

CMP645 -- Assignment 5
100 points
Due Monday, April 23, 11:59 pm

OLAP Analysis in PostgreSQL and Visualization using Jupyter Notebook


A. Software Requirements

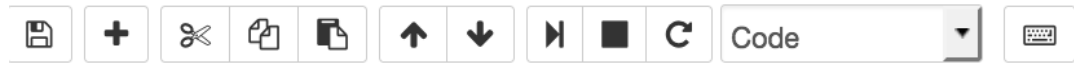
1. PostgreSQL:

- a. Like in Homework 2, we will be using PostgreSQL to store the DBLP data in a well-defined schema
- b. For installation instructions, refer to the Homework 2 handout

2. The Python Anaconda Toolkit:

- a. We will be using a few Python libraries for data analysis and visualization like Pandas, matplotlib and the Jupyter Notebook
- b. The above mentioned packages and a lot more are provided by Anaconda, which is a free and open source distribution of the Python and R programming languages for large-scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment
- c. Installation instructions(In each case, please download the version of Anaconda that uses Python 2.7):
 - i. [Windows](#)
 - ii. [Linux](#)
 - iii. [MacOS](#)
- d. **Using the Jupyter Notebook:** The Anaconda installer would have installed Jupyter Notebook, which we will use for analyzing data and to generate visualizations. Here's a few things to know so you can use it effectively:
 - i. The Jupyter Notebook is an open source web application that allows users to create and share documents that contain live code in languages like Python, R and Scala
 - ii. The Jupyter Notebook file: A Jupyter notebook can be identified by its file extension .ipynb.
 - iii. To start a notebook, cd into the directory that contains the .ipynb file you want to work with from your terminal and type `jupyter notebook`.
 - iv. You will see a File Explorer like interface pop up on a tab in your browser, from where you can choose the Notebook you want to open.
- e. **Structure of a Jupyter Notebook:**
 1. The Jupyter notebook consists of many *cells*, which can contain code snippets whose output can be seen when executed

2. It is very commonly used for conducting exploratory analyses on datasets
3. You can execute each cell by clicking the  button or pressing Shift+Enter, which is the keyboard shortcut. We do recommend looking at the entire list of shortcuts [here](#). They are very convenient and save a lot of time.
4. The main functions you would need are in the toolbar that will look similar to this:



3. **Scripts for visualization:** In this exercise, we provide a Jupyter Notebook showing a few examples of visualization. The notebook can be downloaded [here](#).

B. Dataset and Schema

In this problem set, we continue to use the DBLP publication database, containing information of over 3 million papers published in computer science conferences and journals. (This data was derived from the DBLP system, maintained by Michael Ley at <http://www.informatik.uni-trier.de/~ley/db/>).

The dataset can be downloaded from [here](#).

This database consists of four tables:

- (1) the **authors** table: containing the ids and names of authors
- (2) the **venue** table: containing information about conferences or journals where papers are published. Although attributes 'id' and 'type' have numeric values they can be interpreted as follow:
 - id: refers to a conference/journal in a specific year (e.g., "SIGMOD Conference", 2015)
 - name: "venue_name" refers to the conference without the year (e.g., "SIGMOD Conference")
 - type: type 0 corresponds to 'Journal Articles', type 1 corresponds to 'Conference and Workshop Papers', and type 3 corresponds to 'Books and Thesis'.
- (3) the **papers** table: describing the papers
- (4) the **paperauths** table: indicating which authors wrote which papers.

The schema is the following, where the underline indicates the primary key of a relation.

authors (id: INTEGER, name: VARCHAR(200))

venue (id: INTEGER, name: VARCHAR(200) NOT NULL, year: INTEGER NOT NULL, school: VARCHAR(200), volume: VARCHAR(50), number: VARCHAR(50), type: INTEGER NOT NULL)

papers (id: INTEGER, name: VARCHAR NOT NULL, venue: INTEGER REFERENCES VENUE(id), pages: VARCHAR(50), url: VARCHAR);

paperauths (paperid: INTEGER, authid: INTEGER)

Create these tables in a Postgres database, and use the corresponding CSV files from the dataset to populate them.

C. Tutorial on Analytics and OLAP in PostgreSQL

The goal of this exercise is to run complex analytics over the DBLP dataset. You may find the following PostgreSQL commands useful in this exercise. We have compiled some information about them, and relevant links in a tutorial.

