



**STScI** | SPACE TELESCOPE  
SCIENCE INSTITUTE

EXPANDING THE FRONTIERS OF SPACE ASTRONOMY

# Integrating PyVO with the NAVO Registry

---

Tom Donaldson, Tessa Dower

IVOA Interop – Northern Fall 2022



# Introduction

Context: Working on Jupyter Notebooks for the [NAVO workshop](#)

- Mostly relies on PyVO for queries
- Most exercises begin with registry queries to find relevant resources

```
services = vo.regsearch(servicetype='scs', keywords=['zcat'])
services.to_table()['ivoid', 'short_name', 'res_title']
```

Table length=6

<b>ivoid</b>	<b>short_name</b>	<b>res_title</b>
<b>object</b>	<b>object</b>	<b>object</b>
ivo://cds.vizier/j/mnras/339/652	J/MNRAS/339/652	The FLASH Redshift Survey
ivo://cds.vizier/vii/259	VII/259	6dF galaxy survey final redshift release
ivo://nasa.heasarc/sixdfgs	SIXDFGS	6dFGS Galaxy Survey Final Redshift Release Catalog
ivo://nasa.heasarc/uzc	UZC	Updated Zwicky Catalog
ivo://nasa.heasarc/zcat	CFAZ	CfA Redshift Catalog (June 1995 Version)
ivo://wfau.roe.ac.uk/twompz-dsa		2MASS Photometric Redshift catalogue (2MPZ)



## Introduction (2)

---

PyVO (and Topcat) use the GAVO registry by default

- <http://reg.g-vo.org/tap>
- Has been extremely reliable and works seamlessly with PyVO

Would be interesting to try using the NAVO registry

- Registry is a key component, so good to establish some redundancy
- PyVO uses RegTAP, so should work with other registries including NAVO's
- What will we learn about our service? 🙈
- A work in progress...

These slides summarize the issues encountered and thoughts provoked along the way.



## Using a Non-Default Registry in PyVO

---

Via `IVOA_REGISTRY` environment variable

```
export IVOA_REGISTRY=http://vao.stsci.edu/RegTAP/TapService.aspx
```

Works fine but requires restarting Python to change value

- `pyvo.registry.regtap.REGISTRY_BASEURL` is set on module load.
- Service instance is cached:

```
@functools.lru_cache(1)
def get_RegTAP_service():
    """
    a lazily created TAP service offering the RegTAP services.

    This uses regtap.REGISTRY_BASEURL. Always get the TAP service
    there using this function to avoid re-creating the server
    and profit from caching of capabilities, tables, etc.
    """
    return tap.TAPService(REGISTRY_BASEURL)
```

● Add direct support in API for setting `IVOA_REGISTRY` programmatically





## Find Other RegTAP Services



---

- ? What is the best way to find other RegTAP services?
  - Could be useful for both manual and automated compatibility testing
  - 'regtap' keyword search in Topcat → only NAVO/MAST registry
  - 'regtap' keyword search in PyVO → 3 GAVO
    - includes 2 mirrors, but not <http://reg.g-vo.org/tap> which redirects to main GAVO
      - ▶ ? Does the redirect switch to a mirror during downtime on the main server?
  - Neither search found "EURO-VO Registry TAP"



## PyVO Required Hard outputLimit

---

- A TAP service's `/capabilities` may include an `<outputLimit><hard>` value.
  - That value is not required and is not supplied by NAVO RegTAP
  - PyVO raised an exception if that value wasn't present
  -  Filed [PyVO issue](#) which was rapidly fixed by [Markus' PR](#)
-  Reraised general issue: Some property accessors do web requests as side-effect
  - `hardlimit` property triggers a `/capabilities` query
  - Same for all capabilities values
  - Also happens for `job` and `phase` properties in async TAP queries (see [separate PR comment](#))



## STScI Firewall Issues

---

- STScI's firewalls were blocking some RegTAP requests, both to MAST and GAVO
  - See DAL 1 talk, [ADQL and Firewall SQL-Injection Detection](#)
  - Registry endpoints have mostly been fixed
    - But just today we saw another request blocked



