CS 306 PROJECT PHASE 3

In this phase of our project, we are going to apply indexing on our tables and analyze their effectiveness. First, we will pick two tables to join. After choosing tables, we will populate them in python visual studio code, where those tables need to have at least 1 million records.

First we will choose Warehouse and Shipment tables;

We created database to work on and created tables;

A screenshot of a computer

Description automatically generated

Then we connected our workbench on visual studio code with connection.py python;

A computer screen shot of a code

Description automatically generated



Then we populate the tables in populator.py, where those tables need to have at least 1 million records. It uses the Faker library to create dummy data and random module functions for generating numbers within specified ranges. The Warehouse table is filled with a unique ID, location, capacity, and a responsible person's name for each record. The Shipment table gets a unique ID, supplier ID, warehouse ID, along with the shipment and arrival dates. It avoids inserting duplicates and commits all changes to the database.

A screenshot of a computer program

Description automatically generated

The code below is for interacting with a MySQL database. It begins by joining two tables, Shipment and Warehouse, on a common key. Then it enables profiling to monitor the performance of database queries. It includes two queries to select shipments within a certain date range and warehouses with a specific responsible person, both before and after creating indexes. Indexes are created on Warehouse (ResponsiblePerson) and Shipment (ArrivalDate) to enhance query performance. Finally, it ends with a cleanup section to drop the indexes if they are no longer necessary.A computer screen with text and numbers

Description automatically generated

This is the result of joining two tables with 1000000 rows:

A black and white table with numbers and text

Description automatically generated

We created the shipment table where ArrivalDate is between 2023-02-01 and 2023-03-01. This shipment table is before indexing;

A screenshot of a black and white data

Description automatically generated

And this table is created after indexing;

A screenshot of a computer

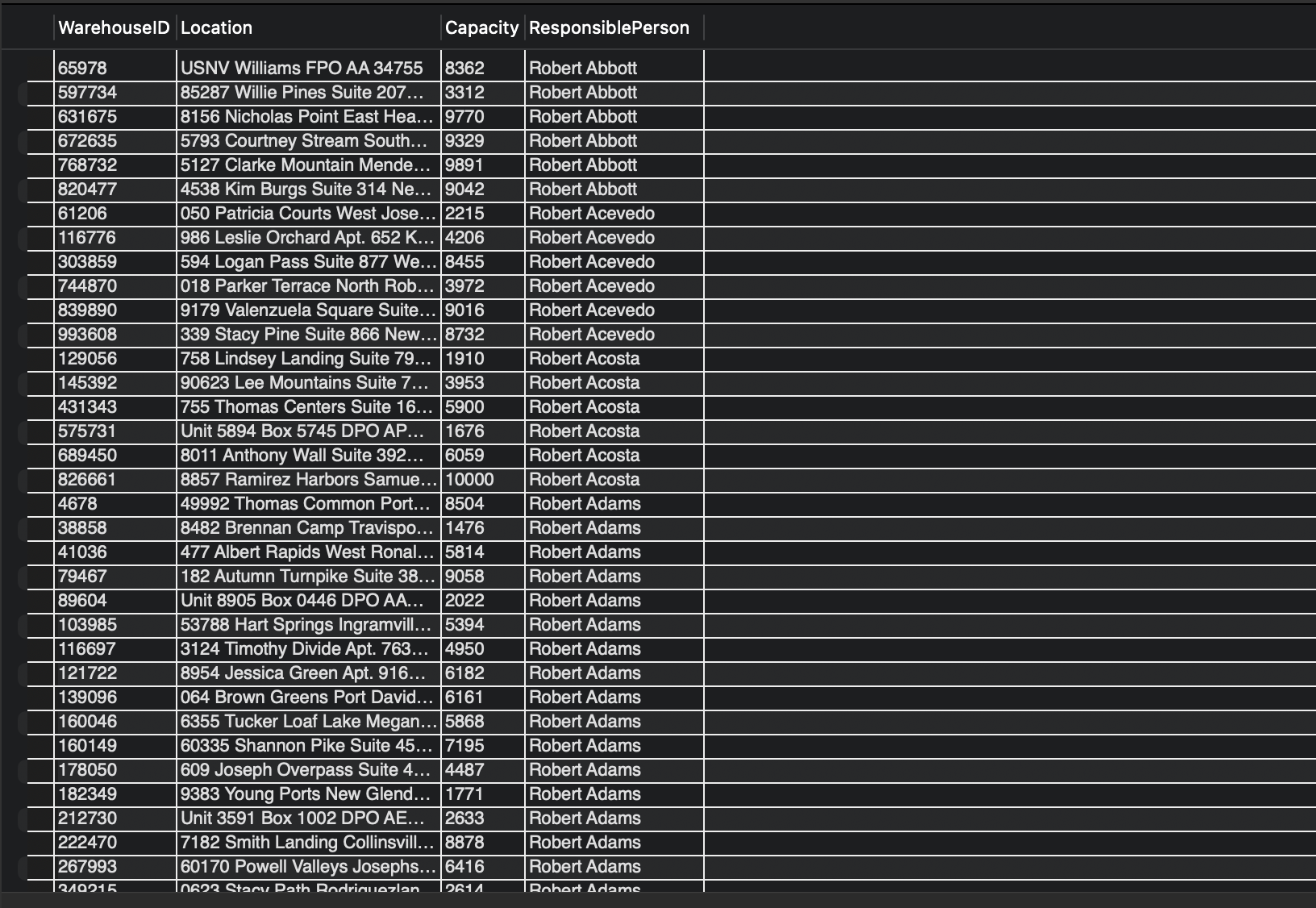
Description automatically generated

We created the Warehouse table where the ResponsiblePerson name’s include Robert, and this is the warehouse table before indexing;

A black and white table with white text

Description automatically generated

The below table is created after indexing;



Indexing in databases is used to speed up the retrieval of data by reducing the number of disk accesses required when a query is processed. An index creates a data structure (typically a B-tree) that allows for quick lookup of rows matching a given condition. Thus, we can create a database easily by using the indexing method and it leads us to save time. And this is the proof that we created the indexing tables faster.

A screenshot of a computer

Description automatically generated