ADQL-2.1 Status

Grégory Mantelet[1]

[1]CDS (Centre de Données astronomiques de Strasbourg)

7th May 2020

Table of Contents

1. ADQL-2.1 Status

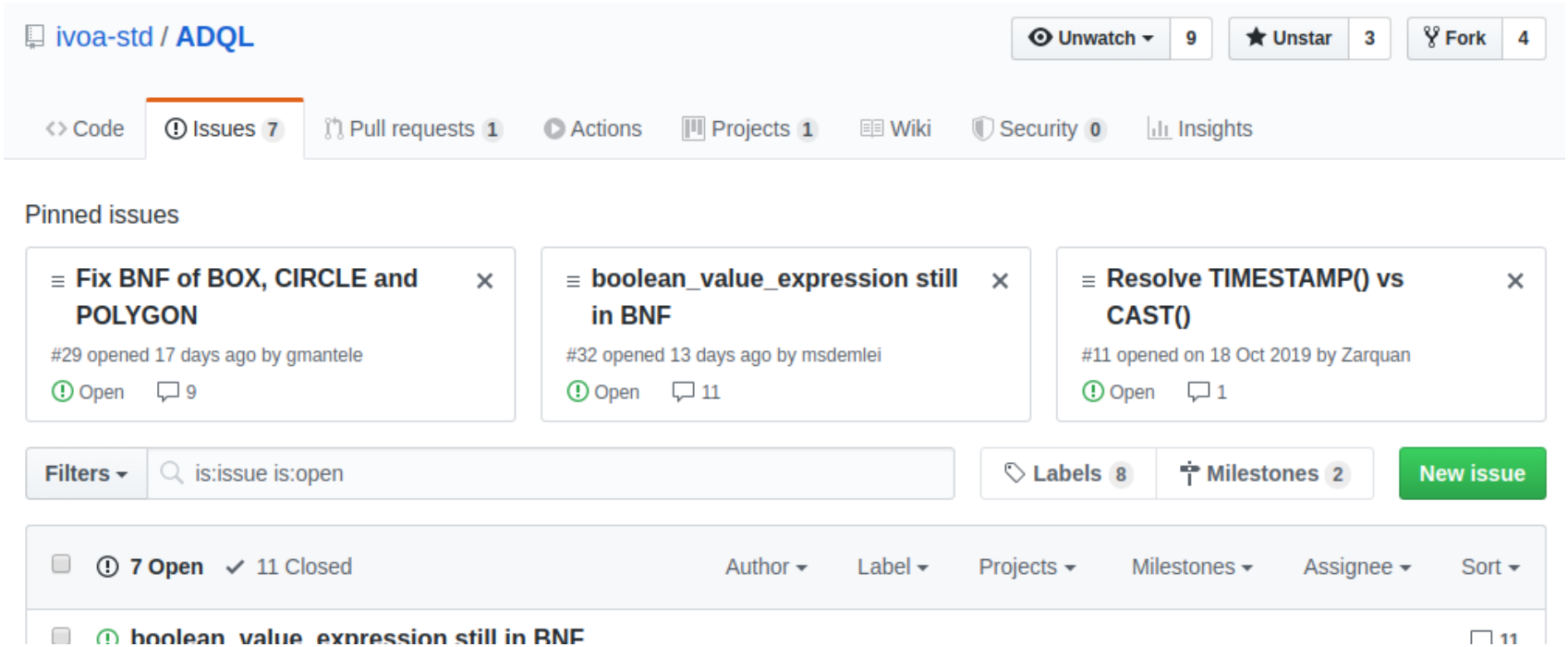
1. ADQL-2.1 Status

- ADQL-2.1 is in **PR since January 2018** *(1st working draft: May 2016)*
- **Goal:** to go RFC (then REC) once the last remaining issues are fixed
- **Status:** few issues left
- ***Note for the next versions of ADQL:*** *try to make less modification but release more often.*

