Finanziato dall'Unione europea NextGenerationEU

Ministero dell'Università e della Ricerca

Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

ICSC

Centro Nazionale di Ricerca in HPC,
Big Data and Quantum Computing

Extending Rucio to support external metadata catalog

Luca Pacioselli (INFN Perugia) on behalf of IDL

"ICSC and Spoke 2 - Where Are We Now?"
Catania, December 10-12 2024

Table of Contents

- Introduction
- Rucio Overview
- Where are we?
- DID-metadata plugin
- IDL Rucio client
- What’s next?

Introduction

- One key point of the interoperable Data Lake (IDL) project:
  - develop an interoperable, distributed **Data Lake** service with **state-of-the-art open-source technologies**

See Nicolò Magini's talk

Interoperable Data Lake (IDL)
Nicolò Magini

"ICSC and Spoke2 – Where Are We Now?",
Catania, 10-12 December 2024

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC
Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing
ICSC Italian Research Center on High-Performance Computing, Big Data and Quantum Computing
Missione 4 • Istruzione e Ricerca

- The presented work focuses on WP1@IDL: developing an **end-to-end prototype** for DM in a distributed environment and on the extension of Rucio to **support external metadata catalogs**

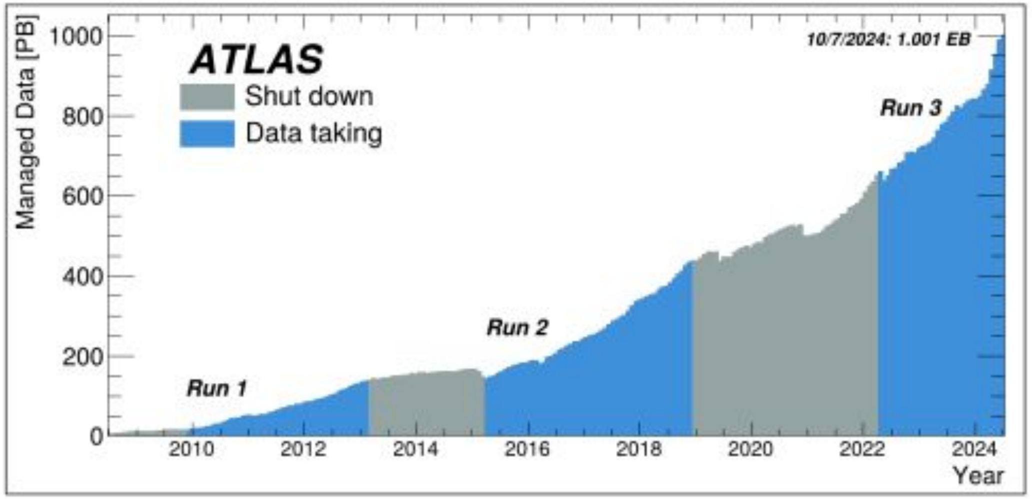
Data Management solution: Rucio

SCIENTIFIC DATA MANAGEMENT
RUCIO

- Rucio is an open-source software for large-scale DM:
  - initially developed by CERN for ATLAS experiment's needs
  - now widely adopted across HEP experiments and beyond (CMS, Belle II, DUNE, etc...)
- Key features:
  - handles large volumes O(ExaBytes) of data
  - data can be stored in geographically distributed storage endpoints
- For IDL we need to **extend** its standard functionality:
  - external metadata database (DB) and tailored client

Proven to work in very challenging environments

Rucio@ATLAS: 1+ Exabyte, 120 data centres



"Volume of Data managed by Rucio: source"

Implementation Strategy: a quick overview

❑ To develop an **automated deployment of a Rucio server** Kubernetes (k8s) cluster

- this include the deployment of **S3 cloud storage** endpoint(s)

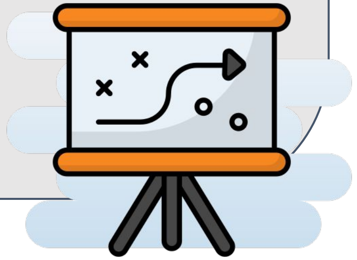
❑ To **extend** the **Rucio** support for external **metadata catalogs** and confine the interactions with the external DB to the server layer:

- strict control on the Rucio-DB interactions (AuthN/Z and atomicity of the operations)

❑ To prototype a **Rucio client wrapper for IDL**-specific requirements :

- to manage data and metadata in a coherent manner
- tailored query system to interrogate the external DB

❑ To provide documentation outlining the steps of the work



Where are we?

✓ Deployed an automated (custom Docker image) Rucio server instance on a k8s cluster (nginx, HTTPS over TLS)

