DS4300 HW1 - Testing the Limits of the Relational Model

Team Members: *Danish Farooq*

Hardware Specs:

2021 Apple M1 Pro Laptop with 32 GB RAM and 512 SSD. This laptop has 10 CPU cores: 8 for High Performance, 2 for efficiency.

Tech Stack:

- Java 11 for all code
    - Libraries: Postgres's drivers + Lombok (for some basic data model annotations)
- Postgres 14.1 as the Relational Database (as this is the only version compatible with the ARM64 architecture to my knowledge)

Analysis:

API Method	Result
postTweet	9994.6 tweets per second
getHomeTimeline	1298.5 timelines per second

These numbers seem very high. I would consider that the new M1 Pro chip with the ARM64 architecture might be a factor, but these numbers seem very high compared to other students with the same chip.

I ensured that all of the tweets existed in the database after posting them, with accurate data as well. I ensured that all tweets were present and all 1000 twitter users and the 30,000 follows in the follows table were there as well. I ensured that we were able to fetch the timeline for various users as well.

So I am a bit baffled by the results. The numbers show that my code is on par with Twitter's servers. Perhaps this has to do with the higher version of Postgres: higher efficiency in drivers. I used Postgres's default index for user_id's in the tweet table and and a multi-column index on the follows table for the user_id and the follows_id. In the release notes of Postgres 14.0, we see that index updates for insertions are handled much more efficiently. But perhaps the most probable reason of this all, indexing + many other operations here are much simpler here than they are in an actual production database.