IVOA Interop. Meeting May 2023 - DAL I

Tuesday 9th May 2023 - 11:00 CEST (Bologna)

Participants (12): Joshua Fraustro (JF), Patrick Dowler (PD), François Bonnarel (FB), Markus Demleitner (MD), Sara Nieto (SN), Gilles Landais (GL), Omar Laurino (OL), Tamara Civera (TC), James Tocknell (JT), Mark Taylor (MT), Severin Gaudet (SG), Alberto Micol (AM)

Agenda

1. Python TAP implementation at MAST: Lessons Learned *(Joshua Fraustro)*
2. TAP-next: user-managed persistent tables *(Patrick Dowler)*
3. ADQL vector functions implementation in the Observatorio Astrofísico de Javalambre (OAJ) *(Tamara Civera)*
4. ADQL-next *(Grégory Mantelet)*

Python TAP implementation at MAST: Lessons Learned

- Implementation of TAP-1.1 as a Python microservice in replacement of a C# service
- Use FastAPI: very simple and fast API
    - Can filter parameters and input requests preventing users from errors
- It is about 6-7x faster for async. queries
- Difficult implementation of a standard from scratch:
    - "Where to start?"
    - Starting from scratch, it's difficult to approach the standard
    - The IVOA diagram not really useful when you want to start with a little/no knowledge about VO standards ; no really useful to implement TAP
    - Often IVOA standards don't agree or are ambiguous
    - Some clients expect some optional features being supported by services (e.g. `WAIT` in TAP-async/UWS)
    - Some useful information is lost (e.g. how to handle overflow in something else than VOTable)
    - UWS/TAP-async defines the `RUNID` param: but what is a valid `RUNID` value? Confusion between UWS, TAP and DALI

    - Adding links toward the precise section in the other standards would be more appreciated instead of just saying "see this or that standard"
    - A summary/cheatsheet of MUST, SHOULD and MAY for each service would be appreciated too.
    - Thanks to Mark Taylor for his TAPLint tool. Really useful to have a functional and compliant service.

TAP-next: user-managed persistent tables

- Suggestions based on the CANFAR services experience
- CANFAR services
    - few columns with unit
    - few columns with UCD ; possibly because of the difficulty to approach UCD
    - TAP service on GItHub
    - A docker image is also built
- API extension for TAP:
    - get/create/load/upload/drop table throw the youcat API `/youcat/tables/{table_name}`
    - the lod action is mainly to append rows to a table with input data in CSV, FITS, ...
    - possibility to index data POSTing at `/youcat/table-update`
    - endpoint for permissions at `/youcat/permissions/{name}`
- Open questions.concerns:
    - prototype with new endpoints:
        - TAP standardID with fragments?
        - a registry extension with <endpoints>, ...
    - extension of the VOSI-tables
        - registry extension?
        - modify VOSI?
        - extension in TAP?
        - a new standard?
    - pushing table metadata to the IVOA registry
        - table metadata are quite dynamic (update of the registry would then occure oftenly)
    - extends TAP_SCHEMA
        - about permissions and owners
- Other API asked features:
    - dynamic table metadata (renaming table or column)
    - multi-column index (not possible in TAP_SCHEMA)
    - explicit PRIMARY KEYs declaration
    - declare FOREIGN KEYs?
    - update: replace rows by PK?
    - delete rows with WHERE close?
- All these user-maganement features would be optional
- Should user-tables be published in the registry?

- Wishlist of additional features?

- [FB] Really ambiguous. Should it be rather TAP-2.0 instead of TAP-1.2?
    - [PD] These features do not break TAP-1.x. So, it should not be an issue for TAP-1.2.