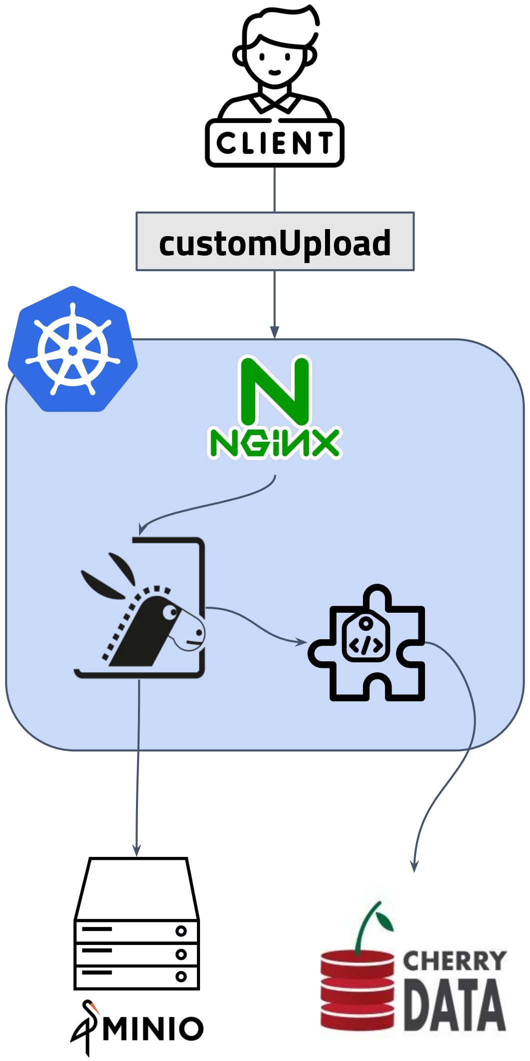
- Configured a storage endpoint with MinIO as an S3-compatible RSE

✓ Functional prototype in Python of a **DID-metadata plugin** to communicate with an external database: AyraDB

✓ Tailored **Rucio client**:

- custom methods to seamlessly interact with the external DB via the DID-metadata plugin
- Python binding to use all the standard methods of Rucio

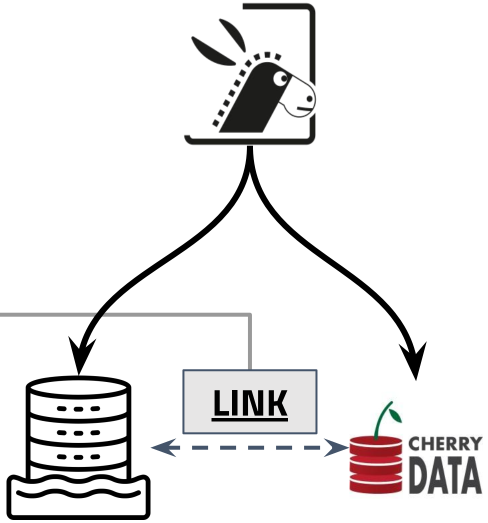
~ Documentation will be available on a Spoke2 github repository



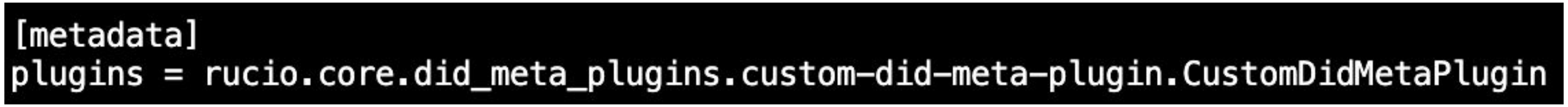
Extension to external metadata catalog: AyraDB

In the IDL project, Rucio plays a central role in the ingestion phase:

- **data** are stored in the **Data Lake**
- **metadata** are stored in the metadata DB, implemented with **AyraDB**
- associates a **LINK**, Data IDentifier (DID), that links the data with its metadata



The concept of **DID-metadata plugins** exists in Rucio:



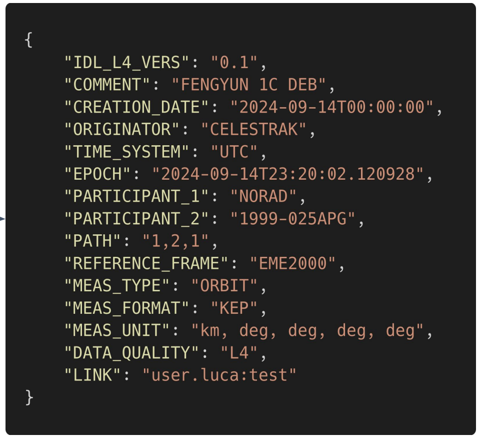
```
[metadata]
plugins = rucio.core.did_meta_plugins.custom-did-meta-plugin.CustomDidMetaPlugin
```

AyraDB, provided by CherryData:

- is a high-throughput DB
- manages read- and write-intensive workloads with all types of data (e.g. JSON files)

Custom DID-metadata plugin

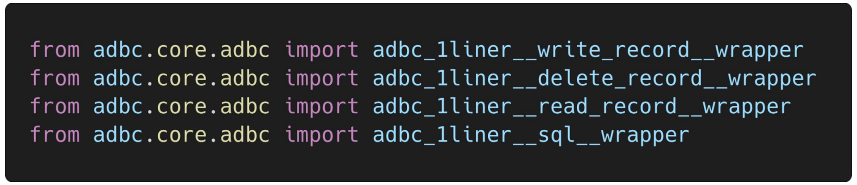
- To **extend metadata catalog** we designed a DID-metadata plugin that allows for external databases to be integrated



```
{
    "IDL_L4_VERS": "0.1",
    "COMMENT": "FENGYUN 1C DEB",
    "CREATION_DATE": "2024-09-14T00:00:00",
    "ORIGINATOR": "CELESTRAK",
    "TIME_SYSTEM": "UTC",
    "EPOCH": "2024-09-14T23:20:02.120928",
    "PARTICIPANT_1": "NORAD",
    "PARTICIPANT_2": "1999-025APG",
    "PATH": "1,2,1",
    "REFERENCE_FRAME": "EME2000",
    "MEAS_TYPE": "ORBIT",
    "MEAS_FORMAT": "KEP",
    "MEAS_UNIT": "km, deg, deg, deg, deg",
    "DATA_QUALITY": "L4",
    "LINK": "user.luca:test"
}
```

