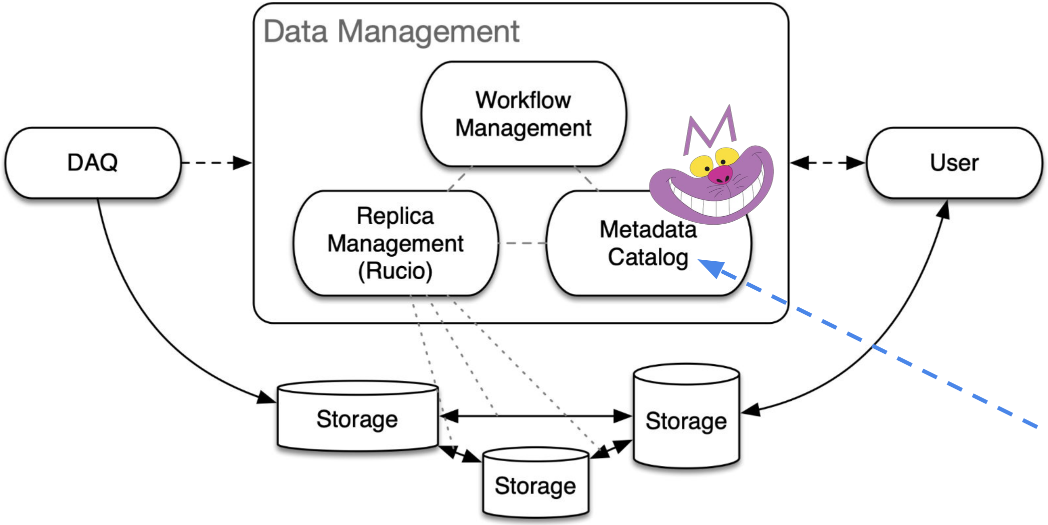
MetaCat - metadata catalog for Rucio-based data management systems

Igor Mandrichenko

Rucio Workshop, 11/10/2022

What is MetaCat ?



MetaCat = **Meta**data **Cat**alog for data management systems where Rucio can be used as the Replica Manager

MetaCat Target Users

- Primary: DUNE
- Other FNAL experiments migrating from SAM
    - SAM is DM system used at FNAL, combining all 3 functions
- HEP experiments
- Rucio users

Functions

- Store and make available metadata associated with objects (files) and object collections (datasets)

- Provide efficient query mechanism to select objects (files) matching the user criteria

- Provide flexible, efficient, integrated access to external metadata sources

Conceptual Compatibility with Rucio

Rucio:

- Scope
- Scope:Name (DID)
- Container
- Dataset

MetaCat:

- Namespace
- Namespace:Name
- Dataset
  - In MetaCat, there is no distinction, a dataset can contain files and/or other datasets

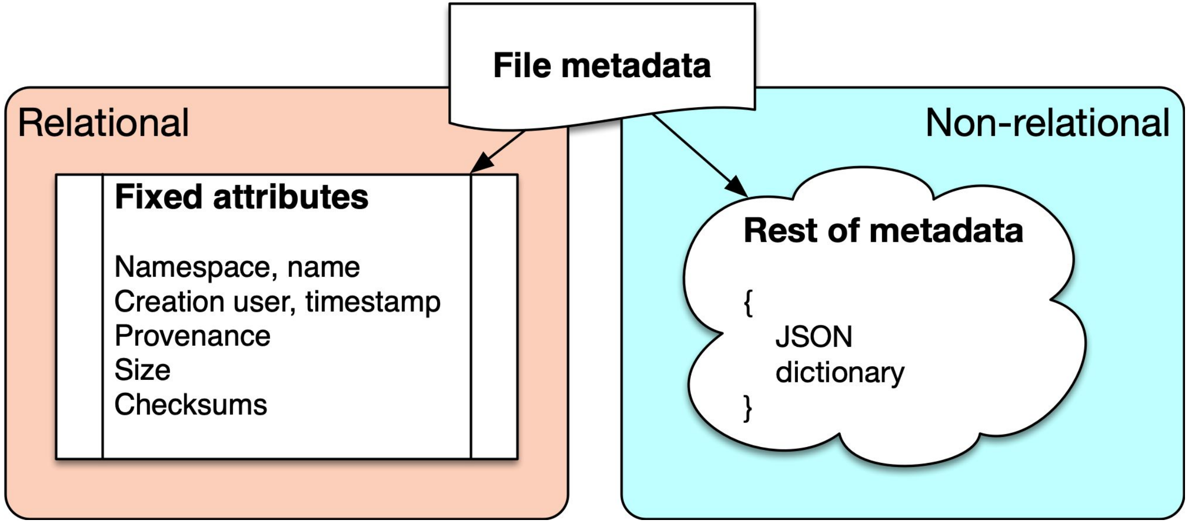
Being conceptually compatible, MetaCat does not depend on Rucio, nor does it communicate with Rucio directly, therefore can be used with other replica management systems