- [MD] Definitely in favor of TAP-1.2.
    - Should be consistent with the current table declaration. Especially for the registry.
    - It would probably good to reject columns with no description.
    - Multi-index columns should be indeed visible. Ok for it.
    - Table management API through TAP standard is a bit dangerous. ESA for instance does it through VOSpace.
    - [SN] In the Gaia Archive, VOSpace connected to the archive, but the actual link with TAP is done via the ESA extension of TAP called TAP+.
        - Not yet information about permissions per row.
    - [GL] Connection with the registry: may be a difficulty to handle in a consistent way the dynamic declaration of table information.
        - [PD] Persons publishing table information should have the good procedure.
        - [GL] Add data origin information.

- [OL] What do you think about the concerns about the difficulties for implementing IVOA standards?
    - [PD] Links in just a click between standards using more precise sections would be indeed a good addition, but would be really possible with the HTML version. It would challenging. Otherwise, yes, give you chance to have a functional service but partly inconsistent and then update it to make it more consistent.
    - [MD] The idea to use the HTML version instead of the PDF one, was already an idea in the past of the IVOA. But apparently today, most people uses the PDF version. Maintaining such links is however hard.

ADQL vector functions implementation in the Observatorio Astrofísico de Javalambre (OAJ)

- Web portal: archive.cefca.es
    - VIsualization
    - TAP
    - Cone Search
    - ...
- Content: various images and tables
- TAP service with Python, Pyramid, Postgres

- Working with vectors since 2017
    - Used to stored objects photometry and flags
    - One measure by band
- Syntax based on Postgres:
    - `my_array[index]` with index starting from 1
    - enumerations with `enumeration::item` (way to assign set of names)
- Math operations using functions (e.g. `array_add`, `array_sub`, ...)
    - working only for float values
    - want to extend it for more datatypes
- Functions to get the max and min
- A function to search whether a value is in an array
- Some differences with the proposal from Jon Juaristi and Markus Demleitner in Oct. 2022
- Suggestion to support vector support to ADQL-2.2
    - As well as enumerations

- [MD] Arrays also important for me. Request: could you comment in DAL list about the differences between your proposal and mine? The idea would be to start a discussion on this topic
    - Question: intrigued about the enumeration. How the users find/discover your enumeration? How could that happen in TOPCAT?
    - [TC] We do not have a declaration of these enumerations yet.
    - [MD] TAPRegExt
    - [JF] What about SQL indexing enumeration?
    - [MD] Supported by Postgres.
    - [TD] Would it be possible to declare enumerations in table.
    - [MD] I'd rather not. Not good to spread information in different places. I'd suggest it should be in TAPRegExt.

ADQL-next

- PEG Grammar
- *OFFSET by default*
- *Boolean support*
- *Hexadecimal value + bitwise op.*
- *Array/Vector support*
- *Big numerics*
- *MOC support*
- *INTERVAL support*
- *Query by timestamp*
- *Unit annotation*
- *More string functions*
- Geometries
- Priorities for v2.2

- PEG
- INTERVAL support
- MOC support
- Array support
- SUBSTRING
- OFFSET

Many of the new suggested extensions aren't SQL related - should they be in ADQL or in another layer?

- Can we use pre-existing libraries?

[JT] Would it make sense to upgrade from SQL 92 to a more recent version as the base?

- [GM] Not against upgrading but would need buy in from other authors / implementors
- [MD] No strong opinions

[MD] Complex geometries - lets see how far we can get with MOCs instead

[MD] CAST INTERVAL - what would be the first parameter
[MD] What about array intervals?
[PD] Have implemented it as a constructir function but CASTing would most likely be from a string value or array of length 2

[AM] CASE expression in ADQL
[GM] No objections but would need to work out the syntax

[GM] Is everyone happy with the priorities?
[PD] Yes, but other favourite sting function - splitpart - there might be other useful ones too
[MT] PEG grammar - how far advanced is it and what is the motivation to change? Will this delay the next version of ADQL?
[GM] Already have a good starting point with a proposed PEG grammar - already have a validator - so in a good place and the advantage is it is magcine readable

[SG] Seeing requests for non-astro use cases - e.g. operations - so things like string functions are really useful