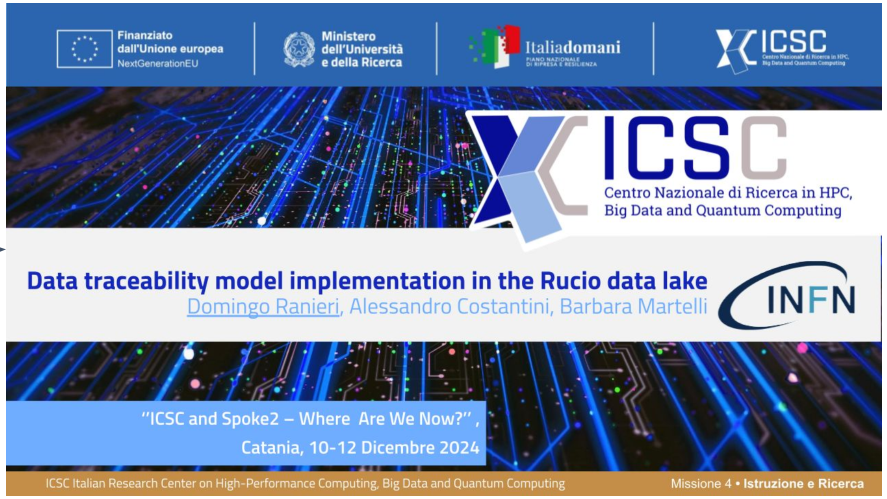
Example of IDL metadata

- The **AyraDB Connector** (**ADBC**) client, by CherryData, allows smooth implementation to perform operations on metadata



```
from adbc.core.adbc import adbc_1liner__write_record__wrapper
from adbc.core.adbc import adbc_1liner__delete_record__wrapper
from adbc.core.adbc import adbc_1liner__read_record__wrapper
from adbc.core.adbc import adbc_1liner__sql__wrapper
```

  - Custom DID-metadata plugin developed also to support interactions with the **blockchain** (see Domingo Ranieri's talk)



Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

Data traceability model implementation in the Rucio data lake

Domingo Ranieri, Alessandro Costantini, Barbara Martelli

INFN

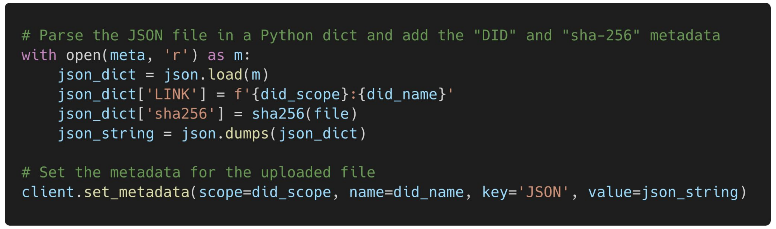
"ICSC and Spoke2 – Where Are We Now?",
Catania, 10-12 Dicembre 2024

ICSC Italian Research Center on High-Performance Computing, Big Data and Quantum Computing
Missione 4 • Istruzione e Ricerca

INFN PERUGIA
Istituto Nazionale di Fisica Nucleare
Sezione di Perugia

Dedicated Rucio client

- To ensure seamless interactions with the plugin, a dedicated **Rucio client** is needed
- **Python bindings** to include Rucio's default methods for a unified client experience



```python
# Parse the JSON file in a Python dict and add the "DID" and "sha-256" metadata
with open(meta, 'r') as m:
    json_dict = json.load(m)
    json_dict['LINK'] = f'{did_scope}:{did_name}'
    json_dict['sha256'] = sha256(file)
    json_string = json.dumps(json_dict)

# Set the metadata for the uploaded file
client.set_metadata(scope=did_scope, name=did_name, key='JSON', value=json_string)
```

customUpload:

- **combines** upload and set-metadata
- **whole JSON files** as metadata

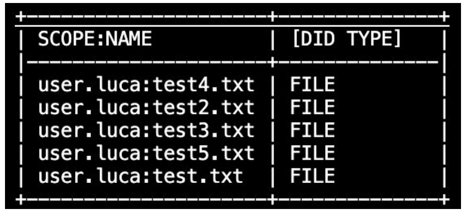
- Queries **AyraDB** via Rucio **without** needing **direct access**

customListDids:

- retrieve the **DIDs** satisfying user-defined filters

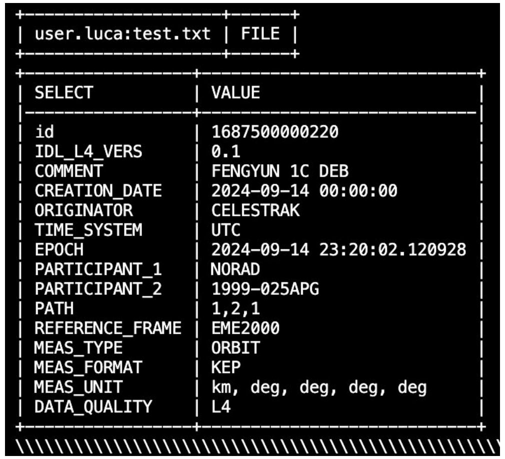
customQuery:

