Datarade

Release 0.3.0

Mike Alfare

Mar 25, 2021

CONTENTS:

The source repository is located here.

CHAPTER

ONE

INDICES AND TABLES

- genindex
- modindex
- search

CHAPTER

TWO

API

These services allow a user to interact with datasets stored in a git-compliant source control repository. This layer should be treated as the interface to this library. In other words, breaking changes may be introduced at lower levels, but this layer should remain relatively stable as the library matures.

datarade.services.**get_dataset_catalog**(*repository*, *organization*, *platform*, *project=None*, *branch='master'*, *username=None*, *password=None*)

A factory function that provides a DatasetCatalog instance

The structure of the files in the dataset catalog should look like this:

```
repository
|
|--- catalog
    |
    |--- my_dataset
        |
        |--- config.yaml
        |--- definition.sql
    |--- my_other_dataset
        |
        |--- config.yaml
        |--- definition.sql
```

The repository can be hosted on Git Hub or on Azure Repos. Multiple branches can be used for managing related dataset catalogs. For instance, you may want to maintain a uat branch and a production branch for managing environments. Or you may want one repo for all of your catalogs, but you want to provide some organization to your datasets.

Parameters

- **repository** (str) – the name of the repository
- **organization** (str) – the name of the organization (or user for GitHub) that owns the repository
- **platform** (str) – that platform that hosts the repo [‘github’, ‘azure-devops’]
- **project** (Optional[str]) – the name of the project that contains the repository, only used for Azure Repos
- **branch** (Optional[str]) – the branch to use in the repository, defaults to ‘master’
- **username** (Optional[str]) – the username with read access to the repository, only used for Azure Repos
- **password** (Optional[str]) – the password with read access to the repository, only used for Azure Repos, can also be the one-time git credentials password that bypasses MFA

Returns: a DatasetCatalog instance

Return type *DatasetCatalog*

datarade.services.**get_dataset_container**(*driver*, *database_name*, *host*, *port=None*, *schema_name=None*, *username=None*, *password=None*)

A factory function that provides a DatasetContainer instance

Parameters

- **driver** (str) – the type of database, currently only 'mssql' is supported
- **database_name** (str) – name of the database
- **host** (str) – the name of the server, including the instance
- **port** (Optional[int]) – the port that the database is listening to on the server
- **schema_name** (Optional[str]) – the name of the schema
- **username** (Optional[str]) – a user with create table and insert permissions on the schema
- **password** (Optional[str]) – the password for the user

Returns: a DatasetContainer instance

Return type *DatasetContainer*

datarade.services.**get_dataset**(*dataset_catalog*, *dataset_name*)

Returns a datarade Dataset object using the identified configuration in the dataset catalog

It collects all of the required files from the dataset catalog repository, puts the contents in a configuration dictionary, passes that dictionary up to the abstract repository for validation, and returns the resulting Dataset instance.

Parameters

- **dataset_catalog** (*DatasetCatalog*) – dataset catalog that contains the dataset
- **dataset_name** (str) – the name of the dataset, which is also the name of the directory containing the files in the repository

Returns: a Dataset object

Return type *Dataset*

datarade.services.**write_dataset**(*dataset*, *dataset_container*, *username=None*, *password=None*)

Writes the supplied dataset to the dataset container

The supplied dataset is exported using the provided credentials. If no credentials are supplied, Windows AD is used for the account running this script. Data is written out to ~/bcp/data and logs are written out to ~/bcp/logs. Data is then imported into the supplied dataset container using credentials in that dataset container. Again, if no credentials were supplied, Windows AD is used. Error records are written out to ~/bcp/data and logs are written out to ~/bcp/logs. On a successful write, the data file is deleted to avoid leaving copies of data behind on the application machine.

Parameters

- **dataset** (*Dataset*) – the dataset to be written
- **dataset_container** (*DatasetContainer*) – the database to store the dataset in
- **username** (Optional[str]) – a user with select/execute permissions on the source database objects

- **password** (Optional[str]) – the password for the user

CHAPTER

THREE

DEVELOPER DOCS

If you are using datarade as a library, you likely can stop after the API section above. But if you're interested in how the library works, or want o contribute to it, please read further.