- a) *Temporary tables, views, and materialized views*: slides 1-8 of the OLAP tutorial
- b) *Full-text search*: slides 9-12 of the tutorial
- c) *PostgreSQL functions – more text search, strings, arrays*: slides 13-14
- d) *Table function – Crosstab*: slides 15-17
- e) *Table expressions – Grouping Sets, Rollup, Cube*: slides 18-20

The tutorial can be accessed [here](#).

D. Analytical tasks

Please complete the following analytical tasks in this exercise.

What to turn in: You need to submit the following items to Gradescope to complete this assignment:

- 1) The individual query files (Gradescope Assignment 1)
- 2) The iPython Notebook as a PDF (Gradescope Assignment 2)

~~1. [15 points] Find all the publications whose title contains the keywords “big” and “data.”~~

~~(a) Store the output as a relation as follows:~~

~~`big_data_pubs(id: INTEGER, name: VARCHAR NOT NULL, venue: INTEGER REFERENCES
VENUE(id))`~~

~~Example Format (big_data_pubs):~~

| id | name | venue |
|-----------------|--|------------------|
| 1234 | Tensor decomposition of Toeplitz Jacket matrices for big data processing. | 12345 |

~~(b) Then count the number of papers in this table.~~

~~**SQL Submission:** Please submit the SQL statements to (a) create the relation and (b) count the number of papers in a single file Q1.sql to Gradescope.~~

- 2. [15 points]** Count the “big data” publications, as identified above,
- by venue, which is identified by attribute id (the id of the venue),
 - by venue_name (identified by the name of the venue), and
 - by type (identified by the type of the venue).

You will have to use grouping sets for this.

The output table should look like:

| type | name | id | count |
|------|----------------|-------|-------------|
| 0 | | | 10001 |
| 1 | | | 2325322 |
| | CloudCom-Asia | | 34253423463 |
| | ACM Multimedia | | 34253423 |
| | | 12345 | 3425323 |
| | | 1234 | 34253 |

SQL Submission: Please submit the SQL query statements used for this part as Q2.sql to Gradescope.

- 3. [20 points]** Count the number of publications in each venue type. Note that in the venue table, type 0 corresponds to ‘Journal Articles’, type 1 corresponds to ‘Conference and Workshop Papers’, and type 3 corresponds to ‘Books and Thesis’.

Example Output:

| type | publications |
|--------------------------------|--------------|
| Journal Articles | 1234 |
| Books and Thesis | 123 |
| Conference and Workshop Papers | 12345 |

SQL Submission: Please submit the SQL query statements in Q3.sql to Gradescope

Visualization:

- For visualizing the results, save the output of the query to the file papers_count_per_type.csv. To do this, you can use the [COPY TO](#) command in SQL.
- While submitting the query to Gradescope, remember not to include the COPY TO portion of the query in the file.**
- In the given Jupyter Notebook, run the cells under the section Query3 to complete the visualization part of the exercise.

4. [25 points] Count the number of publications by type per year. Note that in this dataset, if the year of publication is set to 0, it means that the year of publication is missing. Please ignore such publications in this task.

Example Output:

| year | journal_articles_publications | conference_and_workshop_papers_publications | books_and_thesis_publications |
|------|-------------------------------|---|-------------------------------|
| 1234 | 1234 | 1 | 45 |
| 1235 | 1111 | 123 | 111 |

SQL Submission: Please submit the SQL query statements in Q4.sql to Gradescope Assignment 1.

Visualization:

- For visualizing the results, save the output of the query to the file papers_count_per_type_per_year.csv. To do this, you can use the [COPY TO](#) command in SQL.
- While submitting the query to Gradescope, remember not to include the COPY TO portion of the query in the file.**
- In the given Jupyter Notebook, run the cells under the section Query4 to complete the visualization part of the exercise.

5. [25 points] First compute the number of coauthors, denoted as #coauthors, for each publication between 1995 and 2005. Then count the number of publications for each distinct value of #coauthors between 1995 and 2015. Visualize the result for #coauthors = 1, 2, 3, 4, 5 using 5 distinct bar plots.

Example:

| coauthors | year_1995 | year_1996 | | year_2015 |
|-----------|-----------|-----------|-------|-----------|
| 1 | 12345 | 1234 | | 12345 |
| 2 | 123 | 12 | | 1234 |

SQL Submission: Please submit the SQL queries in Q5.sql to Gradescope

Visualization:

- For visualizing the results, save the output of the query to the file papers_count_per_coauthors_between_1995_and_2015.csv. To do this, you can use the [COPY TO](#) command in SQL.
- While submitting the query to Gradescope, remember not to include the COPY TO portion of the query in the file.**
- In the given Jupyter Notebook, run the cells under the section Query5 to complete the visualization part of the exercise.