cazy_webscraper

Customising scrapes and the structure of the local CAZyme database

Emma Hobbs[1,2], Tracey Gloster[1],
Sean Chapman[2], Leighton Pritchard[3]
[1]University of St Andrews, St Andrews, UK
[2]The James Hutton Institute, Dundee, UK
[3]University of Strathclyde, Glasgow, UK

Introduction

Carbohydrate **A**ctive en**Zymes** (CAZymes) are pivotal in pathogen recognition, signalling, structure and energy metabolism. CAZy (www.cazy.org) is the most comprehensive CAZyme database [1]. CAZy does not provide methods for automating data retrieval or submitting sequences for annotation.

cazy_webscraper retrieves user-specified datasets from CAZy, producing a local SQL database to enable thorough interrogation of the data. cazy_webscraper can also retrieve protein sequences from GenBank [2] and download structure files from RCSB PDB [3].

Methods

Installation via GitHub:
https://github.com/HobnobMancer/cazy_webscraper

Scraping is invoked using the command `python3 cazy_webscraper`. All optional flags can be found in the GitHub repository README.

Expanding the dataset to include protein sequences from GenBank and structure files from PDB is achieved using the expand module.

Full documentation is available at:
https://cazy-webscraper.readthedocs.io/en/latest/?badge=latest

Requirements

- POISx or Mac OS, or a Linux emulator
- Python3 version 3.8+
- Internet access while scrapping CAZy
- Required Python libraries are found in the GitHub repository `requirements.txt`

1. GenBank

Each unique CAZyme is identified by its unique **primary** GenBank accession.
Primary accessions are accessions written in bold in CAZy, which CAZy defines as the 'best' model [4]. **Non-primary** accessions are accessions not written in bold.

cazy_webscraper can retrieve GenBank protein sequences for:

- Specific CAZy classes and families
- Specific kingdoms, genera, species and strains
- CAZymes without sequence records
- CAZymes whose sequence has been updated since it was last retrieved

Protein sequences are added to the local database, and can be written to FASTA files

2. CAZy Families

To scrape **CAZy subfamilies** use the --subfamilies flag.

Specific **CAZy (sub)families** can be scrapped using --families flag. For example, to scrape GH1, GH3 and PL1_2 use the command:

```
python3 cazy_webscraper --families
GH1,GH3,PL1_2 --subfamilies
```

To specify **CAZy classes** to scrape, use the --classes flag followed by the classes to scrape.

Scraping specific families and classes significantly **reduces waiting times.**

Combine as many flags as you wish. For instance, to scrape bacteria from certain families and all CBMs use:

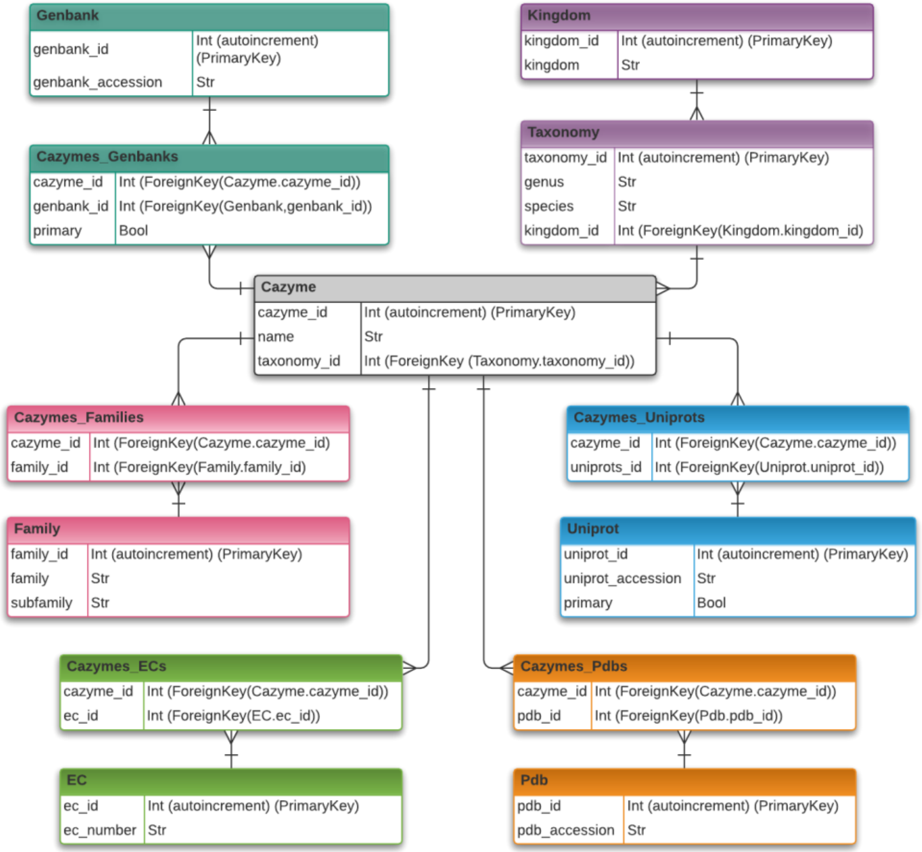
```
python3 cazy_webscraper --classes
CBM --families GH5,GH7,PL8
--kingdoms Bacteria
```

3. EC Numbers

To restrict the scrape to only CAZymes annotated with specific EC numbers, use the --ec flag at the command line, followed by the EC numbers of interest.

```
python3 cazy_webscraper –ec
3.2.1.-,3.2.1.21,2.4.1.152,2.4.1.214
```