Files or Objects

MetaCat unit of operation: file or *object* - abstract entity with the following properties

- Fixed attributes - every file has them
  - Unique text ID (assigned by user or auto generated)
    - Immutable
  - Unique name within a namespace (Rucio: scope, name)
    - Can be renamed
    - Name can be auto-generated
  - Creator username, timestamp
  - File Provenance
    - Parents, children (file A was created from files B, C, D)
- Rest of metadata - arbitrary JSON dictionary

File Metadata



Rest of metadata:

- Application-defined
- Non-relational, fluid
- JSON dictionary - arbitrarily complex
- Restrictions can be defined via categories and/or dataset restrictions
- Implementation: Postgres jsonb type, GIN-indexed

Very few attributes are in relational schema

- Common attributes
- Attributes used by MetaCat itself
- Better indexing, fast lookup, table joins (datasets, namespaces, provenance, …)

Datasets

- Dataset has a name within a namespace
- Contains files
  - Many-to-many (a file can belong to many datasets)
  - Explicitly added/removed
- Combines Rucio dataset and container functionality
  - Datasets may contain other datasets, recursively (Rucio: container)
- Standard attributes:
  - Creator username, timestamp
  - Dataset flags
    - Frozen - files can not be added or removed
    - Monotonic - files can only be added
- Rest of metadata - arbitrary JSON dictionary

Queries

- Querying is the fundamental function of MetaCat
  - Find all files matching a set of criteria expressed in terms of their metadata, provenance, external metadata

- Written in Metadata Query Language (MQL)

- A query can be named, saved and reused inside another query or as is
  - Conceptually similar to a relational database view

Datasets vs Queries

Dataset - explicit collection of files

- Recorded in the database
- Files added/removed explicitly
- Has a name within a namespace

Relational DB: table

Query - *instructions* how to select files from a dataset or datasets

- Recalculated every time it runs
- Results can change at any time
- Can be saved under a name within a namespace and reused by name

Relational DB: SQL "select"

Bridge:

Query results can be saved as a new dataset or added to an existing dataset

Metadata Query Language (MQL)

```
files from dune:raw
    where DUNE.reco_version = “v1.2”
    limit 1000
```

- Keyword - files query
- Dataset to select files from (DID)
- Metadata filtering
- Limit results to first 1000 files

```
files from dune:raw_2019 where
        DUNE.reco_version in (“v1.2”,”v1.3”)
          and core.file_type = “root”
        or DUNE.reco_version = “v1.0”
```

- Parameter category
- Boolean algebra

```
union (
    files from dune:raw_2019
             where DUNE.reco_version = “v1.2”
    ,
    files from dune:raw_2020
             where DUNE.detector = “near” and
                 DUNE.reco_version >= “v1.3”
) where data.format in (“root”, “hdf5”)
```

- Queries can be combined using “union”,”join”, “-” (subtraction)
- Metadata filters can be applied again to the combined query

MQL Compiler

- MQL query is compiled into SQL query and executed by the database engine
  - Exception: external metadata access - executed by the MetaCat application server
- Resulting SQL query complexity in terms of number of table joins *does not* depend on the complexity of the original MQL query
- MQL metadata expressions are compiled into JSON/JSON Path expressions interpreted by Postgres

Parameter Categories

A mechanism to restrict the fluidity of the metadata schema in application-specific way

MetaCat parameter name:

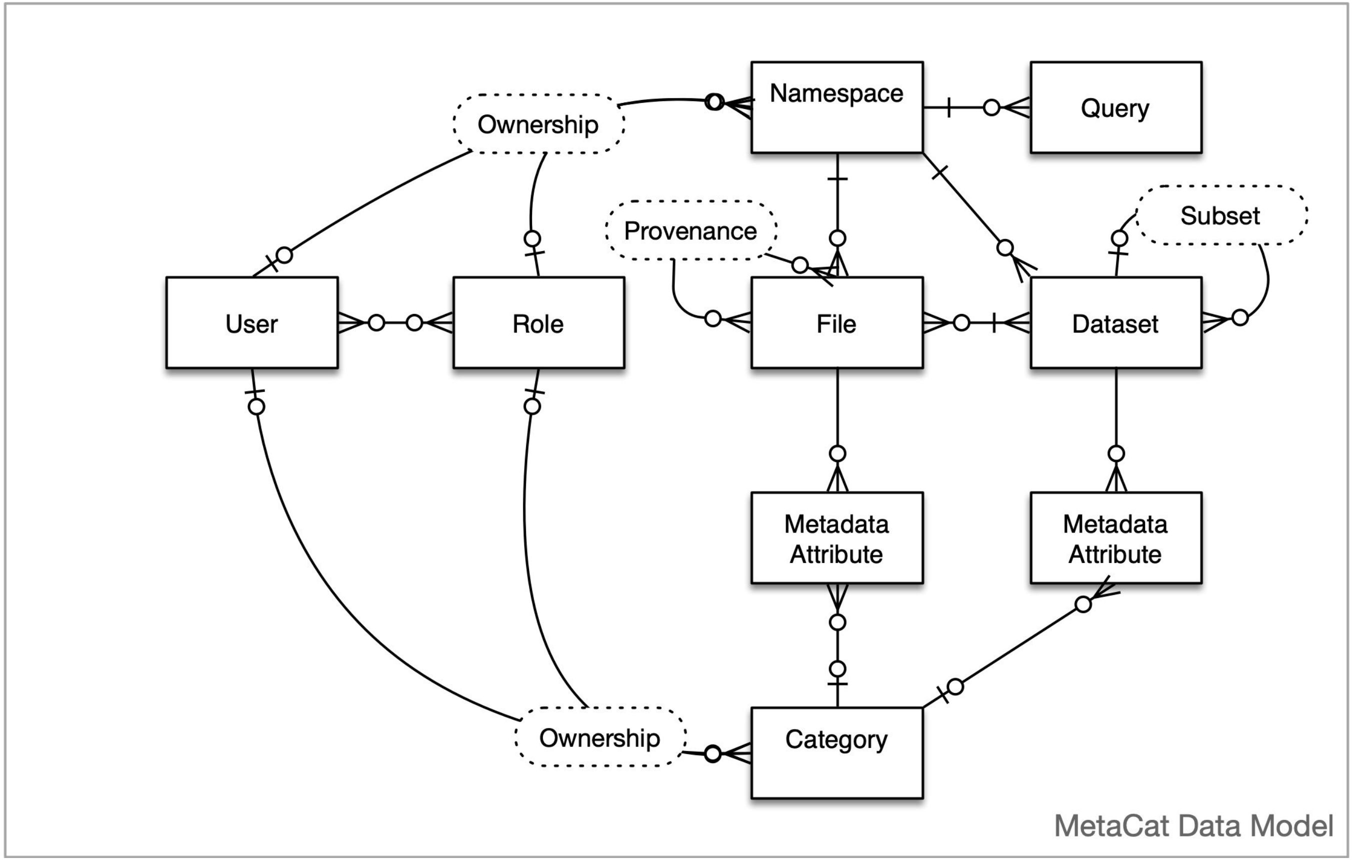
<category>.<parameter name>

Category owner can restrict areas of the metadata namespace

- Parameter types
  - Int, float, string, boolean, list of ints, floats, … dict, …
- Accepted values
  - Range, enumeration, pattern
- Restricted category: only known parameter names are allowed

