AIRBNB ANALYSIS WITH GRAPH DATABASE – NEO4J

Report

Abstract

Airbnb is a trending online marketplace for arranging lodging, homestays or tourism experiences. Neo4j is a new trending graph database engine. In this project, we use Neo4j to create a graph database and analyze the data collect from Airbnb public data sources.

Liu, Lu

CS779 Advanced Database Management System

MSCIS, Boston University

Table of Contents

Introduction

Graph Database

The well-known Swiss mathematician Leonhard Euler was trying to solve the problem *7 bridges of Königsberg.* The following figure is how Euler solve the famous "*7 bridges of Königsberg*" problem, which was the first graph in the world. The graph database originates from the Euler theory and diagram theory. The basic meaning of the graph database is to store and query in a data structure such as "graph". Its data model is mainly represented by nodes and relationships (edges), and it can also handle Key-Value pairs. It has the advantages of quickly solving complex relationship problems. (Depeau, 2019)

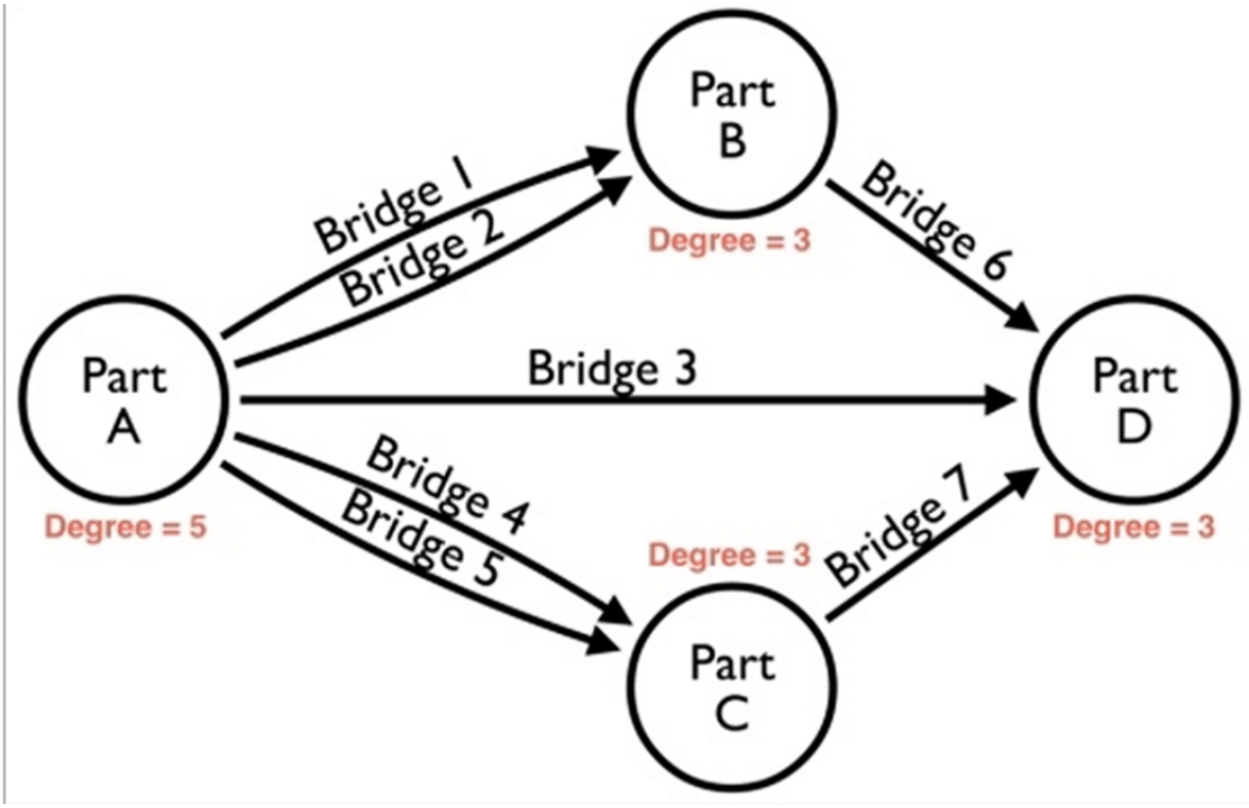


Figure 1. Seven Bridges of Königsberg

There are two basic data types in a graph database: Nodes and Relationships. Nodes and Relationships contain attributes in the form of keys or values. Nodes are connected by the relationships defined by Relationships to form a relational network structure. The nodes of the graph, in the graph database Neo4j, we refer to vertices as nodes. The relationships between the nodes of the graph database looks like the bridges connected two parts from the Figure 1. The structure of the graph is how nodes and relationships connected to each other, such as how many edges have connected to a node which also refers as degree. (Bruggen, 2014)

