DSAIT4000 Indexing

October 4, 2024

Asterios Katsifodimos

This is the second part of the second assignment for the DSAIT4000 Data Management and Engineering.

Installation instructions can be found in the accompanying pdf.

The groups will be the same as the first assignment.

The deadline will be **22/10/2024 23:59:59 sharp**.

You must submit :

- **A PDF containing all of your discussions and results.**

---

1. **An IMDB experience**

In this task you have the chance to play around with the real data in the IMDB (Internet Movie Database) dataset you have been provided with. The task consists of the following steps:

(a) Import an altered version of the database. For this task, we have uploaded another version of the IMDB dataset, which you can download here. Clear instructions can be found here. Create a new database and import it there.

(b) Familiarize yourself with the schema. Check the tables of the schema and their columns. Furthermore, take a look at the properties of these tables and any foreign key constraints! You will notice that PK and FK information are missing.

(c) Learn to use the EXPLAIN mode. First enable costs and timing in explain options, if you are using pgAdmin 4, and run some simple queries in EXPLAIN ANALYZE mode (see Figure 1). (For those running the database from the terminal, simply run the query with EXPLAIN ANALYZE in the front). Afterwards, use EXPLAIN ANALYZE to find the total cost and execution time of the following SQL statement for finding all the staff names of Star Wars Episode IV- A New Hope:

```
SELECT DISTINCT p.full_name
FROM (titles t JOIN cast_info ci ON t.title_id = ci.title_id)
JOIN persons p ON ci.person_id = p.person_id
WHERE t.primary_title = 'Star Wars: Episode IV- A New Hope'
```

(d) Improve your database. The total cost of the aforementioned Star Wars query should be around 0.120 ms on our laptop. Write down your results of the time.

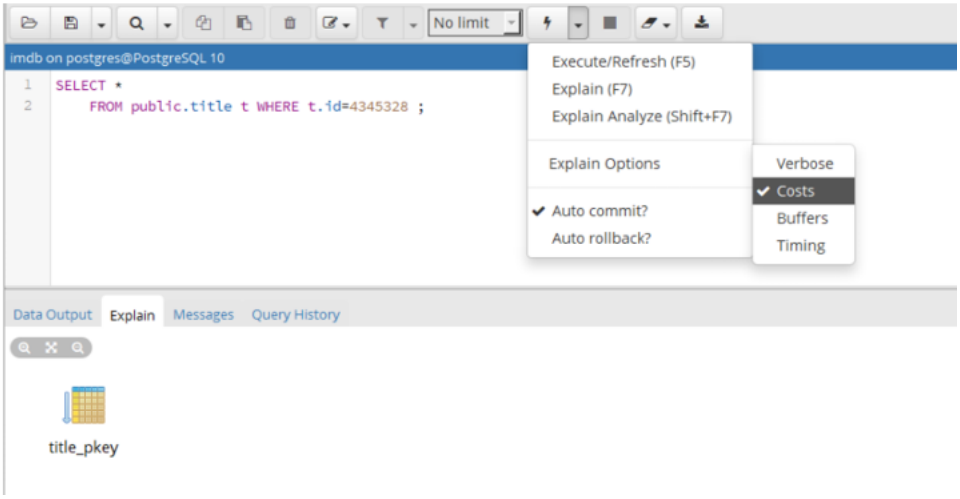

Figure 1: Figure 1 : Using EXPLAIN ANALYZE mode in pgAdmin 4.

If your result is significantly worse, figure out the reason for that. Are there indexes missing? Add them. Fix the problem and make your database go faster! Show us the results of making the query faster.

(e) Database cleaning. Inspect the episode's table. In theory, episode and series IDs derive from title IDs of the titles table. Is this the case here? If not, can you fix it and create proper FK references?

(f) Export the two tables that were used in the query of task c) into CSV format.

(g) Use parallel processing tools such as Python Dusk to execute the same query on the data with dataframes. Show us the processing speeds of performing the parallel vs. single CPU processing. No need to create a cluster, parallel execution on multiple CPU cores on your laptop should suffice.

(h) Shortly discuss your results: the speedup that improved the query's speed. Please also provide screenshots that show what you have done.