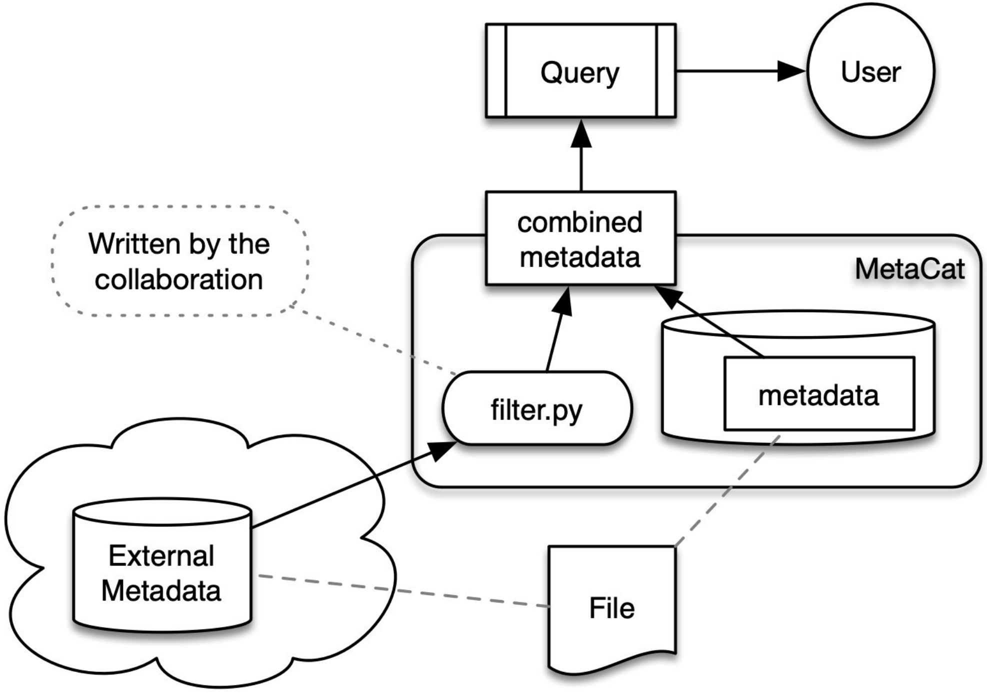
Enforced at the time of the file declaration

Data Model



MetaCat Data Model

External Metadata Sources



Use case:

- Run conditions are stored in the Runs database by run number
- Files need to be selected based on some run conditions values
- We do not want to duplicate run conditions data in MetaCat as file metadata
- Implemented for ProtoDUNE

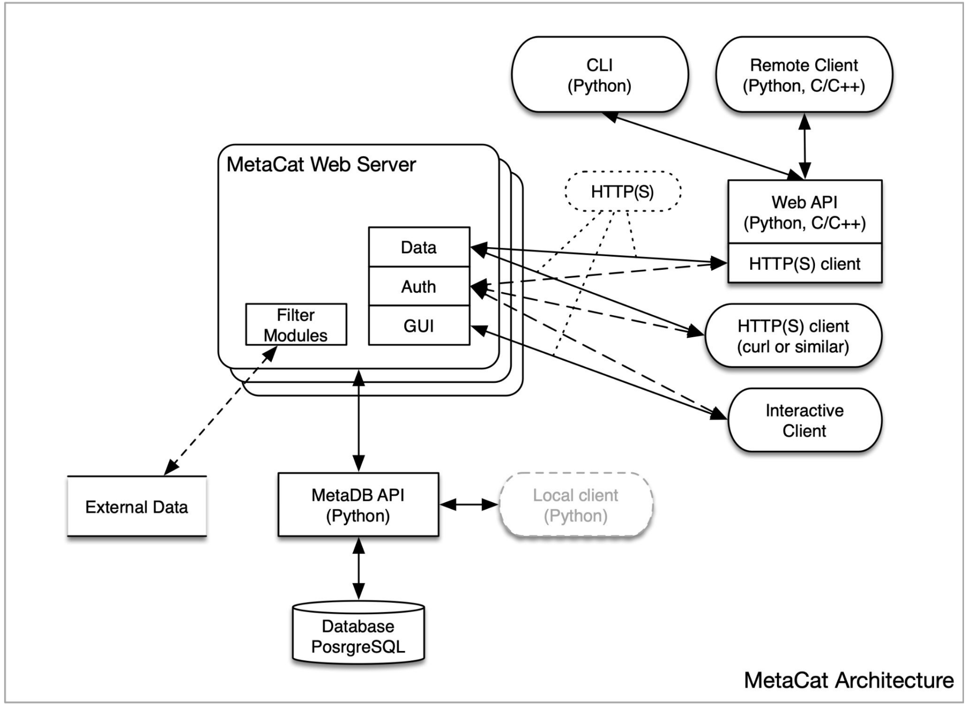
External Filter in MQL

```
# real life DUNE example

filter rucio_replicas() (
    files from dc4:dc4
         limit 100
) where "DUNE_CERN_EOS" in rucio.rses
```

- Filter name - collaboration defined
- Filter is applied to the results of the Intermediate query
- This filter contacts Rucio and gets replica information for the given files
- Injects the replica information as new metadata making it available for querying and as the query output
- Makes the replica location information appear as if it is stored in MetaCat, but it is not

MetaCat Architecture



MetaCat Architecture

Software Stack

- PostgreSQL v12 - the database
- Python3 (both client and server)
- psycopg2 - Python/PostrgeSQL