RDBMS VS Graph Database

As relational database, it has the advantages:

- RDBMS uses less storage space because it does not have to store all relationships into the database, which the graph database may need to have these relationships stored in the graph database.
- The data structure of RDBMS is well-understood and stable because it does not need to change frequently
- The operation of RDBMS is faster than the graph database on huge number of records

As the graph database, it has the advantages:

- The operation of graph database is much faster for connecting data than RDBMS because the graph database does not need to use any JOIN clause, as we know, while we use the JOIN clause in RDBMS will slow down the performance of RDBMS if the scale of database has grown.
- The unique query language – Cypher is a user-friendly query language, which is much simpler and has less syntax problems while user uses it
- The graph database is well-suited to irregular, complex structures. In the real-world situation, the data always irregular and complex, using the graph database is much suitable for these real-time use cases.

The graph database could be used everywhere in our real-life. For example, the Facebook uses graph database to find out more relationships between the person and their friends, and friends of friends, and they can recommend you from the database who you may get interested in and what events you may look for. Another example - Amazon uses Neptune that is a fast, reliable, fully managed graph database service that makes it easy to build and run applications that work with highly connected datasets. Neptune powers graph use cases such as recommendation engines, fraud detection, knowledge graphs, drug discovery, and network security.

Neo4j

Compare with the RDBMS – which uses the tables that includes the attributes, domains and tuples for storing the records, the graph database uses each one record as single node, which has better performance and direct visualization of the data. Within RDBMS, we use JOIN clause to link two or more tables as needed, but in the graph database, we can create the relationships between the nodes and the graph database could give us direct visualization of the deep relationships between each node if mining the data deeply. For example, from the RDBMS, we can only find the relationship from table A and table B using JOIN clause if these two tables have the same unique number, such as id number, but if the table has no common column which means they are not linked and we could not use the two tables together but only linked to their

common table first. For example, table A and table C has no common column and not linked, but table B has the same column with both table A and table C, then we could only use A JOIN B JOIN C to link these three tables if we need to use table A and C. With the graph database, we could directly link the nodes as relationship, and we could find out the relationship from each node easier. For example, if node A is linked to node B, and node B is linked to node C, then node A is linked to node C.

Neo4j is a high-performance, NoSQL graph database that stores structured data on the network rather than in a table. Neo4j can also be seen as a high-performance graph engine with all the features of a mature database. Programmers work in an objected-oriented, flexible network structure rather than in rigorous static tables, but they could have all the benefits of a full transactional, enterprise-class database. (Fernandes, Bernardino, 2018)

Benefits of Neo4j

- Comes with a set of easy-to-learn query language (Cypher)
- Do not use schema – can use any form as your needs
- Compared to relational database, queries for highly correlated data are much faster
- The entity and relationship structure have strong visualization for natural human intuitive feeling
- Support for ACID-compliant transaction operations
- High availability model to support large-volume data queries, support for backup, data locality, and redundancy
- Visual query console

Neo4j Data Structure

- Everything is stored as an edge, node, or attribute
- Node and edge may have any number of attributes
- Node and edge may be labelled
- Labels are used to narrow searches

Neo4j Use Cases

Real-Time Recommendations

Neo4j uses the real-time recommendation engines that helps the online business to make relevant recommendations in real time, which includes the information of correlated product, existing customers, inventory, and even social sentiment data. Neo4j is able to immediately capture the news shown from customer's current visit that matching historical and session data, which the batch processing cannot accomplish. (Rathle, Marzi, 2017) Since Neo4j is implemented based on graph theory, it naturally has advantages in processing maps. Thus, it suitable for logistics

management and transverse big data. Walmart uses the Neo4j as the tool of graph databases because Walmart has so many data of items and customers stored in the database, the Neo4j is a right choice to meet the demands of Walmart with its scalability and availability.

