GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

A Management Interface For Persistent TAP Uploads

Markus Demleitner

College Park IVOA Interop, June 2025

Outline

Previously on this topic

At the Malta Interop:

- Upload tables by PUT-ing to `<access url>/tap_user/<tablename>`,
- . . . or use the `CREATE TABLE` ADQL extension,
- Then use the table in the `tap_user` schema like any other.
- At least four open points:

Deferring destruction

The original implementation cleans up tables a week after they are uploaded.

What if you need the table for longer?

UWS to the rescue. We just copy its behaviour for job life times.

```
$ curl http://dc.g-vo.org/tap/user_tables/my_upload/destruction
2025-05-29T09:55:20Z
$ curl -F DESTRUCTION=2038-01-19T03:14:07Z \
  http://dc.g-vo.org/tap/user_tables/my_upload/destruction
2026-05-23T11:56:44Z
```

Alternative: Posting To The Table URI?

In a first design, I thought we could post to the table URL (rather than a child) with a DESTRUCTION parameter.

Other operations would use different parameters.

I rejected that design because:

1. Different operations give different responses; if users request multiple operations, which response would you give?
2. Having a GET URI to figure out the destruction time (or the indexes) is nice: you don't need to invent an alternative method for exposing such metadata to clients.

Creating Indexes: The Challenge

People *probably* need to be able to create indexes; even for tables with just a few 10’000s of rows, these can make a huge difference.

Let me illustrate the problem with an excerpt from the postgres documentation:

```
CREATE [ UNIQUE ] INDEX [ CONCURRENTLY ]
  [ [ IF NOT EXISTS ] name ] ON [ ONLY ] table_name [ USING method ]
    ( { column_name | ( expression ) }
    [ COLLATE collation ]
    [ opclass [ ( opclass_parameter = value [, ... ] ) ] ]
    [ ASC | DESC ] [ NULLS { FIRST | LAST } ] [, ...] )
    [ INCLUDE ( column_name [, ...] ) ] [ NULLS [ NOT ] DISTINCT ]
    [ WITH ( storage_parameter [= value] [, ... ] ) ]
    [ TABLESPACE tablespace_name ] [ WHERE predicate ]
```

Creating Indexes: Dead Simple API

Just POST to one or more column names in one `INDEX` parameter each to an `index` child resource:

```
$ curl -L -F INDEX=Kmag http://dc.g-vo.org/tap/user_tables/my_upload/index
$ curl -L -F INDEX=_RAJ2000 -F INDEX=_DEJ2000\
  http://dc.g-vo.org/tap/user_tables/my_upload/index
```

This redirects to a UWS job that runs the indexing procedure; the job is created in QUEUED.

Indexes: Debug API

A GET from this resource gives debug information (recommended: the concrete CREATE INDEX statements; TAP_SCHEMA/VODataService info on indexes isn't good enough for debugging:

```
$ curl -L http://dc.g-vo.org/tap/user_tables/my_upload/index
Indexes on table tap_user.my_upload

CREATE INDEX my_upload__RAJ2000__DEJ2000
  ON tap_user.my_upload (q3c_ang2ipix("_RAJ2000","_DEJ2000"))
CLUSTER my_upload__RAJ2000__DEJ2000 ON tap_user.my_upload
CREATE INDEX my_upload_Kmag ON tap_user.my_upload (Kmag)
```

This is supposed to be *human*-readable, *not* machine-readable: It's debug information after all.

Spatial Indexes

To use indexes in ADQL expressions like

```
distance(ra, dec, 14.2, -18.9)<0.2
```

you generally need custom sorts of indexes (cf. the `q3c_ang2ipix` expression above).

Users can't usually know which these are.

Best Effort

In my scheme, the service is asked to guess what a good index is. Rules on my end at this point:

- Column Type spoint: Do a GIST index
- Columns with a unit of degree and a matching pair of longitude and latitude by UCD: Do a spatial index (implementation detail: using q3c or pgsphere according to operator preference).

New Open Issues

- Indexing currently has the indexing job start in QUEUED. Would anyone ask for PENDING (i.e., editable)?
- Indexing UWS job: Do we expect people to destroy their *jobs* manually? Should we just advise very short destruction times for those?
- Spatial or “normal” index when a user asks for (galactic long, declination)?
- Are column references case-insensitive?
- What about delimited identifiers?

PR?

So. . . should I write a pull request with this?

See also a blog post on this:

.