Rayan Bouaazzi

Professor Cankaya

SE 3354

18 November 2022

Section 5: Project Deliverable 2

Comparisons with other Application implementation

For this project, we compare our database querying to the company Instacart. Instacart has a similar vision to our project, which is why we attempted to make a similar structure to how we plan for accessing and CRUDing database information in the future. The model for Instacart is that they spread out the flow of tables and schemas to avoid caching issues and possible factor malfunctions when querying or accessing database information, which is why they have an availability cacher within our databases to ensure we are not exceeding the memory limits of our schemas. Something that Instacart has that could be a possibility and something to consider upon implementation is having multiple databases covering different aspects of their companies ideals. Although our model only has one database at the moment, this aspect is something to consider when expanding our capabilities and expanding the fields we intend to reach based on the growth of our business model. When Instacart implements new tables to their databases, they copy them and distribute them accordingly to avoid merging failures and creating simplicity for CRUDing the rows within them, this is something that may be implemented in our model in the future, but as of right now we feel as if having

different contents will be easier to identify items especially since our goal is to create the quickest path to an item. And finally, something that I believe both our product and Instacart have in common is verifying the integrity and validity of the contents within our databases. This means that our availability cacher (something that is exclusive to our project) will ensure that our queries are preserving memory and cpu usage and also making sure that the items and row content within our databases contain the most up to date and functional information within them.

Source:

https://tech.instacart.com/scaling-at-instacart-distributing-data-across-multiple-postgres-databases-with-rails-13b1e4eba202