Master Data Management

Since more and more businesses become customer centric who may need to make timely business decisions through the data, which requires to unify the master data that includes products, supplier, customer, and logistic information for their operational uses. Neo4j is able to create a 360º view of the master data made in real-time for all of the operations. Cisco uses Neo4j for governance and single source of truth and a one-stop shop for all hierarchies. The master data for applications throughout Cisco, such as customer and product system of record. (Rathe, Marzi, 2017)

Fraud Detection

To detect fraud, normally investigators need to look for abnormal connections between people, events, places, and times. Sometimes, they need to scan large, noisy, complex and often incomplete datasets to understand which connections (between data) are real and which may indicate fraud. By using Neo4j, the real-time graph database could help to connect the sophisticated scenarios with the existing fraud cases with its high-level of accuracy, so that the business could be able to detect the fraud scams in real-time. (Rathe, Marzi, 2017)

Graph Based Search

Because Neo4j has the knowledge graphs that could be used to manage an organization's growing digital assets, some businesses, such as Lufthansa Airline, use Neo4j to get the real-time results to manage their digital assets through the power of graph-based search. (Ratha, Marzi, 2017)

Network & IT-Operations

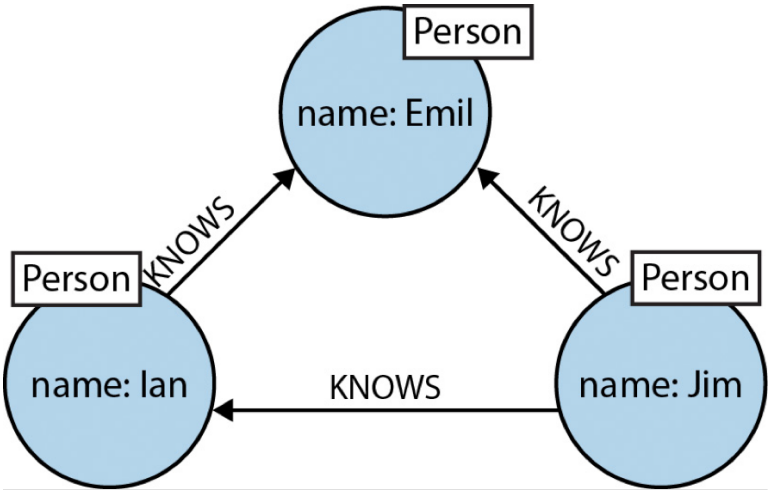
Neo4j allows to correlate the network, data center and IT assets to help with troubleshooting, impact analysis and capacity or outage planning. (Ratha, Marzi, 2017) Because the graph database is able to connect many tools and insights to the complex and deep relationships between the network operations.

Identity & Access Management (IAM)

IAM usually stores the information about entities, resources, and access rules, which the datasets are enormous and high-connected, Neo4j's native-graph query engine traverses millions of relationships per second to maintain application performance and user productivity.

Cypher

Cypher is designed to be easily read and understood by developers because it matches exactly the way we describe graphs using diagrams. For example, (emil) <- [:KNOWS] – (jim) – [:KNOWS] -> (ian) – [:KNOWS] -> (emil) that naturally follows the way in the picture as below that describes a path forms a triangle connects each node with its relationship and type. The Cypher notion allows us to find data follows the pattern. To write a SQL statement we may use many JOIN clauses that will quickly lose sight of what the query actually does because of the technical noise and also slow down the performance when the scale has grown. By using Cypher, the syntax stays clean and allows us to focus on the concepts since the queries are visually. (Hunger et al., 2016)



Datasets Used

Data Source

The dataset was achieved from Kaggle.com and has been pre-processed and cleaned for using to create the graph database with Neo4j.

https://www.kaggle.com/tylerx/melbourne-airbnb-open-data#cleansed_listings_dec18.csv

Data Description

This dataset is collected from Melbourne Airbnb Open Data, as part of the InsiderAirbnb initiative, this dataset describes the listing activity of homestays in Melbourne, VIC, Australia. The dataset used for analyzing Airbnb is included three csv files that are listing.csv, host.csv, and review.csv.