- retrieve a user-defined **list of metadata keys**



```
+----------------------+-------------+
| SCOPE:NAME           | [DID TYPE]  |
|----------------------+-------------|
| user.luca:test4.txt  | FILE        |
| user.luca:test2.txt  | FILE        |
| user.luca:test3.txt  | FILE        |
| user.luca:test5.txt  | FILE        |
| user.luca:test.txt   | FILE        |
+----------------------+-------------+
```

Output: customListDids

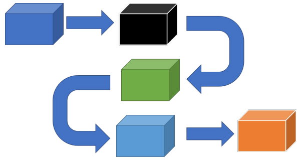


```
+---------------------+------+
| user.luca:test.txt  | FILE |
+---------------------+------+
+-----------------+------------------------------+
| SELECT          | VALUE                        |
|-----------------+------------------------------|
| id              | 1687500000220                |
| IDL_L4_VERS     | 0.1                          |
| COMMENT         | FENGYUN 1C DEB               |
| CREATION_DATE   | 2024-09-14 00:00:00          |
| ORIGINATOR      | CELESTRAK                    |
| TIME_SYSTEM     | UTC                          |
| EPOCH           | 2024-09-14 23:20:02.120928   |
| PARTICIPANT_1   | NORAD                        |
| PARTICIPANT_2   | 1999-025APG                  |
| PATH            | 1,2,1                        |
| REFERENCE_FRAME | EME2000                      |
| MEAS_TYPE       | ORBIT                        |
| MEAS_FORMAT     | KEP                          |
| MEAS_UNIT       | km, deg, deg, deg, deg       |
| DATA_QUALITY    | L4                           |
+-----------------+------------------------------+
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

Output: customQuery

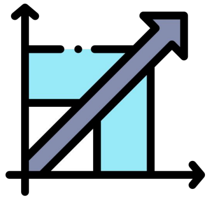
Future Plans: what's next?

- **End-to-End Testing**: Validate the solution using domain-specific datasets to meet real-world scenarios.



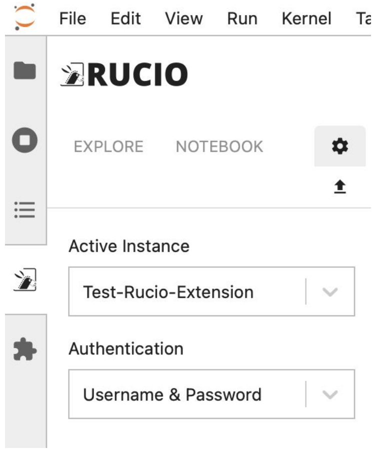
End to End Testing

- **Scalability Test**: Perform stress tests to evaluate the system's scalability as more data is ingested into the Data Lake and AyraDB.