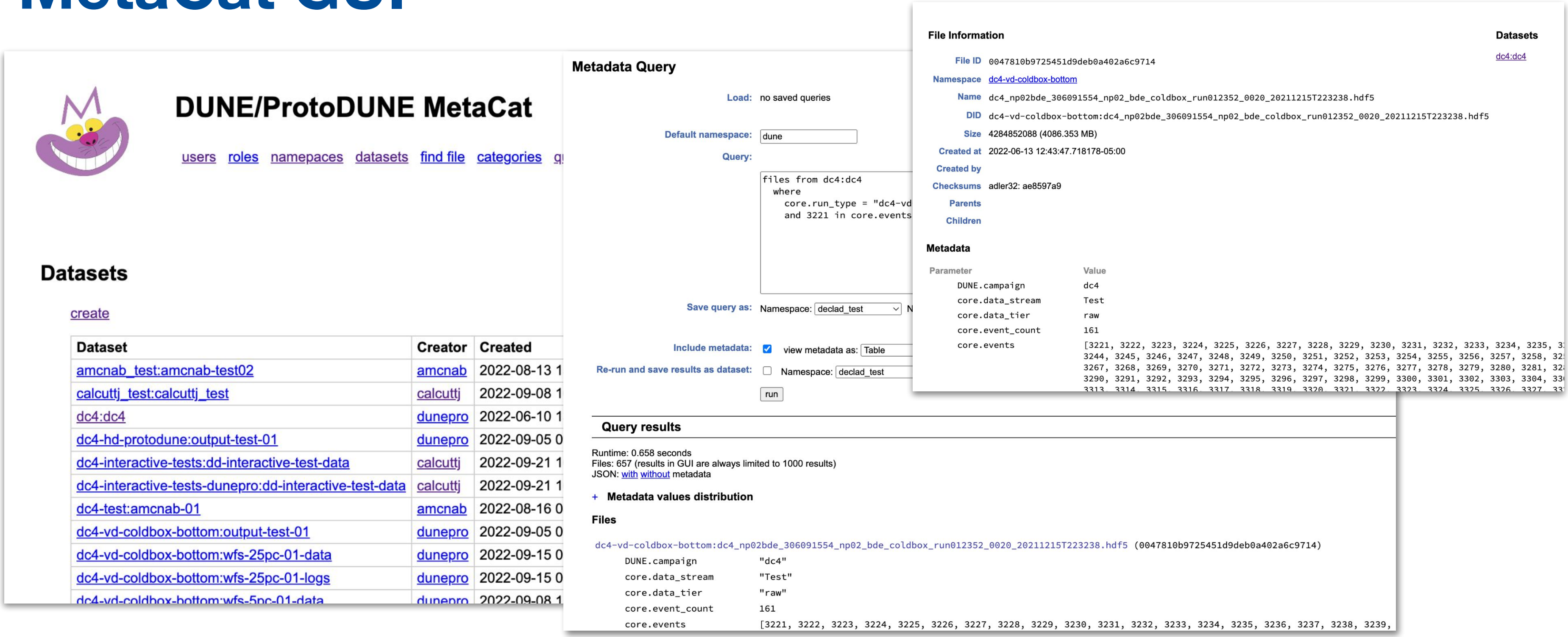
Client API (Python)

- Uses HTTP/HTTPS, requests Python library
- Datasets - create, get, list, add files, remove files, update metadata, …
- Namespaces - create, get, list, …
- Files - declare, get, update metadata, provenance, ...
- Query - run, run asynchronously, save results as dataset, add results to dataset
- Parameter categories, validation
- Authentication
  - JWT tokens

Command Line Interface Functions

- Client authentication
  - Log in (obtain JWT token, save it in local FS, similar to Rucio)
- Datasets
  - Create, list, show, update
- Namespaces
  - Create, list, show
- Files
  - Declare, show, add, update
- Metadata validation
- Parameter categories
- Queries

MetaCat GUI



DUNE/ProtoDUNE MetaCat

users roles namespaces datasets find file categories q

Datasets

create

Dataset	Creator	Created
amcnab_test:amcnab-test02	amcnab	2022-08-13 1
calcuttj_test:calcuttj_test	calcuttj	2022-09-08 1
dc4:dc4	dunepro	2022-06-10 1
dc4-hd-protodune:output-test-01	dunepro	2022-09-05 0
dc4-interactive-tests:dd-interactive-test-data	calcuttj	2022-09-21 1
dc4-interactive-tests-dunepro:dd-interactive-test-data	calcuttj	2022-09-21 1
dc4-test:amcnab-01	amcnab	2022-08-16 0
dc4-vd-coldbox-bottom:output-test-01	dunepro	2022-09-05 0
dc4-vd-coldbox-bottom:wfs-25pc-01-data	dunepro	2022-09-15 0
dc4-vd-coldbox-bottom:wfs-25pc-01-logs	dunepro	2022-09-15 0
dc4-vd-coldbox-bottom:wfs-5pc-01-data	dunepro	2022-09-08 1

