
AthenaCLI Documentation

Zhaolong Zhu

Jan 11, 2022

Contents

1	Quick Start	3
1.1	Install	3
1.2	Config	3
1.3	Create a table	4
1.4	Run a query	4
1.5	REPL	4
1.6	Table of Contents	4

AthenaCLI is a command line interface (CLI) for [Athena](#) service that can do auto-completion and syntax highlighting, and is a proud member of the dbcli community.

- Source: <https://github.com/dbcli/athenacli>

CHAPTER 1

Quick Start

1.1 Install

```
$ pip install athenacli
```

You can refer to [Install](#) page for more options.

1.2 Config

A config file is automatically created at `~/.athenacli/athenaclicrc` at first launch (run *athenacli*). See the file itself for a description of all available options.

Below 4 variables are required.

```
# If you are a user of aws cli, you might want to use some configurations of aws cli,
# please refer to https://athenacli.readthedocs.io/en/latest/awsconfig.html for more
→ information.
aws_access_key_id = ''
aws_secret_access_key = ''
region = '' # e.g us-west-2, us-east-1

# Amazon S3 staging directory where query results are stored.
# NOTE: S3 should in the same region as specified above.
# The format is 's3://<your s3 directory path>'
s3_staging_dir = ''

# Name of athena workgroup that you want to use
work_group = '' # e.g. primary
```

1.3 Create a table

```
$ athenacli -e examples/create_table.sql
```

You can find *examples/create_table.sql* [here](#).

1.4 Run a query

```
$ athenacli -e 'select elb_name, request_ip from elb_logs LIMIT 10'
```

1.5 REPL

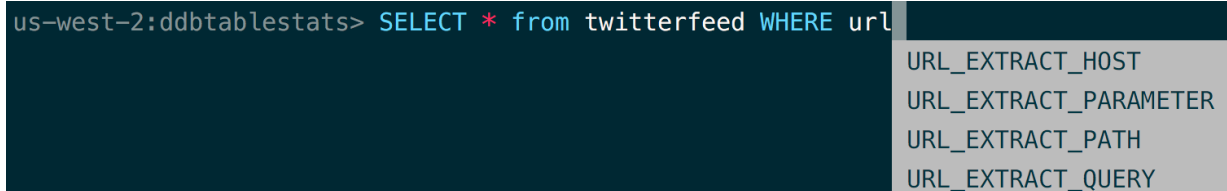
```
$ athenacli [<database_name>]
```

1.6 Table of Contents

1.6.1 Features

Auto completion

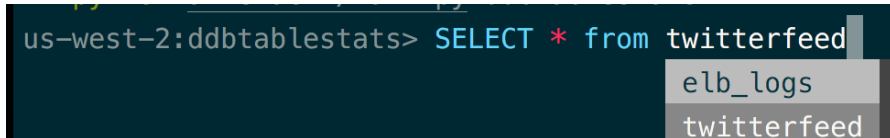
Simple completions such as keywords and sql-functions.

A screenshot of the AthenaCLI interface. The prompt is 'us-west-2:ddbtablestats>'. The user has entered 'SELECT * from twitterfeed WHERE url'. A dropdown menu is visible on the right, listing four options: 'URL_EXTRACT_HOST', 'URL_EXTRACT_PARAMETER', 'URL_EXTRACT_PATH', and 'URL_EXTRACT_QUERY'.

```
us-west-2:ddbtablestats> SELECT * from twitterfeed WHERE url
URL_EXTRACT_HOST
URL_EXTRACT_PARAMETER
URL_EXTRACT_PATH
URL_EXTRACT_QUERY
```

Smart completion

Smart completion will suggest context-sensitive completion.

A screenshot of the AthenaCLI interface. The prompt is 'us-west-2:ddbtablestats>'. The user has entered 'SELECT * from twitterfeed'. A dropdown menu is visible on the right, listing two options: 'elb_logs' and 'twitterfeed'.

```
us-west-2:ddbtablestats> SELECT * from twitterfeed
elb_logs
twitterfeed
```


Alias support

Column completions will work even when table names are aliased.

```
us-west-2:ddbtablestats> SELECT t. FROM twitterfeed t;  
                                "day"  
                                "year"  
                                *  
                                approx_post_time  
                                hour  
                                month  
                                table_name  
                                tweet_id  
                                user_id
```

Syntax highlighting

Syntax highlighting for sql.

```
us-west-2:ddbtablestats> SELECT * FROM twitterfeed WHERE "year" = '2018';
```

