Applausequery

A PyVO application for highlevel access to astronomical photoplate database

Christian Dersch

18th Sep 2019

Philipps-Universität Marburg

What is APPLAUSE? I

- **A**rchives of **P**hotographic **PL**ates for **A**stronomical **USE**
- Archive of about 85000 photographic plates from Bamberg, Hamburg, Potsdam and Tartu

What is APPLAUSE? II

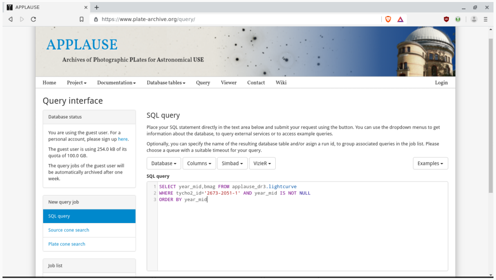
The archive:

- Provides a complete set of database tables containing:
  - metadata
  - calibration process
  - photometry
  - lightcurves
- Data Release 3 (fall 2018)
- Detailed information on whole plate processing available for each measurement
- Provenance documented in detail
- https://www.plate-archive.org/applause/

Intentions

- APPLAUSE does a *very* good job in data publication and access
- I want to support it by providing a polished version of the data access scripts I use, as it is the essential data source for my phd thesis
- Other services (no name dropping here) are harder to use, e.g. only web interface available
- Show them that VO, especially TAP is nice and that PyVO provides a foundation to build service specific packages to support the scientists

How to access APPLAUSE?



Query interface

Database status

You are using the guest user. For a personal account, please sign up here.

The guest user is using 254.0 kB of its quota of 100.0 GB.

The query jobs of the guest user will be automatically archived after one week.

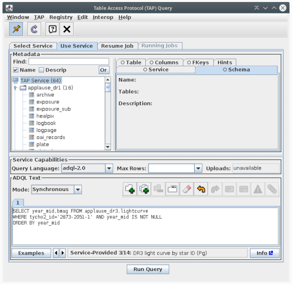
SQL query

Place your SQL statement directly in the text area below and submit your request using the button. You can use the dropdown menus to get information about the database, to query external services or to access example queries.

Optionally, you can specify the name of the resulting database table and/or asign a run id, to group associated queries in the job list. Please choose a queue with a suitable timeout for your query.

SQL query

```
SELECT year_mid,bmag FROM applause_dr3.lightcurve
WHERE tycho2_id='2673-2051-1' AND year_mid IS NOT NULL
ORDER BY year_mid
```



Name [✓] Descrip [ ]

Name:

Tables:

Description:

Query Language: adql-2.0
Max Rows:
Uploads: unavailable

Mode: Synchronous

```
SELECT year_mid,bmag FROM applause_dr3.lightcurve
WHERE tycho2_id='2673-2051-1' AND year_mid IS NOT NULL
ORDER BY year_mid
```

Service-Provided 3/14: DR3 light curve by star ID (Pg)

- APPLAUSE provides a nice web interface and a TAP service
- For explorative data analysis TOPCAT is a nice tool
- On the other hand: Python with astropy stack is the environment of choice for astronomical data analysis
- PyVO provides VO access integrated in astropy universe

Why applausequery

- Essentially started as a set of scripts to get my scientific work done
- My research topic
    - Analysis of lightcurves on longer time scales
    - Changes in lightcurve parameters over time
    - Combination with modern CCD data (e.g. ASAS-SN and ZTF)
- Other scientists working with APPLAUSE data might have similar requirements

My (scientific) requirements

- Easy way to get the data, “Load V-band lightcurve for star UCAC4 104-010297”
- Access to calibration information for single measurements, e.g. color term of used plate or plate scans
  - To discuss outliers caused by different emulsions, scratches etc.
  - To recalibrate if neccessary, e.g. when combining with data in other passbands than B or V (e.g. SDSS filters or Gaia passbands)
- Many quite similar database queries again and again
- Idea: Develop a Python package to abstract these queries and integrates with the other astropy packages: **applausequery**
- Inspired by *astroquery*

PyVO

- “An Astropy affiliated package providing access to remote data and services of the Virtual observatory (VO) using Python”
- Therefore usable for all services providing VO compliant access
- Supports TAP, SIA, SSA, SCS and SLAP
- In case of applausequery: TAP
- https://github.com/astropy/pyvo