- Host report(host.csv): This report data includes the information about the hosts who are registered in Airbnb.
- Review report(review.csv): This report data shows the reviews written by the guests after staying in a certain accommodation.
- Listing report(listing.csv): This listing report stores information about the existing accommodation around Melbourne listed in Airbnb.

Detailed Data Attributes

In host.csv, the data includes the following attributes:

Host_verifications: the verification information of the host

Host_url: the url of the host

Host_location: the address of the host

Host_is_superhost: the indicator of whether the host is a super host or not

Host_response_time: the response time of the host

Host_since: the beginning date of the host

Host_id: the unique id number of the host

Host_name: the name of the host

In the listing.csv, the data includes the following attributes:

Summary: the summary of the host

Amenities: the amenities that the host have

picture_url: the url of the host pictures

Listing_id: the unique id number of the listing

Listing_url: the url of the listing

Host_id: the unique id number of the host

Extra_people: the price of adding an extra people

Zipcode: the zipcode of the host

Calculated_host_listings_count: the calculated count of the host listing

Price: the price of the host

Street: the street address of the host

Neighborhood: the neighborhood area of the host

Name: The room name of the host

Room_type: the type of the room

Longtitude: the longtitude of the room

In the Review.csv, the data includes the following attributes:

Date: the data of the reviewing

Review_id: the unique id of the review

Review_scores_rating: the score rating of the review

Comments: the comments of the host

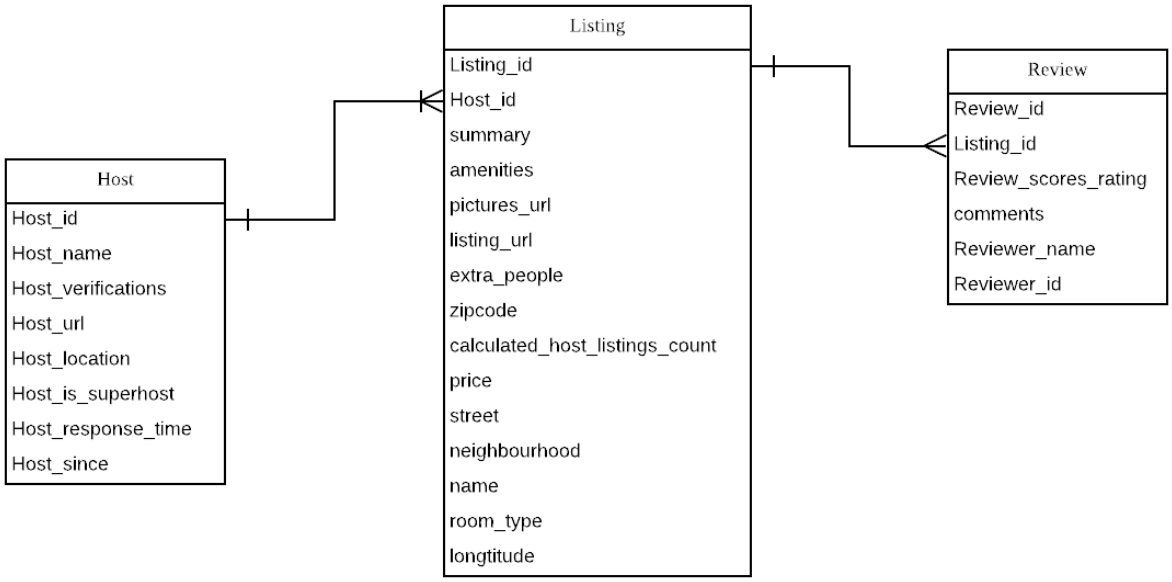
Listing_id: the id number of the listing linked to the review

