Brian Reicher

Dr. Rachlin

DS4300

27 January 2023

MySQL Twitter Backend Analysis

For my implementation of Twitter's backend in a relational database format, I chose to use Java and MySQL 8.0.32 (with InnoDB), utilizing JDBC to interact with the database from my API. All API calls were run on a 16" Macbook Pro with MacOS 13.1 (22C65). It has a 2.3 GHz 8-Core Intel Core i9 processor and 16 GB of RAM with a 16 GB 2667 MHz DDR4 chip. Due to time constraints with other assignments, I was unable to implement batched insertions for the postTweet API method. All API call metrics are shown below.

API Method	**API Calls/Sec**
postTweet	1923.8
getHomeTimelines	39.8

I believe that there are several clear reasons as to why my API functions the way it does. Given that the postTweet method shows moderate-to- poor performance (in comparison to Twitter's rates), I feel that not batching tweet-inserts harmed the method's overall production. Additionally, having a Mac M1 or M2 chip would have significantly improved performance. Despite the mediocre performance of tweet posting, the API shows relatively decent performance for fetching home timelines in comparison to the rest of the group, yet still poor compared to Twitter's rates. I believe that this is because my only limitation here is the quality of my hardware and inescapability of using JOINS, since my queries are written as efficiently as I could.