

1. ivo ADQL UDFs in the Pipeline

Markus Demleitner
msdemlei@ari.uni-heidelberg.de

- Reminder: What this is about
- Two UDFs for UDF catalogue 1.1
- Six UDFs for UDF catalogue 1.2



Funded by e-inf-astro, BMBF FKZ 05A20VH5

Distributed under CC0



2. ADQL User Defined Functions

TAP operators can add functionality to their ADQL engines by providing extra functions. Their names should start with a provider-specific prefix, as in:

```
select top 1 gavo_to_mjd('2023-10-18T15:57')  
from tap_schema.tables  
[=60235.664583]
```

When multiple operators provide the same functionality, they should converge on a common name and use the ivo_ prefix.

The "Catalogue of ADQL User Defined Functions", now in Version 1.0 (Campillo and Demleitner 2021) lists these ivo_ UDFs.

3. Towards Version 1.1

Version 1.1 of the UDF cat is currently proposed (see <http://ivoa.net/documents/>). There are two new UDFs in there:

`ivo_epoch_prop_pos(ra, dec, parallax, pmra, pmdec,
radial_velocity, ref_epoch, out_epoch) -> POINT`

and

`ivo_histogram(val, lower, upper, nbins) -> INTEGER[]`

Please review and post to the DAL list if you have thoughts about this.

4. Towards Version 1.2

There are already several functions slated for version 1.2. Please chime in if you think they should be defined differently.

5. ivo_normal_random

`ivo_normal_random(mu REAL, sigma REAL) -> REAL`

The function returns a random number drawn from a normal distribution with mean mu and width sigma.

6. ivo_simbadpoint

`ivo_simbadpoint(identifier TEXT) -> POINT`

gavo_simbadpoint queries simbad for an identifier and returns the corresponding point. Note that identifier can only be a literal, i.e., as simple string rather than a column name.

e.g., `ivo_simbadpoint('GJ 699')`
`-> POINT(269.452076958619, 4.69336496657667)`

7. ivo_to_jd, ivo_to_mjd

`ivo_to_jd(d TIMESTAMP) -> DOUBLE PRECISION`

`ivo_to_mjd(d TIMESTAMP) -> DOUBLE PRECISION`

The functions converts database timestamps to (modified) julian dates. This is naive; no corrections for timezones, let alone time scales or the like are done. You can thus not expect this to be good to second-precision unless you are careful in the construction of the timestamp. This is so you can do date computations even if the table schema (regrettably) has timestamps.

8. ivo_transform

`ivo_transform(from_sys TEXT, to_sys TEXT, geo GEOMETRY) -> GEOMETRY`

The function transforms ADQL geometries between various reference systems. geo can be a POINT, a CIRCLE, or a POLYGON, and the function will return a geometry of the same type. From_sys and to_sys must be literal strings. Reference frame names are case-sensitive and must be taken from the IVOA reframe vocabulary.

9. ivo_epoch_prop

```
ivo_epoch_prop(ra DOUBLE PRECISION, dec DOUBLE PRECISION,  
  parallax DOUBLE PRECISION, pmra DOUBLE PRECISION,  
  pmdec DOUBLE PRECISION, radial_velocity DOUBLE PRECISION,  
  ref_epoch DOUBLE PRECISION, out_epoch DOUBLE PRECISION)  
-> DOUBLE PRECISION[6]
```

Returns a 6-vector of (ra, dec, parallax, pmra, pmdec, rv) at out_epoch for these quantities at ref_epoch. Units on input and output are degrees for ra and dec, mas for parallax, mas/yr for pmra and pmdec, and km/s for the radial velocity. ref_epoch and out_epoch are given in Julian years. parallax, pmra, pmdec, and radial_velocity may be None and will enter the computations as 0 then, except in the case of parallax, which will be some small value.

10. Your Turn!

Feedback on all of these is welcome on the DAL list (or perhaps as a bug on <https://github.com/ivoa-standard/udf-catalogue>).

If you have useful UDFs in your TAP service and want to promote it to ivo.: Talk to me; perhaps I'll be your second implementation!

Thanks!