Multiline queries

Support for multiline queries.

```
us-west-2:ddbtablestats> SELECT *  
-> FROM twitterfeed  
-> WHERE "year" = '2018' LIMIT 1;
```

Pager

Output of an sql command is automatically piped through less command.

```
+-----+-----+
| user_id | tweet_id |
+-----+-----+
| Deborah | 81897555816210836 |
| Mary    | 10126258856962902 |
| Brent   | 86711732862762545 |
| Michael | 90951751963667749 |
| Patricia | 82888881169217478 |
| Marjorie | 78311710214870723 |
: 
```

Favorite queries

Save a query using fs alias query and execute it with f alias whenever you need.

```
us-west-2:ddbtablestats> \fs q1 SELECT user_id, tweet_id from twitterfeed LIMIT 2
Saved.
Time: 0.001s
us-west-2:ddbtablestats> \f q1
> SELECT user_id, tweet_id from twitterfeed LIMIT 2
+-----+-----+
| user_id | tweet_id |
+-----+-----+
| Deborah | 81897555816210836 |
| Mary    | 10126258856962902 |
+-----+-----+
Time: 3.094s
```

Various table format

Support various table format, e.g. ascii, csv, html etc.

1.6.2 Install

Pip

If you already know how to install python packages, then you can do:

```
$ pip install athenacli
```

You might need sudo, or you can install it in a virtualenv.

```

us-west-2:ddbtalestats> \T csv
Changed table format to csv
Time: 0.000s
us-west-2:ddbtalestats> SELECT user_id, tweet_id from twitterfeed LIMIT 2;
user_id,tweet_id
Deborah,81897555816210836
Mary,10126258856962902
2 rows in set
Time: 4.506s

```

Docker

If you already know how to use docker, then you can do:

```
$ docker run --rm -ti -v $(pwd) :/home/athena zz10/athenacli athenacli
```

Note: we map the home directory (*/home/athena*) of docker container to current directory, *athenacli* will create a config file in it (*.athenacli/athenaclicrc*), you might need to change some variables (please refer to *quick start* section of *AthenaCLI* page).

MacOS

For MacOS users, you can also use Homebrew to install it:

```
$ brew install athenacli
```

1.6.3 AWS Configs

AthenaCLI tries to reuse the AWS credentials and configurations configured by [AWS CLI](#).

Precedence

The AthenaCLI looks for credentials and configuration settings in the following order:

1. **Command line options** – `aws-access-key-id`, `aws-secret-access-key`, `region`, `s3-staging-dir`, `work-group` can be specified as command options to override default settings.
2. **AthenaCLI config file** – typically located at `~/.athenacli/athenaclicrc` on Linux, macOS, or Unix. This file can contain multiple named profiles in addition to a default profile. Just adds `-profile [PROFILE_NAME]` at the end of `athenacli` command to use those configurations.
3. **Environment variables** – `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, `AWS_DEFAULT_REGION`, `AWS_ATHENA_S3_STAGING_DIR`, `AWS_ATHENA_WORK_GROUP`
4. **AWS credentials file** – located at `~/.aws/credentials` on Linux, macOS, or Unix. This file can contain multiple named profiles in addition to a default profile. Please refer to *AWS CLI* for more information.
5. **AWS CLI config file** – typically located at `~/.aws/config` on Linux, macOS, or Unix. This file can contain multiple named profiles in addition to a default profile. Please refer to *AWS CLI* for more information.

- **Note:** Whether or not a particular value will be used from a given option above depends on the truthiness of the values. e.g. if the 'aws_access_key_id' field is present in the AthenaCLI config file, but its value is empty, it will not be considered (since the truthiness of an empty string is False) and the program will try to resolve to the next available option.

Available configs

Some variables are not available in all the config files, below table lists the config files in which you can set a variable.

