Metadata SIG Update

Rob Barnsley

Context

- For most communities, metadata is a very important aspect of data management
  - Organise data meaningfully
  - To aid findability (FAIR)

- Metadata handling within Rucio is a nascent development area
  - Historically developed to meet the requirements of HEP but a lot of burgeoning communities
  - Metadata catalogues can be very bespoke, want to keep the interfaces largely agnostic
  - Purpose of this group is to identify missing functionality and align requirements across communities

- From an SKAO perspective
  - Operator level: Rucio metadata commands CLI/API
  - User level: Looking to leverage **I**nternational **V**irtual **O**bservatory **A**lliance (IVOA) protocols/standards (Dave M. will talk more about this)
  - Work done by the team has focused on enabling functionality and interfaces

Rucio metadata refresher (1/2)

- Out of the box, Rucio has two distinct metadata stores:
  - A “base” metadata store that uses columns from the **dids** table; these are fixed fields, often HEP specific or related to file metadata
  - A “custom” metadata store that stores key-value pairs in a JSON column in the **did_meta** table

- When a user sets metadata, the server will call `manages_key(`*`key_name`*`)` for each store *in order* until it evaluates true:
  - As the base metadata store is always first in the list, and `manages_key` evaluates to **true** if the requested key name == a column name (with some exceptions), it will always be set in this store
  - If the key is not set in this base metadata store, Rucio will insert it into the custom metadata store (provided that the backend database implements JSON columns)

Rucio metadata refresher (2/2)

- Rucio uses a plugin system for metadata stores
- Alternative custom metadata stores can be implemented using this system
- Able to write interfaces to different backends by (see stub class DidMetaPlugin: did_meta_plugin_interface.py):
  - Defining functions for how to get/set/delete metadata,
  - Defining a function to handle listing dids (`list-dids`) with metadata filtering,
  - Defining a function to specify which keys the plugin should handle, and
  - Specifying this plugin to be used in the Rucio server configuration file
- Two functions for listing DIDs by metadata: `list-dids and list-dids-extended`
  - The former **only** allows filtering from the base metadata store
  - The latter allows filtering selecting the relevant metadata store using the plugin system

Work done between 1.26 to 1.29 (Filtering engine)

- Up until 1.26, there was limited support for metadata filtering via the `list-dids` and `list-dids-extended` functions:
  - **only** the equality operator, and
  - **only** the logical AND operator

- Following a request for inequality operators, Gabriele F. worked on creating a prototype language / implementation => “Filter engine” created
  - This engine supports an extended syntax including inequalities and wildcards, viz =, !=, >=, <=, >, <, LIKE, NOT LIKE, as well as logical OR

- From 1.27 onwards, `list-dids` passed `--filter` input through to the filtering engine
  - But only applied to filtering on columns in the base metadata store

- From 1.28 onwards (Oracle support in 1.29), `list-dids-extended` also passed the `--filter` input through to the filtering engine
  - This meant you could now filter the custom (JSON) metadata store

Work done between 1.26 to 1.29 (Filtering engine)

Example 1: Date ranges and logical operators

Set metadata (two files with a date and one with a string):

```
[root@74ba4f5f5141 rucio]# rucio set-metadata --did test:file1 --key test_key_1 --value 2022-06-01
[root@74ba4f5f5141 rucio]# rucio set-metadata --did test:file2 --key test_key_1 --value 2022-06-02
[root@74ba4f5f5141 rucio]# rucio set-metadata --did test:file3 --key test_key_2 --value example_text
```

Query with date range and logical AND:

```
[root@74ba4f5f5141 rucio]# rucio list-dids-extended test:* --filter "test_key_1 <= 2022-06-02, test_key_2 == example_text"
+--------------+--------------+
| SCOPE:NAME   | [DID TYPE]   |
|--------------+--------------|
+--------------+--------------+
```

Same clauses but with logical OR:

```
[root@74ba4f5f5141 rucio]# rucio list-dids-extended test:* --filter "test_key_1 <= 2022-06-02; test_key_2 == example_text"
+--------------+--------------+
| SCOPE:NAME   | [DID TYPE]   |
|--------------+--------------|
| test:file1   | N/A          |
| test:file2   | N/A          |
| test:file3   | N/A          |
+--------------+--------------+
```

Compound inequalities also supported (e.g. `2022-06-01 <= test_key_1 <= 2022-06-02`)

Work done between 1.26 to 1.29 (Filtering engine)

Example 2: Wildcard queries

Set metadata (three files with string):

```
[root@74ba4f5f5141 rucio]# rucio set-metadata --did test:file1 --key test_key_1 --value test_value_1
[root@74ba4f5f5141 rucio]# rucio set-metadata --did test:file2 --key test_key_1 --value test_value_2
[root@74ba4f5f5141 rucio]# rucio set-metadata --did test:file3 --key test_key_1 --value test_value_3
```

Query with equals and wildcarded value:

```
[root@74ba4f5f5141 rucio]# rucio list-dids-extended test:* --filter "test_key_1 == test_value_*"
+--------------+--------------+
| SCOPE:NAME   | [DID TYPE]   |
|--------------+--------------|
| test:file1   | N/A          |
| test:file2   | N/A          |
| test:file3   | N/A          |
+--------------+--------------+
```

NOT LIKE also possible with != and * operators.

Work done between 1.26 to 1.29 (Plugins)

- From 1.28 onwards, a new metadata plugin to interface to an external mongodb collection was made available, e.g. rucio.cfg

```
[metadata]
plugins = rucio.core.did_meta_plugins.mongo_meta.MongoDidMeta
mongo_service_host=mongo
mongo_service_port=27017
mongo_db=test_db
mongo_collection=test_collection
```

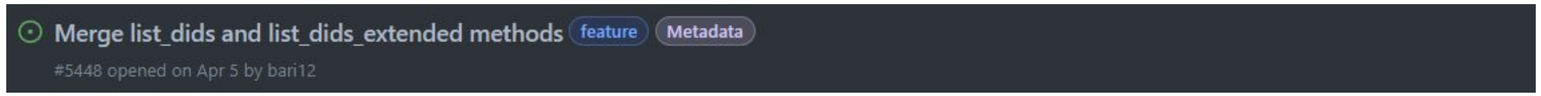
- From 1.29 onwards, another metadata plugin to interface to an (external) postgres database is available, e.g. rucio.cfg

```
[metadata]
plugins = rucio.core.did_meta_plugins.postgres_meta.ExternalPostgresJSONDidMeta
postgres_service_host=postgres
postgres_service_port=5433
postgres_db=metadata
postgres_user=rucio
postgres_password=secret
postgres_db_schema=public
postgres_table=dids
postgres_table_is_managed=True
```

  - Dumps everything into JSON column

Future

- Currently only one issue open that directly touches on metadata work: #5448



  I will be working on this for the next release, however

- Work done in this area by myself and the team is mostly driven by SKAO requirements with timescales defined by our program roadmap

- Further substantial work has not been officially planned for this current increment (-> mid Dec) but may happen in the next. Such work *may* include:
  - To test how performant searching is (internal and external) and how this scales with number of files, complexity of query and structure of metadata
  - Enabling better integration with IVOA protocols/standards e.g.
    - a “mixed mode” external postgres plugin,
    - the ability to use more than 2 plugins

Thanks!

- Welcome suggestions and contributions to this component
- #metadata on Rucio Slack