2. GitHub usage

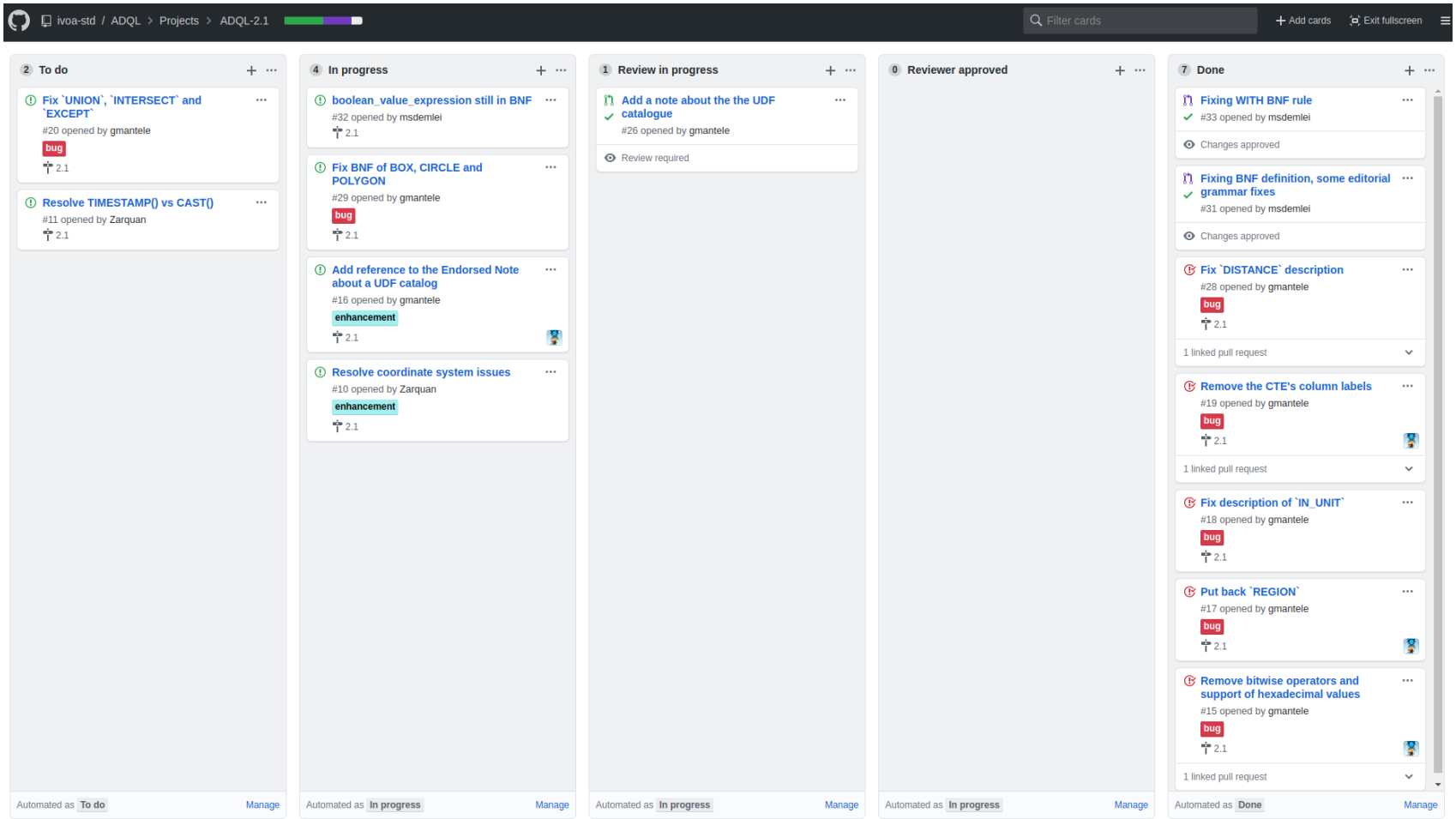
2.a. Issues

- Pinned issues



2.b. Project

- Project “ADQL-2.1”



3. What is done

3.a. IN_UNIT - #18

- No conversion from or into unitless values
- Optional *unit inference mechanism* for complex expressions

Correct

```
IN_UNIT(ra+ra_error, 'rad')
IN_UNIT(ra+10, 'rad')
IN_UNIT(sqrt(power(pm_ra,2)+power(pm_dec,2)), 'rad/yr')
```

Incorrect

```
IN_UNIT(ra, '')
IN_UNIT(phot_cnt, 'deg')
IN_UNIT(ra, 'foo')
IN_UNIT(ra, 'kg')
IN_UNIT(42, 'kg')
IN_UNIT(40+2, 'deg')
IN_UNIT(sqrt(42), 'deg')
```

3.b. DISTANCE - #28

- Only between 2 POINT values
- POINT values have been extended to CENTROID and UDFs
- In a future version of ADQL, possibly between other geometries

```
DISTANCE(POINT(ra, dec) ,  POINT(0., 0.)) -- OK (2.0)
DISTANCE(   my_point    ,   alien_point ) -- OK (2.0)
DISTANCE(   ra   ,  dec   ,    0.   ,  0.   ) -- OK (2.1+)
DISTANCE(my_udf('12 23'), CENTROID(...) ) -- OK (2.1+)

DISTANCE(POINT(ra, dec),
         CIRCLE(ra_target, dec_target, 1./60.))
         -- Not yet!
```

3.c. Misc

- Remove bitwise operators and hexadecimal values - *#15*
- Remove CTE's column labels *(e.g. `WITH (lblCol1, lblCol2) AS SELECT ...`)* - *#19*
- Fix BNF grammar literals: meta-symbol characters like : must be double quoted - *#31*
- ...