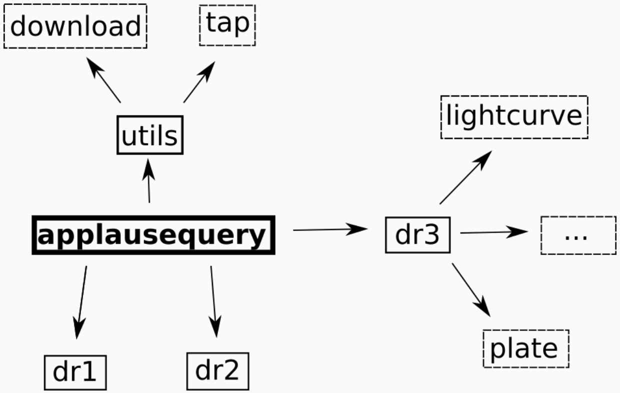
Package structure I - requirements

- Map database structure into package structure
- Abstract the underlying query language (SQL) as much as possible
- Give answers to questions common to the data
  - “Load V-band light curve for star UCAC4 104-010297”
- Provide a way to access TAP directly, so user can work with specific queries without having to handle PyVO directly
- Functions to download non table products, e.g. plate scans (FITS files) or logbook scans
- Authentication token support to have queries in personal space
- Reproducibility
  - Handle the data releases seperately

Package structure II

- One subpackage per data release, e.g. *applausequery.dr3* for DR3
  - submodules, e.g. lightcurve, scan, logbook
- *utils* subpackage
  - *ApplauseTAP* class (inherits PyVO TAPService, enhancing it with Authentication token support)
  - Generic functions not specific to a data release
- Use astropy package structure (by using their package template)
- BSD license, as most astronomical Python packages

Package structure - overview



How does a query function look like?

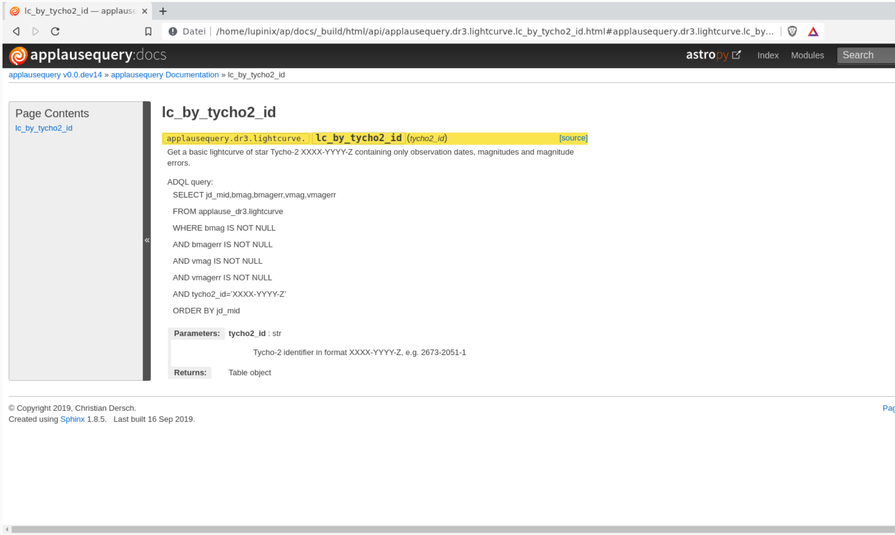
```
def lc_by_tycho2_id(tycho2_id):
# docstring left out here to save space
    query = "SELECT jd_mid,bmag,bmagerr,vmag,vmagerr \
        FROM applause_dr3.lightcurve \
        WHERE bmag IS NOT NULL \
        AND bmagerr IS NOT NULL \
        AND vmag IS NOT NULL \
        AND vmagerr IS NOT NULL \
        AND tycho2_id=\'%s\' \
        ORDER BY jd_mid" %(tycho2_id)
    lc = tap_session.run_async(query)
    return lc.to_table()
```

Quite simple: Specify query and call run_async() function of PyVO TAPService instance