3.1 Models

This module contains all models for datarade.

exception datarade.models.**DatasetCatalogNotSupportedException**
Occurs when an invalid platform is supplied to a DatasetCatalog instance.

exception datarade.models.**DriverNotSupportedException**
Occurs when an invalid driver is supplied to a Database instance.

class datarade.models.**Field**(*name*, *type*, *description=None*)
Represents a column in a dataset

Parameters

- **name** (str) – name of the field
- **type** (str) – field type, one of: [Boolean, Date, DateTime, Time, Float, Integer, Numeric, String, Text]
- **description** (Optional[str]) – non-functional, short description of the field, can include notes about what the field is or how it's populated

property sqlalchemy_column
Converts a datarade Field object into a sqlalchemy Column object

Returns: a sqlalchemy Column object

Return type Column

class datarade.models.**Database**(*driver*, *database_name*, *host*, *port=None*, *schema_name=None*)
Represents a database, either as a source for a Dataset, or as a target in a DatasetContainer

Parameters

- **driver** (str) – the type of database, currently only 'mssql' is supported
- **database_name** (str) – the name of the database
- **host** (str) – the name of the server, including the instance
- **port** (Optional[int]) – the port that the database is listening to on the server
- **schema_name** (Optional[str]) – the name of the schema

sqlalchemy_metadata(*username=None*, *password=None*)
Takes credentials and returns a sqlalchemy MetaData object for this database

Parameters

- **username** (Optional[str]) – the username for the database
- **password** (Optional[str]) – the password for the database

Returns: a sqlalchemy MetaData object

Return type MetaData

bcp(*username=None*, *password=None*)
Takes credentials and returns a BCP object for this database

Parameters

- **username** (Optional[str]) – the username for the database
- **password** (Optional[str]) – the password for the database

Returns: a BCP object

Return type BCP

full_table_name(*table_name*)
A utility method that is needed for MS SQL Server databases which have schemas

Parameters **table_name** (str) – the one part name of the table

Returns: the three part name of the table, if the schema is present

Return type str

property _sqlalchemy_driver_name
Selects the sqlalchemy package to use given the database driver

Returns: the sqlalchemy driver in '<database driver>+<sqlalchemy package>' format

Return type str

property _odbc_driver_name
Finds the appropriate ODBC driver on the machine given the database driver

Returns: the latest SQL Server Native Client for MS SQL Server databases

Return type str

class datarade.models.**User**(*username*)
Represents the user that should be used to access the data.

This should not store the password for obvious reasons, but can be used in conjunction with the password that is passed to the Database object. This makes it so that the client application that's consuming this dataset only needs to know the password for the account, not the account or where the account needs to be setup. It effectively turns the password into a token. This currently supports database users (e.g. a SQL Server account). To connect as an AD account, run your client application as that account and don't store the user in the dataset in your dataset catalog. For backwards compatibility, this is not a necessary attribute on a Dataset.

Parameters **username** (str) – the username, possibly with a domain (e.g. 'username', 'DOMAIN/username')

class datarade.models.**Dataset**(*name*, *definition*, *fields*, *description=None*, *database=None*, *user=None*)
Represents a dataset as metadata

Parameters

- **name** (str) – an identifier for the dataset that is unique within the DatasetCatalog
- **definition** (str) – the sql defining the dataset
- **fields** (List[*Field*]) – a list of Field objects in the dataset
- **description** (Optional[str]) – non-functional, short description of the dataset, can include notes about what the dataset is or how it's populated
- **database** (Optional[*Database*]) – a Database object that contains the data for the dataset
- **user** (Optional[*User*]) – a User object that can be used to connect to the database to access the data

class datarade.models.**DatasetCatalog**(*repository*, *organization*, *platform*, *project=None*, *branch='master'*, *username=None*, *password=None*)

Represents a git repo that hosts datasets in a predetermined structure

This can be thought of as a place to host datasets for data pipelines. But it can also be thought of as a place to advertise datasets to a broad audience since it only contains metadata and not the underlying data.

Parameters

- **repository** (str) – the name of the repository
- **organization** (str) – the name of the organization (or user for GitHub) that owns the repository
- **platform** (str) – that platform that hosts the repo ['github', 'azure-devops']
- **project** (Optional[str]) – the name of the project that contains the repository, only used for Azure Repos
- **branch** (str) – the branch to use in the repository
- **username** (Optional[str]) – the username with read access to the repository, only used for Azure Repos
- **password** (Optional[str]) – the password with read access to the repository, only used for Azure Repos, can also be the one-time git credentials password that bypasses MFA

_get_git_client(*platform*)

Configures the appropriate git client given the platform

Parameters **platform** (str) – the source control platform, one of 'github' or 'azure-devops'