Scalability Test

- **JupyterHub Integration**: Deploy a JupyterHub instance over a k8s cluster to enable seamless interaction between the Rucio based Data Lake and data analysis workflows



Active Instance

Test-Rucio-Extension

Authentication

Username & Password

Thank you for your attention!

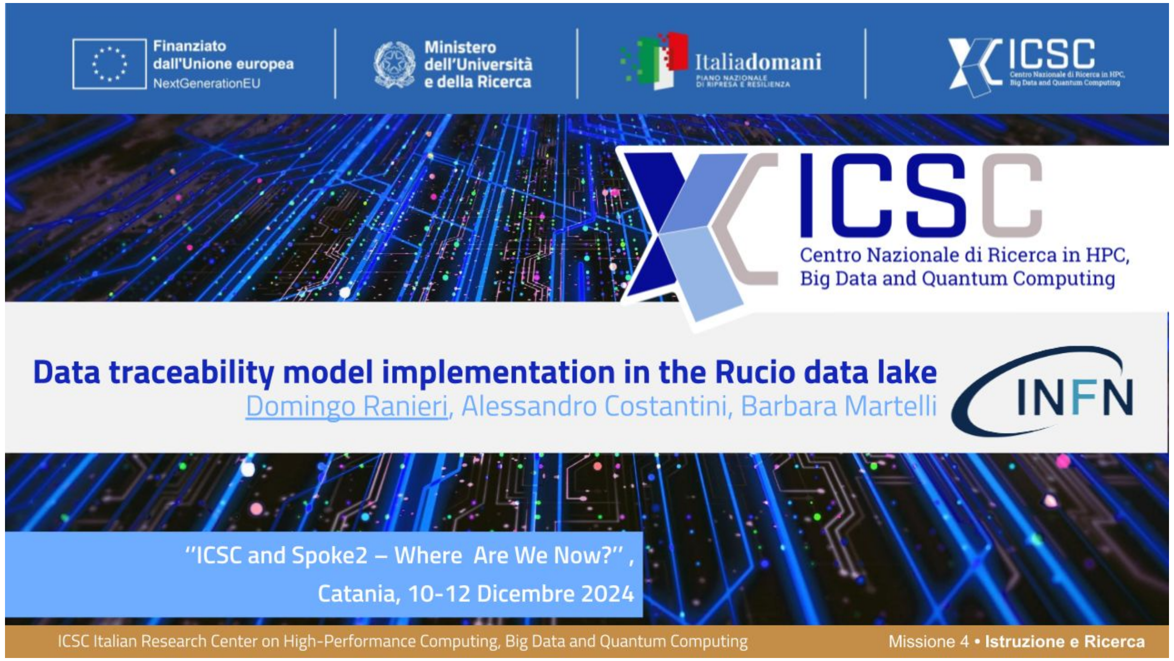
luca.pacioselli@pg.infn.it

Backup

Blockchain implementation

Blockchain functionality, in collaboration with Domingo Ranieri:

- **Hashes Storing**: blockchain stores SHA-256 hashes of data and metadata
- **Data Validation**: when file uploaded, hashes of data and metadata are stored in the blockchain
- **Verification Process**: when files retrieved, both metadata and data hashes are compared against blockchain-stored hashes



Finanziato dall'Unione europea NextGenerationEU | Ministero dell'Università e della Ricerca | Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA | ICSC

ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

Data traceability model implementation in the Rucio data lake

Domingo Ranieri, Alessandro Costantini, Barbara Martelli

INFN

"ICSC and Spoke2 – Where Are We Now?",

Catania, 10-12 Dicembre 2024

ICSC Italian Research Center on High-Performance Computing, Big Data and Quantum Computing — Missione 4 • **Istruzione e Ricerca**

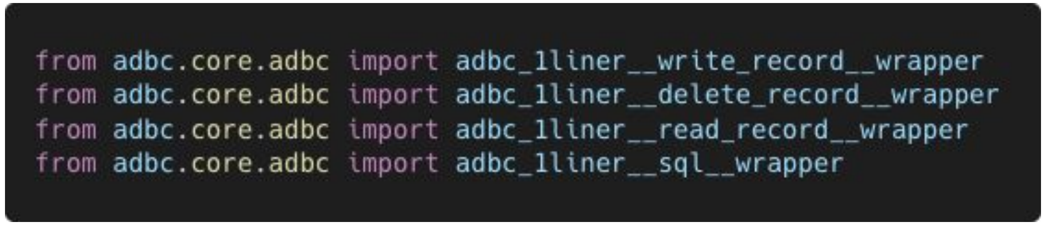
INFN PERUGIA
Istituto Nazionale di Fisica Nucleare
Sezione di Perugia

Custom DID-metadata plugin

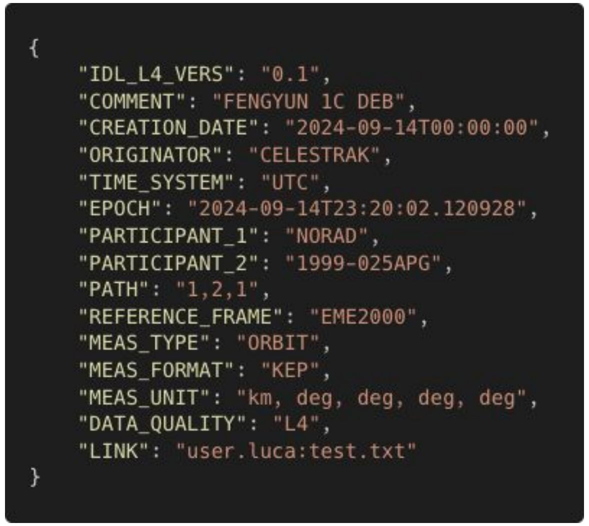
To **extend** the Rucio **metadata catalog** we designed a DID-metadata plugin that allows for external databases to be integrated for storing and querying metadata: AyraDB in our case

Functional methods for metadata handling are: set-, get-, delete-metadata and list-dids

The **AyraDB Connector** (ADBC) client, by CherryData, allows smooth implementation to perform operations like writing, deleting, or querying metadata:



```python
from adbc.core.adbc import adbc_1liner__write_record__wrapper
from adbc.core.adbc import adbc_1liner__delete_record__wrapper
from adbc.core.adbc import adbc_1liner__read_record__wrapper
from adbc.core.adbc import adbc_1liner__sql__wrapper
```

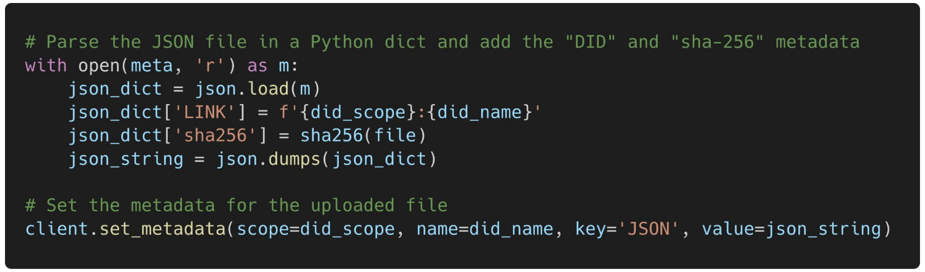


```json
{
    "IDL_L4_VERS": "0.1",
    "COMMENT": "FENGYUN 1C DEB",
    "CREATION_DATE": "2024-09-14T00:00:00",
    "ORIGINATOR": "CELESTRAK",
    "TIME_SYSTEM": "UTC",
    "EPOCH": "2024-09-14T23:20:02.120928",
    "PARTICIPANT_1": "NORAD",
    "PARTICIPANT_2": "1999-025APG",
    "PATH": "1,2,1",
    "REFERENCE_FRAME": "EME2000",
    "MEAS_TYPE": "ORBIT",
    "MEAS_FORMAT": "KEP",
    "MEAS_UNIT": "km, deg, deg, deg, deg",
    "DATA_QUALITY": "L4",
    "LINK": "user.luca:test.txt"
}
```

Example of IDL metadata

Dedicated Rucio client

- To ensure seamless communication between users, the plugin, and the blockchain, a dedicated **Rucio client** is needed
- Extended functionalities to upload, get-metadata, list-dids methods and new sql method to do custom queries
- **Python bindings** to include Rucio's default methods for a unified client experience



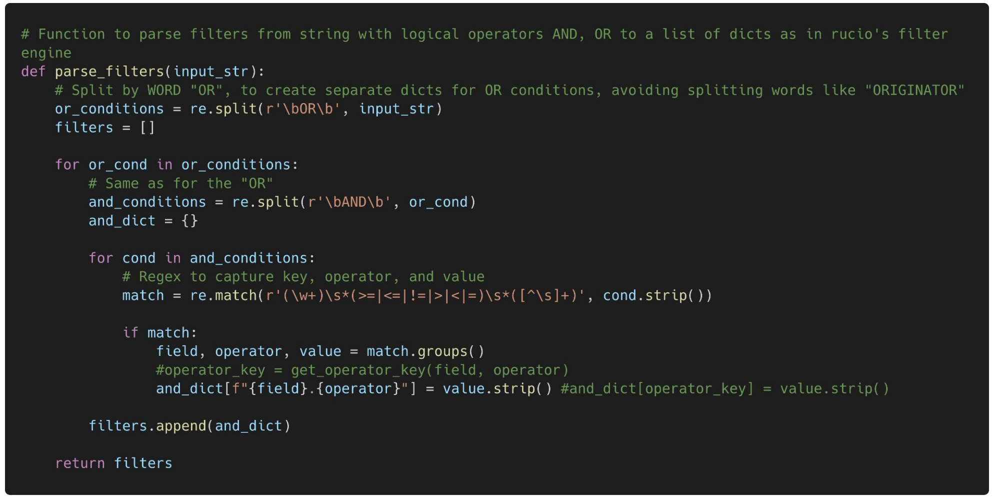
```python
# Parse the JSON file in a Python dict and add the "DID" and "sha-256" metadata
with open(meta, 'r') as m:
    json_dict = json.load(m)
    json_dict['LINK'] = f'{did_scope}:{did_name}'
    json_dict['sha256'] = sha256(file)
    json_string = json.dumps(json_dict)

# Set the metadata for the uploaded file
client.set_metadata(scope=did_scope, name=did_name, key='JSON', value=json_string)
```

customUpload:

- Rucio uploads data first, then sets metadata
- customUpload **combines** the two steps
- Handles specific integration needs with the plugin and AyraDB, e.g. passing **JSON file** content as a **whole**

Improves user experience and efficiency, ensuring smooth interaction with AyraDB

Dedicated Rucio client (cont.)

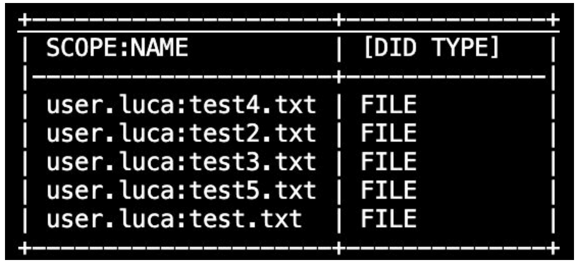


```python
# Function to parse filters from string with logical operators AND, OR to a list of dicts as in rucio's filter
engine
def parse_filters(input_str):
    # Split by WORD "OR", to create separate dicts for OR conditions, avoiding splitting words like "ORIGINATOR"
    or_conditions = re.split(r'\bOR\b', input_str)
    filters = []

    for or_cond in or_conditions:
        # Same as for the "OR"
        and_conditions = re.split(r'\bAND\b', or_cond)
        and_dict = {}

        for cond in and_conditions:
            # Regex to capture key, operator, and value
            match = re.match(r'(\w+)\s*(>=|<=|!=|>|<|=)\s*([^\s]+)', cond.strip())

            if match:
                field, operator, value = match.groups()
                #operator_key = get_operator_key(field, operator)
                and_dict[f"{field}.{operator}"] = value.strip() #and_dict[operator_key] = value.strip()

        filters.append(and_dict)

    return filters
```

Methods for Database Interaction:

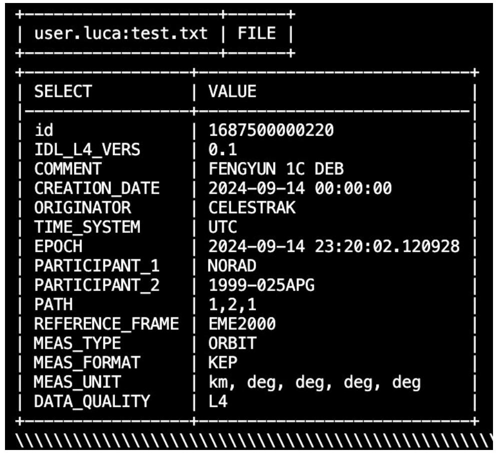
1. **customListDids** (extension of “rucio list-dids”)
   - Interrogates AyraDB through the plugin to retrieve a **list of DIDs** based on user-specified filters
2. **customQuery** (added functionality to Rucio)
   - Queries AyraDB via the plugin to retrieve a user-defined **list of keys** (DIDs are included by default)

Users can query **database queries through Rucio** to interact with the external database **without** needing **direct access**