Note, when listing any scraping criteria, separate the items with commas only and no spaces



Genbank	
genbank_id	Int (autoincrement) (PrimaryKey)
genbank_accession	Str

Cazymes_Genbanks	
cazyme_id	Int (ForeignKey(Cazyme.cazyme_id))
genbank_id	Int (ForeignKey(Genbank,genbank_id))
primary	Bool

Kingdom	
kingdom_id	Int (autoincrement) (PrimaryKey)
kingdom	Str

Taxonomy	
taxonomy_id	Int (autoincrement) (PrimaryKey)
genus	Str
species	Str
kingdom_id	Int (ForeignKey(Kingdom.kingdom_id)

Cazyme	
cazyme_id	Int (autoincrement) (PrimaryKey)
name	Str
taxonomy_id	Int (ForeignKey (Taxonomy.taxonomy_id))

Cazymes_Families	
cazyme_id	Int (ForeignKey(Cazyme.cazyme_id)
family_id	Int (ForeignKey(Family.family_id)

Family	
family_id	Int (autoincrement) (PrimaryKey)
family	Str
subfamily	Str

Cazymes_Uniprots	
cazyme_id	Int (ForeignKey(Cazyme.cazyme_id))
uniprots_id	Int (ForeignKey(Uniprot.uniprot_id))

Uniprot	
uniprot_id	Int (autoincrement) (PrimaryKey)
uniprot_accession	Str
primary	Bool

Cazymes_ECs	
cazyme_id	Int (ForeignKey(Cazyme.cazyme_id))
ec_id	Int (ForeignKey(EC.ec_id))

EC	
ec_id	Int (autoincrement) (PrimaryKey)
ec_number	Str

Cazymes_Pdbs	
cazyme_id	Int (ForeignKey(Cazyme.cazyme_id))
pdb_id	Int (ForeignKey(Pdb.pdb_id))

Pdb	
pdb_id	Int (autoincrement) (PrimaryKey)
pdb_accession	Str

Fig.1 Relationship-entity model of the local CAZyme database created by cazy_webscraper
A log table is also included. Logging when data is added and the commands used in each scrape

4. Taxonomy

To scrape by taxonomy use the following flags in any combination:

Kingdoms, use the --kingdoms flag, and list Archaea, Bacteria, Eukaryota, Viruses and/or unclassified.

Genera, --genera flag, and list of genera to retrieve CAZymes from.

Species, --species flag, and list the species to scrape CAZymes from all strains of these species.

Strains, --strains flag to specify specific strains of species.

5. UniProt

CAZy incorporates the accessions of associated UniProt records. UniProt [4] is a highly comprehensive protein database, incorporating data from a variety databases.

The **primary** UniProt accessions are accessions written in bold in CAZy, which CAZy defines as the 'best' model (see 'Help' on the CAZy website). **Non-primary** accessions are accessions not written in bold.

6. RCSB PDB

PDB is the most comprehensive protein structure database. cazy_webscraper uses the BioPython.PDB [5] module to retrieve PDB structure files for specific families, classes and taxonomies.

Use the script get_pdb_structures.py in the expand module

7. SQL Database

Queries to the local CAZyme database are written in SQLite. To learn how to query SQL database we recommend the WiseOwl SQL tutorials https://www.wiseowl.co.uk/sql/.

```
SELECT C.cazyme_id, families.family
FROM cazymes AS C
  LEFT JOIN families ON families.family_id =
  cazymes_families.family_id
  LEFT JOIN cazymes_families ON
  cazymes_families.cazyme_id = C.cazyme_id
WHERE families.family = "GH103"
```

The example query (left) retrieves all CAZyme records in the CAZy family GH103.
A copy of the database structure is stored in the GitHub repository to help define joins in queries.

Add to an Existing Database

cazy_webscraper can scrape and add data to an existing local CAZyme database.
Use the --database flag followed by the path to the local CAZyme database.

Create a HTML Page Library

Large scrapes of CAZy can takes hours or days for scraping all 2,000,000+ entries in CAZy.
For concerns of CAZy updating before the scraping is completed, or limited internet access, use the expand module to retrieve all HTML pages of interest from CAZy, and save them to a hard drive in minutes. Then scrape the CAZy data from the local HTML page library.

Conclusions

cazy_webscraper provides new, **previously unachievable** access to the proteomic data within CAZy. This facilitates inclusion of CAZy data in many studies, including functional, evolutionary, structural, genomic and metabolic studies. Thus, cazy_webscraper opens up numerous new avenues of investigation.

- **Automate** retrieving CAZy annotations, protein sequences and structure files
- **Expand** the dataset beyond that stored in CAZy
- **Thoroughly** interrogate the dataset using complex queries in SQL

References

1. Lombard, V. *et al.* (2014) 'The carbohydrate-active enzymes database (CAZy) in 2013, *Nucleic Acids Research,* 42, pp.D490–D495
2. Sayers, E. W. *et al.* (2020) 'GenBank', *Nucleic Acids Research,* 49(D1), pp.D92-96
3. Berman, Helen M. *et al.* (2000) 'The Protein Data Bank', *Nucleic Acids Research*, 28(1), pp.235-242
4. UniProt Consortium (2019) 'UniProt: a worldwide hub of protein knowledge', *Nucleic Acids Research*, 47(D1), pp.D506-D515
5. Hamelryck, T., Manderick, B. (2003) 'PDB parser and structure class implemented in Python', *Bioinformatics* 19(17), pp.2308-2310

Acknowledgements

We thank the EASTBIO Doctorial Training Partnership for funding out work.