Variable	Environment Variable	Available files
aws_access_key_id	AWS_ACCESS_KEY_ID	<ul style="list-style-type: none"> • AthenaCLI config • AWS credentials • AWS CLI config
aws_secret_access_key	AWS_SECRET_ACCESS_KEY	<ul style="list-style-type: none"> • AthenaCLI config • AWS credentials • AWS CLI config
token	AWS_SESSION_TOKEN	N/A
region	AWS_DEFAULT_REGION	<ul style="list-style-type: none"> • AthenaCLI config • AWS CLI config
s3_staging_dir	AWS_ATHENA_S3_STAGING_DIR	<ul style="list-style-type: none"> • AthenaCLI config
work_group	AWS_ATHENA_WORK_GROUP	<ul style="list-style-type: none"> • AthenaCLI config

1.6.4 Usages

Options

```
$ athenacli --help
Usage: athenacli [OPTIONS] [DATABASE]

A Athena terminal client with auto-completion and syntax highlighting.

Examples:
- athenacli
- athenacli my_database

Options:
-e, --execute TEXT      Execute a command (or a file) and quit.
-r, --region TEXT      AWS region.
--aws-access-key-id TEXT AWS access key id.
--aws-secret-access-key TEXT AWS secretaccess key.
--s3-staging-dir TEXT  Amazon S3 staging directory where query
                        results are stored.
--work_group TEXT      Amazon Athena workgroup in which query is run,
                        default is primary
```

(continues on next page)

(continued from previous page)

<code>--athenaclic FILE</code>	Location of athenaclic file.
<code>--profile TEXT</code>	AWS profile
<code>--table-format TEXT</code>	Table format used with <code>-e</code> option.
<code>--help</code>	Show this message and exit.

Connect to a database

Connect a specific database with AWS credentials, region name and S3 staging directory or work group. AWS credentials, region name and S3 staging directory are optional. You can set those variables in *athenaclic* config file, and then run below command.

```
$ athenaclic ddbtablestats
```

Exit athenaclic

Press `ctrl+d` or type *quit* or *exit*.

Special Commands

Save 'SELECT user_id, tweet_id from twitterfeed LIMIT 2' as a favorite query called 'q1':

```
> \fs q1 SELECT user_id, tweet_id from twitterfeed LIMIT 2
```

Run the named query:

```
> \f q1
```

Execute a command (or a file)

Execute a command and quit:

```
$ athenaclic -e 'show databases'
```

Execute a file and quit:

```
$ athenaclic -e examples/create_table.sql
```

1.6.5 Development Guide

This is a guide for developers who would like to contribute to this project.

Fork this project

Firstly, You need to fork this project and clone your fork into your computer.

```
$ git clone <url-for-your-fork>
```

Local setup

The installation instructions in the README file are intended for users of athenacli. If you're developing athenacli, you'll need to install it in a slightly different way so you can see the effects of your changes right away without having to go through the install cycle everytime you change the code.

It is highly recommended to use virtualenv for development. If you don't know what a virtualenv is, [this guide](#) will help you get started.

Create a virtualenv (let's call it athenacli-dev):

```
$ virtualenv athenacli-dev
```

Activate it:

```
$ source ./athenacli-dev/bin/activate
```

Once the virtualenv is activated, cd into the local clone of athenacli folder and install athenacli using pip as follows:

```
$ pip install -e .
```

This will install the necessary dependencies as well as install athenacli from the working folder into a virtualenv. Athenacli is installed in an editable way, so any changes made to the code is immediately available in the installed version of athenacli. This makes it easy to change something in the code, launch athenacli and check the effects of your change.

Running the tests

Currently we don't have enough tests for athenacli, because we haven't found an easy way to test AWS Athena locally, we have an [issue](#) track this problem. But we do have some unit tests for other parts, below are the steps to run them.

First, install the requirements for testing:

```
$ pip install -r requirements-dev.txt
```

After that, tests can be run with:

```
$ pytest
```

Create a pull request

After making the changes and creating the commits in your local machine. Then push those changes to your fork. Then click on the pull request icon on github and create a new pull request. Add a description about the change and send it along. I promise to review the pull request in a reasonable window of time and get back to you.

In order to keep your fork up to date with any changes from mainline, add a new git remote to your local copy called 'upstream' and point it to the main athenacli repo.

```
$ git remote add upstream https://github.com/dbcli/athenacli.git
```

Once the 'upstream' end point is added you can then periodically do a [git rebase](#) to update your local copy.

1.6.6 FAQs

How can I get support for athenacli?

There is [Gitter chat](#). We also track our bugs and feature requests in Github Issues for this project.

I found a bug, what do I do?

I'm sorry you encountered a bug. Please file a bug on Github Issues for this project. I'll fix it asap.

I have a feature request, what do I do?

Sweet! Open a new item on Github Issues. Alternatively, you can take a stab at implementing the new feature yourself. If you'd like some guidance, I'm just an email away. Don't hesitate to contact me.