Kenny Callaghan

Ali Erkan

Database Systems

19 December 2019

YouTube Database Writeup

I am not sure exactly what kind of information I should be presenting in this writeup. Initially, I considered simply copy-pasting the paragraphs I wrote on my poster into a Word document, but I feel as if that would be not only disingenuous, but also insulting to your intelligence. Instead, I will try to add supplemental information about my project that I couldn’t find a place for on the poster.

When this project was initially assigned, I felt lost and overwhelmed. At the same time, however, I felt like I was being a great and unique opportunity to leave my comfort-zone and produce something not just because it was for a class, but because I wanted to. I was granted the opportunity to produce, as you said, an artifact that serves as a testament to my ability and knowledge that I have gained throughout this course.

After getting on my way with this project, I really did not encounter a lot of difficulty. If I had more time, I would have loved to incorporate actual data from YouTube itself, working with their API or using a web-crawler in some way, but I am content with the final project either way. One challenge I did face involved the population of the tables, but that was resolved by moving from AWS to a local installation of PostgreSQL. I do have one anecdote that I would have liked to mention during my presentation, especially if it was to first-year CS majors. When I was populating the URL table, I created a list of every letter, upper and lowercase, number, and two additional symbols (\_ and -). When running my program however, it was only populating at about 50 rows a minute. In order to produce my goal of 100,000 rows, I would have spent close to a day and a half waiting for the program to finish executing. Clearly, this was unacceptable. Upon inspection of my code, however, I realized that I was doing something very silly: I was recreating the list of alphanumeric characters on each iteration of the loop to insert a row. I did a fast calculation, to realize that this was impacting performance by adding an additional 64 steps for each insert, with a total of about **1.6** **billion** futile steps in all. After optimizing my code, and moving to a local PostgreSQL installation, the insertion jumped from about 50 rows per minute, to over 500 rows per minute, sometimes getting very close to 1,000. Always double check your work!

Overall, I found this project and this course to be very enjoyable. I had some database experience prior to this class, and it was just dry enough to convince myself that anything is better than working with data. After taking this course, however, I’ve realized that under the right circumstances, data, which to me felt so bland and dry, can really come alive. I enjoyed all of my time in class and working on this project, and I hope I can incorporate large-scale data in my future coursework and professional career.