## COALESCE?

The MAST ADQL parser raised an error, not recognizing COALESCE

- Handling COALESCE is required for ADQL 2.0
- What query is PyVO using? Use `get_RegTAP_query()` with same args as `regsearch()`:

```
from pyvo.registry.regtap import get_RegTAP_query
query_text = get_RegTAP_query(servicetype='image', keywords=['allwise'])
print(query_text)
```



```
SELECT
  ivo_id, res_type, short_name, res_title, content_level, res_description, reference_url, creator_seq, content_type
  ivo_string_agg(COALESCE(access_url, ''), ':::py VO sep:::') AS access_urls,
  ivo_string_agg(COALESCE(standard_id, ''), ':::py VO sep:::') AS standard_ids,
  ivo_string_agg(COALESCE(intf_type, ''), ':::py VO sep:::') AS intf_types,
  ivo_string_agg(COALESCE(intf_role, ''), ':::py VO sep:::') AS intf_roles
FROM
  rr.resource
NATURAL LEFT OUTER JOIN rr.capability
NATURAL LEFT OUTER JOIN rr.interface
WHERE
  (standard_id IN ('ivo://ivoa.net/std/sia'))
  AND (ivo_id IN (SELECT ivo_id FROM rr.resource WHERE 1=ivo_hasword(res_description, 'allwise') UNION SELECT ivo_id
GROUP BY
  ivo_id, res_type, short_name, res_title, content_level, res_description, reference_url, creator_seq, content_type
```






## COALESCE? (2)

---

Handling COALESCE is not required until RegTAP 1.2 ([in Working Draft](#))

### MAST uses SQL Server

- Relies on parser and translator from Grégory Mantelet (CDS) and other contributors
  - See [CDS TAP Library](#) and the [code base on github](#)
  - Multiple SQL dialects are supported at various levels
- MAST wrapped the translator in a web service that can be run independently from TAP
-  Pulling upstream changes for the parser fixed COALESCE

### ? When to have future features in PyVO?

- Important to exercise implementation \*prior\* to REC
- Obviously breaks interoperability with some services
- Even after REC, how soon should PyVO assume service compliance?



## Error on ivo\_string\_agg()

- SQL Server error executing ivo\_string\_agg()
  - ✓ Fixed with a configuration change to SQL Server
  - ✓ Added that call to regression tests

```
SELECT
  ivo_id, res_type, short_name, res_title, content_level, res_description, reference_url, creator_seq, con
  ivo_string_agg(COALESCE(access_url, ''), '::::py V0 sep::::') AS access_urls,
  ivo_string_agg(COALESCE(standard_id, ''), '::::py V0 sep::::') AS standard_ids,
  ivo_string_agg(COALESCE(intf_type, ''), '::::py V0 sep::::') AS intf_types,
  ivo_string_agg(COALESCE(intf_role, ''), '::::py V0 sep::::') AS intf_roles
FROM
  rr.resource
NATURAL LEFT OUTER JOIN rr.capability
NATURAL LEFT OUTER JOIN rr.interface
WHERE
  (standard_id IN ('ivo://ivoa.net/std/sia'))
  AND (ivo_id IN (SELECT ivo_id FROM rr.resource WHERE 1=ivo_hasword(res_description, 'allwise') UNION SE
GROUP BY
  ivo_id, res_type, short_name, res_title, content_level, res_description, reference_url, creator_seq, con
```



## ADQL Error for Union

```
l(res description, 'allwise') UNION SELECT ivoid FROM rr.resource WHERE  
tion, reference url, creator seq, content type, source format, source v
```

Exception translating ADQL: Encountered "UNION". Was expecting one of: ")" "AND" "OR"  
"GROUP" "HAVING" "ORDER"

(HINT: "UNION" is not supported in ADQL, but is however a reserved word. To use it as a  
column/table/schema name/alias, write it between double quotes.)

? Hint suggests **UNION** is not required. Is that true?

- Is in the grammar as a SQL reserved word
- Will be explicitly added in [optional section 4 in ADQL 2.1](#) (in Proposed Recommendation)
  - [https://ivoa.net/documents/ADQL/20210528/PR-ADQL-2.1-20210528.html#tth\\_sEc4.6.1](https://ivoa.net/documents/ADQL/20210528/PR-ADQL-2.1-20210528.html#tth_sEc4.6.1)
- ? Again, when to have future features in PyVO? Optional features?



## Summary of NAVO Issues

---

- STScI's firewalls blocking some RegTAP requests
- ✓ The MAST ADQL parser raised an error, not recognizing COALESCE
- ✓ SQL Server error executing `ivo_string_agg()`
- ADQL Error for Union



## Summary of Other Issues and Questions

---

### New PyVO issues

- ● Add direct support in API for setting `IVOA_REGISTRY` programmatically
  - ✓ PyVO Required Hard outputLimit
  - ● General issue: Some property accessors do web requests as side-effect
  - ● ADQL Error for Union
- ➔ ● Add multiple registries to pyvo CI test suite

### Questions

- ? What is the best way to find RegTAP services?
  - ? Does the GAVO redirect switch to a mirror during downtime on the main server?
- ? Is `UNION` is required?
- ? When to have future features in PyVO? Optional features?
  - Important to exercise implementation \*prior\* to REC
  - Obviously breaks interoperability with some services
  - Even after REC, how soon should PyVO assume service compliance?
  - If an optional feature is really needed or really nice, should it be required?