Metadata Query

Load: no saved queries

Default namespace: dune

Query:

```
files from dc4:dc4
  where
    core.run_type = "dc4-vd
    and 3221 in core.events
```

Save query as: Namespace: declad_test N

Include metadata: ☑ view metadata as: Table

Re-run and save results as dataset: □ Namespace: declad_test

run

Query results

Runtime: 0.658 seconds
Files: 657 (results in GUI are always limited to 1000 results)
JSON: with without metadata

+ **Metadata values distribution**

Files

```
dc4-vd-coldbox-bottom:dc4_np02bde_306091554_np02_bde_coldbox_run012352_0020_20211215T223238.hdf5 (0047810b9725451d9deb0a402a6c9714)
    DUNE.campaign         "dc4"
    core.data_stream      "Test"
    core.data_tier        "raw"
    core.event_count      161
    core.events           [3221, 3222, 3223, 3224, 3225, 3226, 3227, 3228, 3229, 3230, 3231, 3232, 3233, 3234, 3235, 3236, 3237, 3238, 3239,
```

File Information

File ID	0047810b9725451d9deb0a402a6c9714
Namespace	dc4-vd-coldbox-bottom
Name	dc4_np02bde_306091554_np02_bde_coldbox_run012352_0020_20211215T223238.hdf5
DID	dc4-vd-coldbox-bottom:dc4_np02bde_306091554_np02_bde_coldbox_run012352_0020_20211215T223238.hdf5
Size	4284852088 (4086.353 MB)
Created at	2022-06-13 12:43:47.718178-05:00
Created by	
Checksums	adler32: ae8597a9
Parents	
Children	

Datasets

dc4:dc4

Metadata

```
Parameter             Value
    DUNE.campaign         dc4
    core.data_stream      Test
    core.data_tier        raw
    core.event_count      161
    core.events           [3221, 3222, 3223, 3224, 3225, 3226, 3227, 3228, 3229, 3230, 3231, 3232, 3233, 3234, 3235, 3
                          3244, 3245, 3246, 3247, 3248, 3249, 3250, 3251, 3252, 3253, 3254, 3255, 3256, 3257, 3258, 32
                          3267, 3268, 3269, 3270, 3271, 3272, 3273, 3274, 3275, 3276, 3277, 3278, 3279, 3280, 3281, 32
                          3290, 3291, 3292, 3293, 3294, 3295, 3296, 3297, 3298, 3299, 3300, 3301, 3302, 3303, 3304, 33
                          3313, 3314, 3315, 3316, 3317, 3318, 3319, 3320, 3321, 3322, 3323, 3324, 3325, 3326, 3327, 33
```

https://metacat.fnal.gov:9443/dune_meta_prod/app/gui/index

Existing Databases

- DUNE/ProtoDUNE
  - 16.7 M files
  - ~480 M name-value metadata pairs
  - 21 GB “files” table + 6 GB metadata index over non-relational JSON data
  - Total database size: ~40 GB
    - ~ 2.5 KB/file
  - Passed the data challenge, plan is to make it official production soon
- NOvA - *not used, but imported to test scalability*
  - 191.2 M files
  - ~5.3 B name-value metadata pairs
  - 221 GB “files” table + 35 GB metadata index over non-relational JSON data
  - Total database size: ~326 GB
    - ~ 1.7 KB/file

References

Documentation: https://metacat.readthedocs.io

vCHEP 2021 paper: https://cdcvs.fnal.gov/redmine/attachments/download/64700/MetaCat%20CHEP%202021%20paper%20v5.pdf

DUNE MetaCat GUI: https://metacat.fnal.gov:9443/dune_meta_prod/app/gui/index

Backup

MQL syntax: file queries

```
<file query>: files [from [dataset|datasets] <dataset selector list> [,...]]
            | <file query> [where <metadata expression>]
            | <file query> [skip <integer>]
            | <file query> [limit <integer>]
            | query <saved query namespace>:<saved query name>
            | filter <filter name>( <parameter> [,...] ) ( <file query> [,...] )
            | union ( <file query> [,...] )
            | join ( <file query> [,...] )
            | <file query> - <file query>
            | children ( <file query> )           # file provenance
            | parents ( <file query> )
            | ( <file query> )
```