```
+----------------------+--------------+
| SCOPE:NAME           | [DID TYPE]   |
|----------------------+--------------|
| user.luca:test4.txt  | FILE         |
| user.luca:test2.txt  | FILE         |
| user.luca:test3.txt  | FILE         |
| user.luca:test5.txt  | FILE         |
| user.luca:test.txt   | FILE         |
+----------------------+--------------+
```

Output: customListDids



```
+---------------------+------+
| user.luca:test.txt | FILE |
+---------------------+------+

+------------------+-----------------------------+
| SELECT           | VALUE                       |
|------------------+-----------------------------|
| id               | 1687500000220               |
| IDL_L4_VERS      | 0.1                         |
| COMMENT          | FENGYUN 1C DEB              |
| CREATION_DATE    | 2024-09-14 00:00:00         |
| ORIGINATOR       | CELESTRAK                   |
| TIME_SYSTEM      | UTC                         |
| EPOCH            | 2024-09-14 23:20:02.120928 |
| PARTICIPANT_1    | NORAD                       |
| PARTICIPANT_2    | 1999-025APG                 |
| PATH             | 1,2,1                       |
| REFERENCE_FRAME | EME2000                     |
| MEAS_TYPE        | ORBIT                       |
| MEAS_FORMAT      | KEP                         |
| MEAS_UNIT        | km, deg, deg, deg, deg      |
| DATA_QUALITY     | L4                          |
+------------------+-----------------------------+
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

Output: customQuery

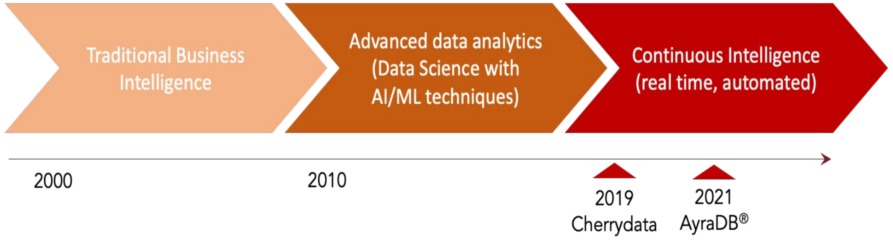
INFN PERUGIA
Istituto Nazionale di Fisica Nucleare
Sezione di Perugia

AyraDB as a metadata database for the “space debris” use case, objectives

Original slide by: Nicolò Magini (thanks!)

- In the IDL system, data are stored on a data lake, while **metadata** are stored on a **dedicated database**
- AyraDB (high-performance database designed by **Cherrydata**, **www.ayradb.com**) has been chosen as metadata DB
- The objective is to **maximise query performance** by executing SQL queries on the metadata database (AyraDB) and retrieving from the data lake only the requested data

Cherrydata is a startup (and a spinoff of PoliMi), offering consulting, innovation, and research services on big data and analytics.


Traditional Business Intelligence
Advanced data analytics (Data Science with AI/ML techniques)
Continuous Intelligence (real time, automated)
2000
2010
2019 Cherrydata
2021 AyraDB®

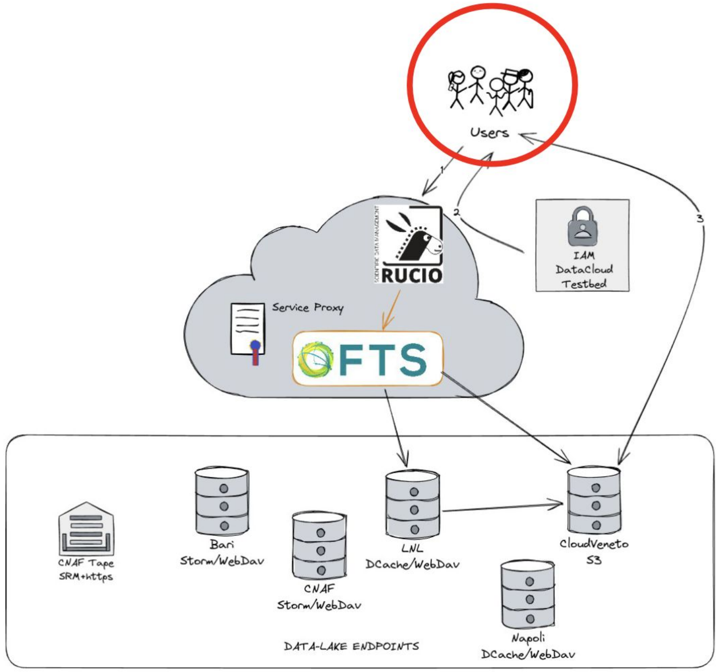
- *AyraDB has been tested on Leonardo Davinci-1 supercomputer in 2022, as part of Euro NCC project.*
- *Cherrydata is involved in IDL as technology provider, to test AyraDB in the context of storing and querying astrophysical data and satellite measurements.*

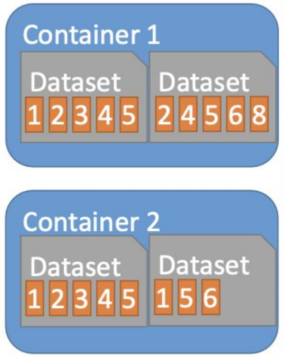
- The implementation of AyraDB has been designed to **minimise response time** to queries operating **on large tables**
- Preliminary tests have been performed on synthetic metadata (1 billion records)