Returns: the appropriate git client

Return type *AbstractGitClient*

class datarade.models.**DatasetContainer**(*database*, *username=None*, *password=None*)

Represents a target data repository that stores datasets, currently a database

Parameters

- **database** (*Database*) – the database to write datasets to
- **username** (Optional[str]) – a user with create table and insert permissions on the schema
- **password** (Optional[str]) – the password for the user

create_table(*dataset*)

Creates a table in the DatasetContainer with the correct attributes to store the data

Parameters **dataset** (*Dataset*) – the dataset to use as a blueprint for the table

3.2 Dataset Schemas

These schemas are all part of the aggregate schema Dataset. Reading the datasets out of a dataset catalog can lead to a lot of user input, similar to reading input data on a REST api. As such, it makes sense to apply validation to all data entered this way.

class datarade.schemas.**FieldSchema**(*, *only=None*, *exclude=()*, *many=False*, *context=None*, *load_only=()*, *dump_only=()*, *partial=False*, *unknown=None*)

A marshmallow schema corresponding to a datarade Field object

This schema is only called indirectly as an attribute for DatasetSchema

class datarade.schemas.**DatabaseSchema**(*, *only=None*, *exclude=()*, *many=False*, *context=None*, *load_only=()*, *dump_only=()*, *partial=False*, *unknown=None*)

A marshmallow schema corresponding to a datarade Database object

This schema is only called indirectly as an attribute for DatasetSchema

class datarade.schemas.**UserSchema**(*, *only=None*, *exclude=()*, *many=False*, *context=None*, *load_only=()*, *dump_only=()*, *partial=False*, *unknown=None*)

A marshmallow schema corresponding to a datarade User object

This schema is only called indirectly as an attribute for DatasetSchema

class datarade.schemas.**DatasetSchema**(*, *only=None*, *exclude=()*, *many=False*, *context=None*, *load_only=()*, *dump_only=()*, *partial=False*, *unknown=None*)

A marshmallow schema corresponding to a datarade Dataset object

This is used to control and validate input from an end user's DatasetCatalog. It verifies that the proper structure was received.

3.3 Git Client

This client allows a user to access files stored in a git-compliant source control repository. It supports publicly available repos hosted on GitHub and public or private git-compliant repos hosted on Azure Repos.

class datarade.git_client.**AbstractGitClient**

abstract get_file_contents(*file_path*)

This returns the contents of a file in the repo.

Parameters **file_path** (str) – the relative path to the file within the repo

Returns: the file contents as a string

Return type str

class datarade.git_client.**GitHubClient**(*repository*, *organization*, *branch*)

This client grants access to files on a public repo hosted on GitHub. The current implementation just goes right to the raw file to get the contents.

Parameters

- **repository** (str) – the name of the repo (e.g. https://github.com/<organization>/<repository>)

- **organization** (str) – the user or organization that owns the repo (see repository example)
- **branch** (str) – the name of the branch to use

get_file_contents(*file_path*)
This performs a basic get on the raw contents on GitHub. It's not ideal, but does the job.

Parameters **file_path** (str) – the relative path to the file within the repo

Returns: the contents of the file as a string

Return type str

class datarade.git_client.**AzureReposClient**(*repository*, *organization*, *project*, *branch*, *username*, *password*)
This client grants access to files on a public or private git-compliant repo hosted on Azure Repos. It uses Microsoft's azure-devops package, which is currently in beta for versions 5.0 and 6.0.

Parameters

- **repository** (str) – the name of the repo (e.g. https://dev.azure.com/<organization>/<project>/_git/<repository>)
- **organization** (str) – the organization that owns the Azure DevOps instance (see repository example)
- **project** (str) – the project within the organization that contains the repo (see repository example)
- **branch** (str) – the name of the branch to use
- **username** (str) – the username for the repo
- **password** (str) – the password for the repo

static _get_client(*organization*, *username*, *password*)
This method configures this client to connect to Azure Repos.

Parameters

- **organization** (str) – the organization within the Azure DevOps instance
- **username** (str) – the username for the organization
- **password** (str) – this can be a password for no MFA, or the git credentials password that overrides MFA

Returns: an instance of the azure-devops v6.0 GitClient

Return type GitClient

get_file_contents(*file_path*)
This uses get_item_content() to get the file contents from Azure Repos.

Parameters **file_path** (str) – the relative path to the file within the repo

Returns: the contents of the file as a string

Return type str

PYTHON MODULE INDEX

d

INDEX

Symbols

A

B

C

D

F

G

S

U

W