MQL syntax: metadata expressions

```
<metadata expression>: <scalar> <cmp op> <constant>
         | <attribute> [[not] present]                  # file has this attr
         | <constant> [not] in <attribute>              # in list or dict
         | <scalar> [not] in <constant> : <constant>    # range
         | <scalar> [not] in ( <constant> [,...] )      # enumeration
         | ( <metadata expression > )
         | ! <metadata expression>
         | <metadata expression> and <metadata expression>
         | <metadata expression> or <metadata expression>

<cmp op>: = == != < <= > >= ~ ~* !~ !~*
```

Named Queries

```
query DUNE:supernova_production_latest_version
    where len(core.events) > 10
    limit 100


join (
 query DUNE:supernova_production_latest_version,
 query joe:my_favorite_files,
 files from dune:all
    where
        run.quality > 10
        and core.runs[any] in 7375:7380
)
```

A query could be created, debugged and saved to be reused by name

Datasets and Subsets in MQL

```
files
    from dune:raw_2019
        with children
        recursively
    where
        created_timestamp > ‘2019-05-01’
        and reco_version = “v1.2”
```

Include files from the top dataset

- and its subsets
- recursively

Other ways to use filters

```
filter random_mix(0.4, 0.6)
(
    files from dune:raw_2019                          # file set 1
             where reco_version = “v1.2”,

     files from dune:raw_2020                         # file set 2
             where detector = “near” and
                 reco_version >= “v1.3”
)
```

A filter *does not* actually have to access any external data.

This “random_mix” filter mixes two file sets into one according to target ratios

File Provenance

File provenance supported by MetaCat

- A file can have zero or more parent files and zero or more child files
- Which files were used to create which files

File Provenance in MQL

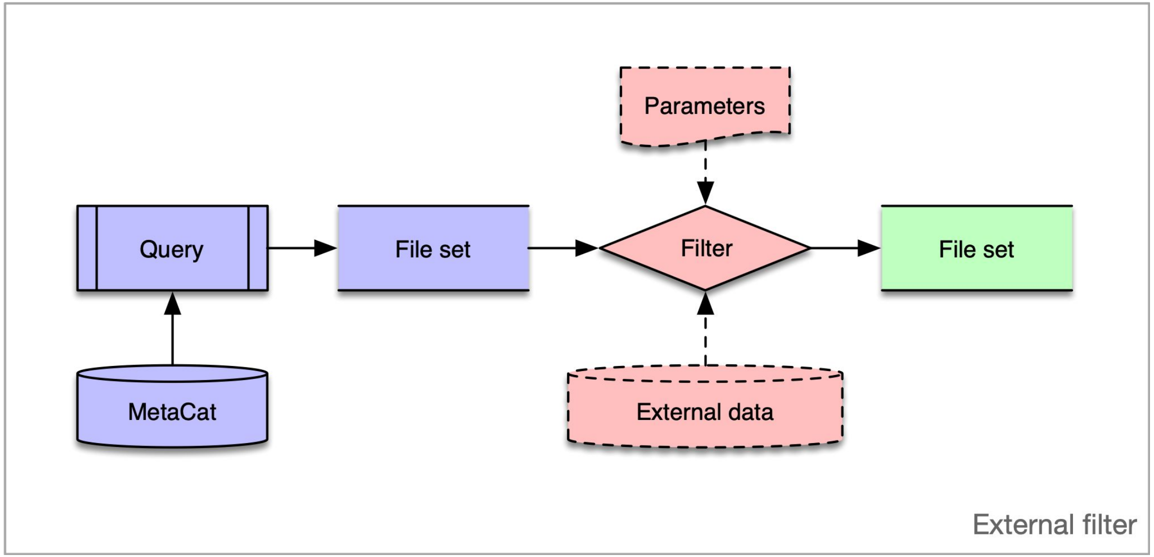
```
children ( file scope:file.data )          # children of a single file

parents (                                  # parents of all files
    files from dune:raw_2019               # in the file set
        where reco_version = “v1.2”
)

files from dune:raw - parents(         # unprocessed files
    files from dune:processed
)

files from dune:raw -                      # files without any children
    parents(
        children(files from dune:raw)
    )
```

External Filter



External filter

- A filter takes a file set (one or more) - results of a query (or queries)
- Produces new file set
  - *Optionally*: accesses the external data
  - removing or even adding files
  - modifying metadata
  - injecting new metadata

Filter is a python class provided by the collaboration and plugged into MetaCat server instance.

