Final Project Report CS327E

Robert Saenz rs42744

Stefanos Kapetanakis sk35337

**Introduction**

**Main Goal**

The purpose of the course, CS327E Elements of Databases, was to create a data warehouse and pipeline through the conglomeration of several datasets/databases together into a larger whole, while also gleaning useful data from this warehouse using SQL queries, and data visualizations. In particular, we conducted a data-driven analysis of Airbnb data in Austin and addressed the revenue being generated, the occupancy rate, the amount of rentals available, and how these metrics have changed over time.

**Datasets/Databases**

The datasets that we were instructed to use were based on housing, with long term and short term data being available. First off was the Airbnb rental data for three different cities, those being Austin. Portland, and Boston. Each of these three cities had 10 tables associated with them, containing information like listing prices, neighborhoods of listings, and host information for each host that uploaded a listing. Eventually joined with this data was the *Zillow* database information, containing the monthly prices for houses listed on *Zillow*’s website, where each kind of apartment/house was classified as 1 to 5+ bedroom, and organized into the appropriately named file based on this information.

Overall there were 36 files that were being interpreted and accessed in the BigQuery data warehouse..

**Software and APIs**

The tools used during this course were largely based on Google’s GCS (Google Cloud Services) tools. These include Google Cloud Storage for holding .csv files for upload and storing data about Dataflow jobs (which were used for data pipelining in place of MapReduce). Another major tool was BigQuery, the actual big data warehouse tool that was used to store and access the previously mentioned data files. These files were accessed using SQL, as a way to easily query them, and were finally visualized for easier consumption using Google’s Data Studio visualization tool. Apache’s Beam API was used alongside BigQuery and Dataflow to easily pipe data into BigQuery. Lastly Git was used for easy control of project files and bug/problem tracking for each of our labs, and UT’s Stache was used for easy password access among partners and teaching assistants/professors.

**Analysis of Data and Processes**

**Initial Setup**

The initial setup involved creating a GitHub for version control between multiple people. We did not extensively use GitHub to work remotely, but rather for easier collaboration in person so that we could simultaneously improve the project at the same time. Most of our final milestone files were placed in relevant folders within the GitHub, where we turned in and submitted them for each milestone in a weekly fashion. From here we also worked cooperatively through Google’s GCS implementation where we could both view and query the necessary files for the project using the IAM (Identity Access and Management) feature.

**Database Design and Schema Information**

We were handed 5 files in the Zillow dataset to use, and to incorporate among our 3 Airbnb city datasets. Our Airbnb datasets were organized by city for easier classification and interpretation of data so that we could easily compare among the three geographic locations. Each of these was populated from earlier labs in a normalized form so that data manipulation and access was as easy as possible, and so SQL query generation was as simplified as we could possibly make it for easier interpretation of the statistics for each city, especially for data regarding Austin long term and short term rental data.

In order to link the Airbnb and Zillow datasets for more information about Austin, we created a new SQL view that was intended to link together the 2 tables for optimal visualization and comparison. This view was called the “Revenue Crossover Point” and was extremely useful for gleaning information about the tables. This data includes connections on the number of Airbnb rentals available, the information about the change of this metric over time, and the impact of Airbnb earnings on long term rentals in Austin (represented by proxy through the Zillow data).

In this view, we joined the Airbnb and Zillow datasets on the bedroom, zipcode, and date fields. Then we computed the revenue crossover point metric, which was equal to the ceiling of Zillow’s median rental price per month over Airbnb’s median rental price per day. This metric we computed represents the amount of days a month an Airbnb host would have to rent out their property in order to receive the same revenue from a median long-term rental.

**Data Analysis (SQL Queries)**

So once we joined all of the data together we wanted to create some metrics in order to figure out and visualize how many short-term rentals are available, what their occupancy rate is, how much revenue is being generated over time, and how these metrics changed as time went on. To create these metrics we made 3 different SQL queries. In order to figure out the revenue generated for the rentals, we multiplied together the listing price, minimum nights, and number of reviews for each rental. We then ordered the results by the date for each listing, which would display the changes in revenue over time. For our second metric, we wanted to find the number of short-term rentals that were available in Austin and show the change over time. In order to do this, we use the COUNT() function in the listing id, which would count the amount of different listings there were. We then grouped and ordered these results by the date, so the results would show the number of rentals for each date as time went on. For our last metric, we wanted to find the occupancy rate of the rentals. In order to estimate the number of days that were occupied for each rental, we multiplied together the minimum nights and number of reviews for each listing. This would give us a good estimate of the number of days each rental was occupied. From there we ordered the results by the date, and then grouped it by the listing id so we wouldn’t get any overlap in our results.