Documentation

- Follow astropy documentation style
- Set of examples (partly based on examples at APPLAUSE website)
- API documentation
- Include ADQL/SQL queries in documentation
- https://applausequery.readthedocs.io (not yet online)

Documentation – Example for previously shown function


lc_by_tycho2_id — applause
Datei
/home/lupinix/ap/docs/_build/html/api/applausequery.dr3.lightcurve.lc_by_tycho2_id.html#applausequery.dr3.lightcurve.lc_by...
applausequery:docs
astropy
Index
Modules
Search
applausequery v0.0.dev14 » applausequery Documentation » lc_by_tycho2_id
Page Contents
lc_by_tycho2_id

lc_by_tycho2_id

`applausequery.dr3.lightcurve.` **`lc_by_tycho2_id`** (*tycho2_id*) [source]

Get a basic lightcurve of star Tycho-2 XXXX-YYYY-Z containing only observation dates, magnitudes and magnitude errors.

ADQL query:

```
SELECT jd_mid,bmag,bmagerr,vmag,vmagerr

FROM applause_dr3.lightcurve

WHERE bmag IS NOT NULL

AND bmagerr IS NOT NULL

AND vmag IS NOT NULL

AND vmagerr IS NOT NULL

AND tycho2_id='XXXX-YYYY-Z'

ORDER BY jd_mid
```

Parameters: **tycho2_id** : str

Tycho-2 identifier in format XXXX-YYYY-Z, e.g. 2673-2051-1

Returns: Table object

© Copyright 2019, Christian Dersch.
Created using Sphinx 1.8.5. Last built 16 Sep 2019.

Development and Installation

- On GitHub https://github.com/lupinix/applausequery
- Focus on DR3 for now, older releases might be added later
- Quite early stage of development
- Will be available in PyPI, *pip install applausequery*
- For now: pip install git+https://github.com/lupinix/applausequery.git
- Some work required on job handling, for larger queries

Trivial example: One lightcurve, as with web interface

```
from applausequery.dr3 import lightcurve

l = lightcurve.lc_by_ucac4_id("104-010297")
print(l)

# Output (cutted)
    jd_mid      bmag  bmagerr    vmag  vmagerr
                mag     mag      mag     mag
------------- ------- -------- ------- --------
2438292.59583 9.96053 0.168098 9.81953 0.208428
2438314.59444 8.64122 0.144016 8.50023 0.203487
   2438315.55 9.55363 0.179155 9.41263 0.221363
2438315.59514 8.97377 0.197298 8.83277  0.23988
...
Length = 158 rows
```

Example: Get many lightcurves in DR3

```
from applausequery.dr3 import lightcurve

# List of UCAC4 identifiers of stars we're interested in,
# example from my daily work: Crossmatch of Mira type
# stars in ASAS-SN catalog of variable stars with UCAC4
ucac4_stars = [star_1, ..., star_n]
for star in ucac4_stars:
    lc = lightcurve.lc_by_ucac4_id(star)
    lc.write("star"+".fits", format="fits")

perform_some_shiny_analysis() # for example: feets package
```

In fact this needs a bit more code in reality, but: we can get many (thousands) of light curves without requiring to use SQL directly

How to contribute?

- Right now: Mostly queries inspired by my scientific questions
- Open development
- Module easily extendable
    - New query $\rightarrow$ new function in matching submodule
    - New set of questions specific to a topic $\rightarrow$ new submodule
- Proposals: File issue or pull request on GitHub

Conclusion

- Data providers: Provide your data by using VO! It makes life for scientists much easier :)
- APPLAUSE is a very good example!
- Applausequery shows how to use PyVO as a foundation to provide support packages for your service
- APPLAUSE users: Feel free to provide ideas on further applausequery enhancements!

Questions

Thank you very much for your attention!

Questions?

Resources

General:

- APPLAUSE: https://www.plate-archive.org/applause/
- PyVO:
  - https://github.com/astropy/pyvo
  - https://pyvo.readthedocs.io/en/stable/

Pictures:

- Slide 3: Plate example
  https://doi.org/10.17876/plate/dr.3/plates/301_48552
- Slide 5:
  - Screenshot APPLAUSE https://wwww.plate-archive.org
  - Screenshot TOPCAT (selfmade)
- Slide 11: Package structure (selfmade)