/* directory itself *should be excluded from version control.*

Note: Version control systems have different mechanisms to prevent files from being "tracked". Git users, for instance, should add the *data/* directory to a project's .gitignore file.

---

[2] Leading slashes must be dropped to enable datakit to differentiate between its own flags and those intended for pass-through to the underlying AWS S3 sync utility.

CHAPTER 5

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:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/associatedpress/datakit-data/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.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

5.1.4 Write Documentation

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

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/associatedpress/datakit-data/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 :)

5.2 Get Started!

Ready to contribute? Here's how to set up *datakit-data* for local development.

1. Fork the *datakit-data* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/datakit-data.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 datakit-data
$ cd datakit-data/
$ 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 datakit_data tests
$ 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.

5.3 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 version 3.3 or higher. Check https://travis-ci.org/associatedpress/datakit-data/pull_requests and make sure that the tests pass for all supported Python versions.

CHAPTER 6

Contributors

Development Lead

- Serdar Tumgoren <stumgoren@ap.org>

Contributors

- Michelle Minkoff <mminkoff@ap.org>

CHAPTER 7

Credits

This package was created with Cookiecutter and the associatedpress/cookiecutter-datakit-plugin project template (a modified version of the most excellent audreyr/cookiecutter-pypackage).

CHAPTER 8

History

8.1 0.2.0 (2018-01-17)

- Switch from colon- to space-separated plugin commands

8.2 0.1.0 (2016-03-16)

- Project initialization command
- Push/Pull commands
- Custom syntax for pass-through of boolean flags to aws sync utility
- Project-level configuration inheritance from global configs
- Support for customizing S3 path with prefix/suffix from global configs

CHAPTER 9

Indices and tables

- genindex
- modindex
- search