4. Work in progress

4.a. Boolean values - #32

- **Idea:** simplify the syntax

```
WHERE 1=CONTAINS(POINT(ra, dec), CIRCLE(12, 23, 1/60.))
WHERE  CONTAINS(POINT(ra, dec), CIRCLE(12, 23, 1/60.))
```

- **Problem:**
  - Boolean equivalent to a numeric (i.e. `True`=1 and `False`=0)
  - A condition is no longer only a comparison but can be a (which can be a lot of things. . . and most of the time, not boolean)
  - No explicit grammar rule restricting the allowed expression
    - need for a smarter parser able to infer the datatype and throw an error if not boolean
  - *Note: From ADQL-2.1, `CONTAINS` is no longer the recommended xmatch syntax*

4.a. Boolean values - #32

- **Proposed solutions:** *(details on GitHub)*
  1. Try *harder* to make it work
  2. Keep the boolean type but force `= True` or `= False` (instead of =1 and =0)
  3. Keep the boolean type but no numeric equivalence
  4. No boolean at all *(for the moment)*
  5. No boolean but `CONTAINS` and `INTERSECTS` becomes predicate valued functions *(i.e. the =1 or =0 is no longer required)*
- **Status:** *still under debate*

4.b. Coord. transform - #10

- **Problem:**
  - Deprecation of the coord. sys. argument of geometries.
  - Some services (e.g. GAVO, VizieR) can use this parameter to perform coordinates transform.
  - *How can they keep doing that in ADQL-2.1+?*
- **Proposed solution:**
  - No addition to ADQL itself.
  - Rely on UDFs listed in the Catalogue of UDFs : `gavo_transform(from_sys, to_sys, geo)`
  - If enough implemented renamed into `ivo_`
- **Status:**
  - Issue will be closed when the UDF Catalogue will become *EN (currently RFC)*

4.c. Geometries arg. - #29

- **Problem:**
  - The coord. sys. argument is also now optional
  - ADQL-2.1 allows a 2nd signature for `CIRCLE` and `POLYGON`:

```
CIRCLE(  12   ,  23   , 1/60.) -- ADQL-2.0
CIRCLE(POINT(12, 23), 1/60.) -- ADQL-2.1+
CIRCLE(   my_point   , 1/60.) -- ADQL-2.1+
```

  - OK for `CIRCLE`, **but not** for `POLYGON`

Example with POLYGON

```
POLYGON(colCooSys,
        colRa1, colDec1,
        colRa2, colDec2,
        colRa3, colDec3)
POLYGON(colRa1, colDec1,
        colRa2, colDec2,
        colRa3, colDec3)
POLYGON(colPoint1,
        colPoint2,
        colPoint3)
-- but also:
POLYGON(colPoint1,
        colPoint2, colPoint3,
        colPoint4, colPoint5,
        colPoint6, colPoint7)
-- which is ambigus with the example with colCooSys !!!!
```

4.c. Geometries arg. - #29

- **Real problem:** the coord. sys. argument can be a string literal but also a column ref. or a UDF

  => ambiguity with any of the other argument

- **Proposed solution:**
  - Erratum to ADQL-2.0 to make the coord. sys. argument *just* a literal
- **Status:**
  - *Would this Erratum break an existing service?*
    - If no, just do it. Then, the problem with `POLYGON` is automatically solved.

5. To do

5. To do

- **UNION, INTERSECT, EXCEPT** - *#29*
  - basically *just* grammar fix
- **CAST vs Constructors** - *#11*

CAST	Constructor
`CAST('2020-05-07', TIMESTAMP)`	`TIMESTAMP('2020-05-07')`
`CAST(30, SMALLINT)`	`SMALLINT(30)`
`CAST('23 42', POINT)`	`POINT(23, 42)`

6. After ADQL-2.1...

6. After ADQL-2.1...

- ~~BNF~~ -> PEG grammar
- Geometries:
  - *see OGC standard (OpenGIS® Implementation Standard for Geographic information) for ADQL evolution*
  - `DISTANCE` between other geometries than `POINT`
  - complex regions defined with intersections
- Support of arrays
- Support of Interval *(see DALI)*
- Support of MOC
- Support of the boolean data-type
- Hexadecimal notation + Bitwise operations
- Unit annotation