Inf553 Project

Ioana Manolescu, Pierre Bouhris, Théo Bouganim

October 2023

Overview

The ultimate goal of the project is to set up a Web application allowing users to inspect and update bibliographical data from Pubmed[1], a well-known repository of bibliographical information about publications from the biomedical domain.

The project is divided into three phases:

- Create and load the database.
- Express a set of queries and updates in SQL.
- Write the Web application that allows to interact with the system through a Web browser.

1 Creating and loading the database

1.1 Data outline

We provide the data as a set of CSV files. It describes a set of scientific **articles**, together with their **authors**, and **affiliations** of their authors. Two other main types of information are present for some, but not all, the articles:

- **Grant** information: research is often financed by research grants, provided by state agencies or (more rarely) private foundations or companies;
- **Conflict of interest** information: sometimes, an author of an article could reap some personal benefits from decisions that may be taken based on the science presented in the article. This is the case, for instance, when the article discusses a drug that the author co-patented, or manufactured by a company in which the author has shares. Some journals require authors to state any such situations. The authors may use this field to say that no conflict of interest exists.

pubmed_article has six attributes:

- article_id which is an integer primary key;
- title which is the article title, not null;
- journal_title which is the title of the journal in which the article was published, not null;
- doi, which is the Digital Object Identifier (bibliographic code) of the article, not null;
- pubmed_link, which is the URI of the paper in the PubMed repository, not null;
- year, an integer, the publication year of the paper, not null.

pubmed_affiliation has two attributes:

- affil_id which is an integer primary key;
- norm_affil which is an affiliation string (non-null), such as 'university of saskatchewan college of medicine saskatoon canada'. The values of this attribute have been *normalized*, that is: moved to lowercase, and punctuation has been removed.

pubmed_author has three attributes:

[1] `https://pubmed.org/`

- author_id which is an integer primary key;
- author_name which is a string (non-null);
- affil_id which is a foreign key into pubmed_affiliation.

article_author connects articles to their authors. It has two attributes:

- article_id is a foreign key into pubmed_article;
- author_id is a foreign key into pubmed_author.

grant_info has two attributes:

- grant_id which is an integer primary key;
- grant_val which is the grant information supplied by the authors, e.g., 'ANR-15-IDEX-01'. This field is not null.

article_grant connects articles to their grants. It has two attributes:

- article_id which is a foreign key into pubmed_article;
- grant_id which is a foreign key into grant_info.

article_coi specifies the conflict of interest data of each article. It has three attributes:

- article_id which is a foreign key into pubmed_article;
- coi_id which is a primary key of article_coi;
- coi_text which is a Conflict of Interest textual description.

1.2 Loading the data

You will find at this address:

`https://drive.google.com/file/d/1q3LGbjCqjxg2KbM1DsCmwj6-B7p-WZi6/view?usp=share_link`

a file titled PUBMED_DATA.zip. Decompress it and you will obtain seven files, one for each relation. Each file has the name of a table described above. In each file, the first line contains the attribute names described above. The largest file corresponds to article_author: it contains 697103 records. The smallest corrersponds to grant_info, it contains 32121 records.

Create a database called **pubmed**. In this database, create the seven tables, and load the data into them, using `COPY` (or `\COPY`) as seen in the first PostgreSQL lab. You can:

- Create the tables with all the constraints; then, load the data. Loading in this way may take longer since the system needs to check that all the constraints are satisfied.
- Create the tables without constraints; then, load the data; then, modify each table (see `https://www.postgresql.org/docs/current/sql-altertable.html`, look for "Examples" in the page) to add the constraints it needs.

Use SchemaCrawler to generate an image of the database schema:

1. Download SchemaCrawler
2. Within the directory you downloaded, access the _schemacrawler directory, then:

```
./schemacrawler.sh --server=postgresql --user=[username] --database=[dbname] --command=schema
--info-level=standard --portable-names --output-format=png --output-file=schema.png
```

1.3 What to turn in

1. An .sql file with **all the SQL commands you ran to create and populate the database**. The name(s) of the student(s) who wrote them need to be in the first line of the file in a comment (SQL comments are prefixed with `--`).
2. The .png file with a **drawing of the schema of the database**, produced by SchemaCrawler.