Not every user can add a filter - security

Arrays and Dictionaries

`bit_mask[2] = 1`	Array element by index
`config[“version”] > “2.3”`	Dictionary access by key
`runs[any] = 1234`	Any array element
`1234 in runs`	Array contains element
`runs[all] < 1234`	All elements
`config[any] != “raw”`	Any dictionary element
`len(core.events) > 10`	Array length

Ranges and Enumerations

```
x in 1234:1332
```

Range of values

```
x in (1234,1235,2345)
```

Enumerated set of values

```
run_type in (“calibration”,”test”)

file_type not in (“mc”,”test”)
```

- Can be strings too

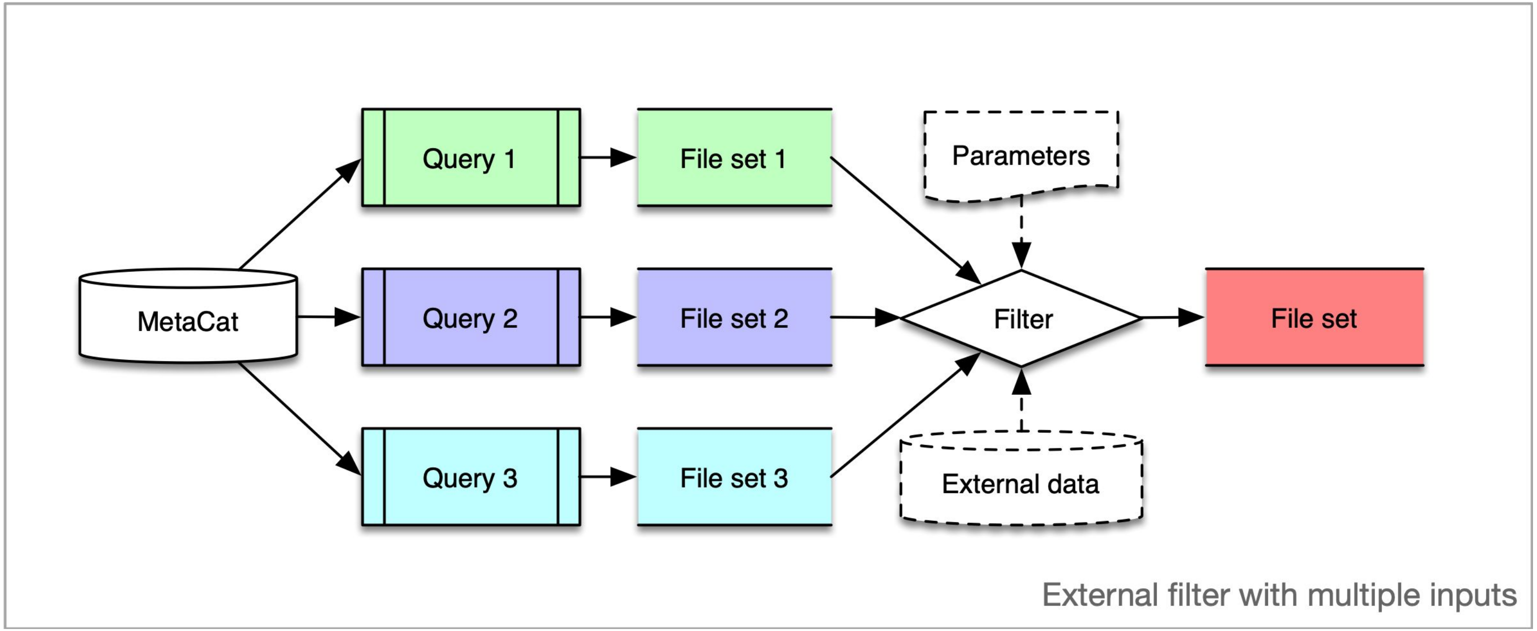
Dataset Restrictions on File Metadata

Dataset can have restrictions on metadata for its files

- Parameter types, allowed values
  - Similar to categories
- Required parameters

When declaring a file or adding a file to a dataset, both dataset and parameter category restrictions apply

External Filter with Multiple Inputs



External filter with multiple inputs

A filter can take multiple file sets as input
and combine them into a single output file set