Toward TAP-1.2: User-managed Tables
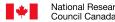
Patrick Dowler
Canadian Astronomy Data Centre

IVOA InterOp, November 2024

National Research Council Canada   Conseil national de recherches Canada

Canada

Overview

- use cases for user-created tables in TAP (Table Access Protocol)

- core functionality

- necessary functionality

- stretch goals

Use Cases

- upload temporary table to use (join) in one query: UPLOAD

- upload a temporary table to use in multiple queries – different scope

- upload a permanent table for private use – different lifetime

- … for use within a project – grant permissions

  ○ depends on AAI (authentication/authorization/identity)

  ○ depends on GMS (Group Membership Service)

- … and publish it for general use

  ○ registry implications

Core Functionality

- view table definition: GET /tables/{name}

- create table: PUT {description} -> /tables/{name}

- drop table: DELETE /tables/{name}

- update table: POST {description} -> /tables/{name}

  - mainly to fix/improve table and column metadata

- add rows / data

  - for small tables: PUT VOTable with rows

  - for large tables: append rows to table (scalability and robustness)

Core Functionality

- table description:
  - VOSI-table document (XML) – current core table model
  - VOTable document (XML) – alternative for user convenience

- update VOSI-tables with new methods (DELETE, POST, PUT)
  - most existing services: provide the base (read-only) tables endpoint
  - some services: provide the complete tables endpoint
  - OpenAPI: how to make the extended API optional?

Necessary Functionality

- create index (drop index)

  - youcat: this is an async job – UWS endpoint

- grant and revoke permissions

  - youcat: an allocation is a schema owned by a user

  - youcat: tap_schema extension (details available) + custom endpoint

- update TAP to specify index jobs and permissions (optional)

Stretch Goals

- enable service descriptor(s) for user tables
    - enable users to attach service descriptor(s) to column(s)
    - exploratory/prototype work in progress
- publish: make table visible to registry search
    - some QA (table and column metadata) necessary
    - add table description to appropriate registry records
    - need a plan to handle future updates to metadata
- probably implementation-specific… not part of standard

questions? comments?