Rucio in a nutshell

Data Management tool:

- manages data **transfers**, **deletions** and **storage**
- integrates with **many storage solutions**
- data can be stored across **multiple sites**, with diverse **setups** and **protocols**
- **data can be anything**: scientific observations, measurements, objects, events, images saved in files, etc…

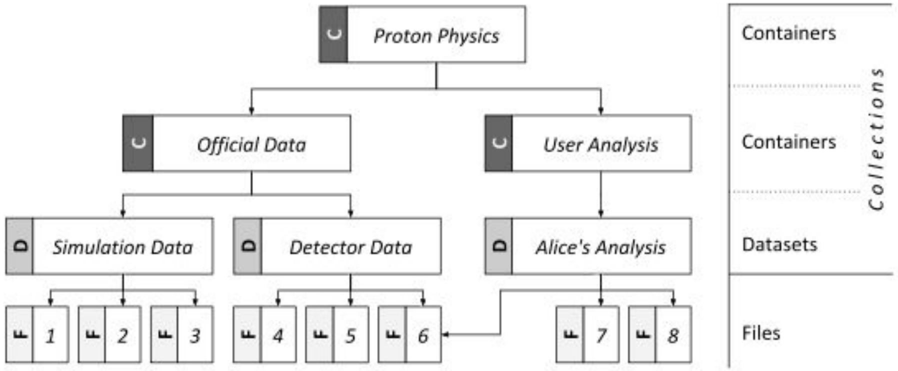




Rucio **Files**, **Datasets** and **Containers**:

- Files can be replicated using **rules**
- Files are grouped in **datasets** (can belong to multiple datasets)
- **Containers** are collections of datasets and containers

Rucio in a nutshell (cont.)



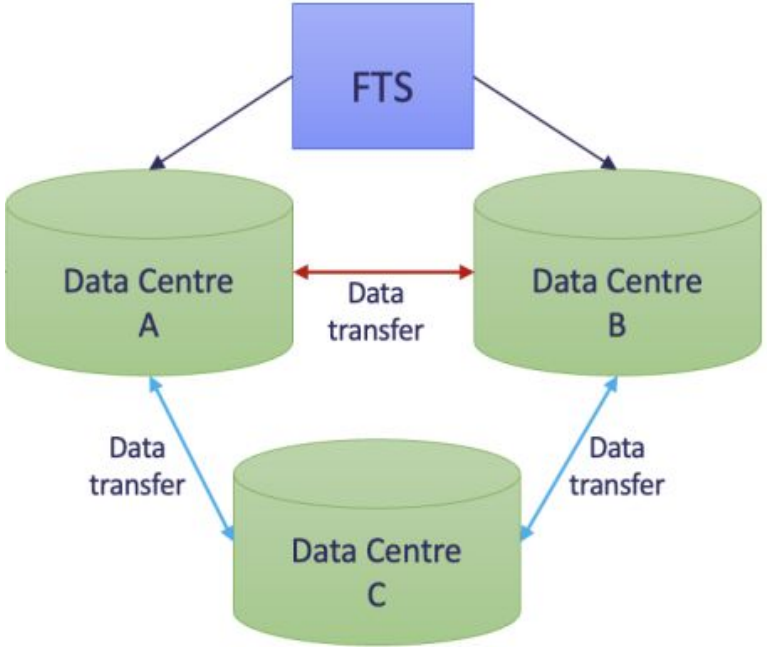
1. **Namespace** handling:
   - Data are organized using Data IDentifiers (**DIDs**)
     - three levels of granularity: files, datasets, and containers
   - DIDs are globally **unique**
     - **identified forever**: cannot reuse names of deleted DIDs
     - name change when data have been changed
   - Global namespace containing all DIDs can be partitioned into **scopes**
   - DIDs are always tuples **<scope>:<name>**
     - names are unique in a scope, whereas DIDs are globally unique

2. **Storage** abstraction:
   - Rucio associates physical location of DIDs with Rucio Storage Elements (**RSEs**)
   - RSE holds **all attributes** to **access** the **storage** space: hostname, port, protocol, and local file system path
   - No software needed to run at the data centers providing storage as RSE configs are in Rucio

Data transfer between multiple RSEs

FTS : File Transfer Service responsible for Bulk data movement

- Open-source software for large-scale queuing and reliable execution of file transfers
- Efficiently **schedules data transfers**
- **Maximizes** use of available **network** & **storage** resources whilst respecting any limits



WP1@IDL enhances the existing services

- to provide a seamless integration with external metadata
- to integrate datalake and compute enviroment

WP1@IDL in a nutshell

- **Test solutions for managing data in a geographically distributed environment (aka the DataLake)** by building end-to-end prototype and testbeds to demonstrate the capability to analyze the astrophysical observations and simulations available data in a cloud environment.

- **The Data Management capability**
  - Store/inject data (meant as files or data objects) in the DataLake
- **The data and compute integration for a effective processing and analysis**
  - deploy Platform as a Service (PaaS) services for the actual processing of the data ingested into the Datalake.

INFN PERUGIA
Istituto Nazionale di Fisica Nucleare
Sezione di Perugia