Reviewer_name: the name of the review

Reviewer_id: the unique id number of the reviewer

Schema

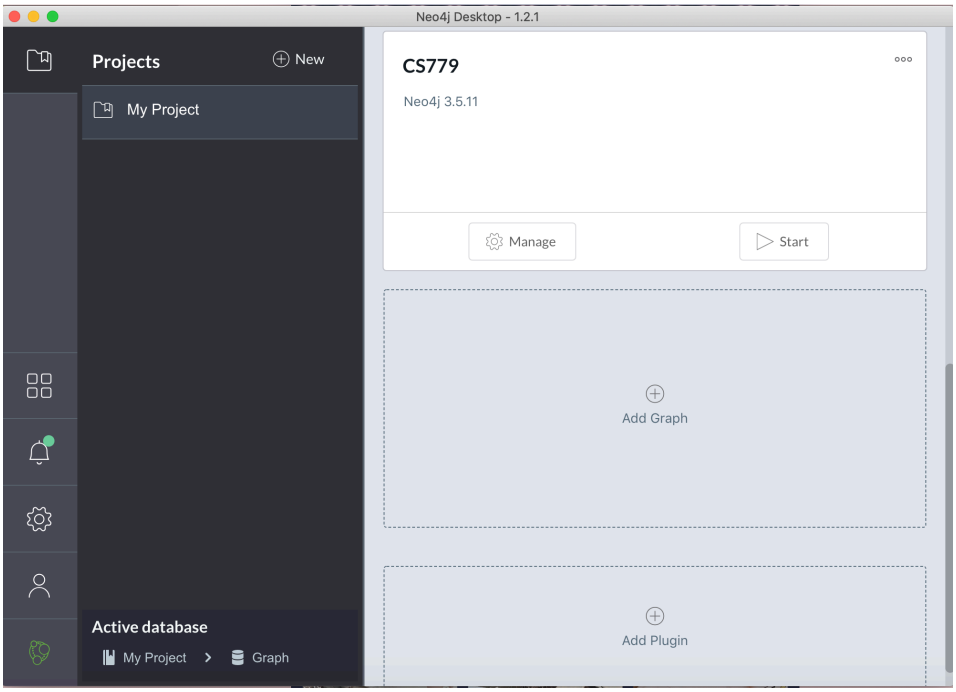
The schema includes three data reports and relationships as followed:



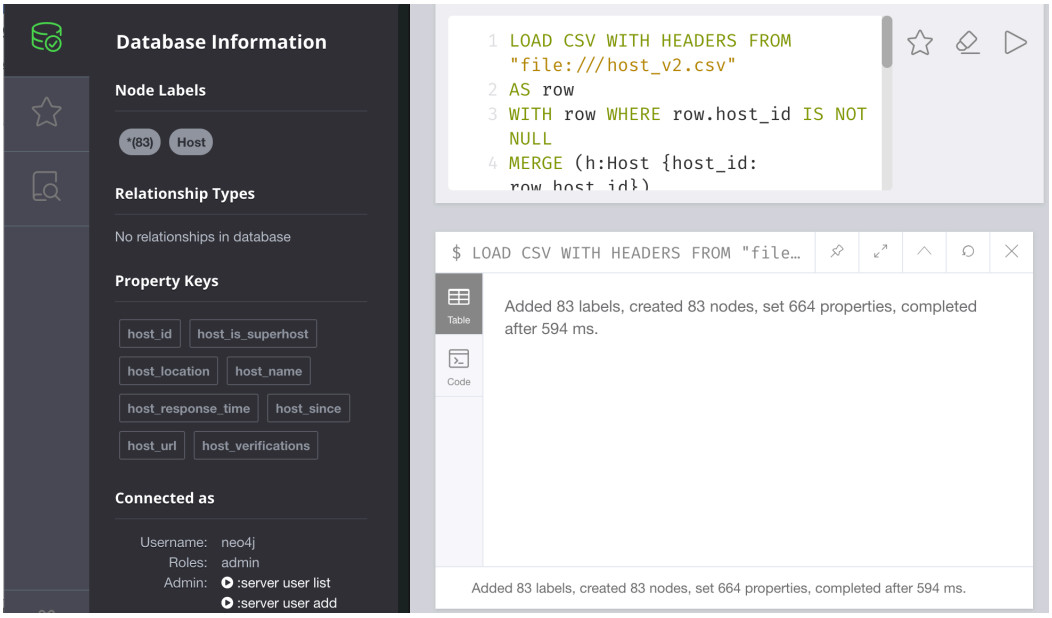
Host table has host ID as primary key and refers as foreign key in Listing table. Listing table has listing ID as primary key and refers as foreign key in Review table. Review table has Review ID as primary key, it also includes Reviewer ID which is the primary key of the Reviewer table but we do not have the reviewer data here, so we only use review ID as primary key and listing ID as foreign key link to listing table.

Graph Database Create

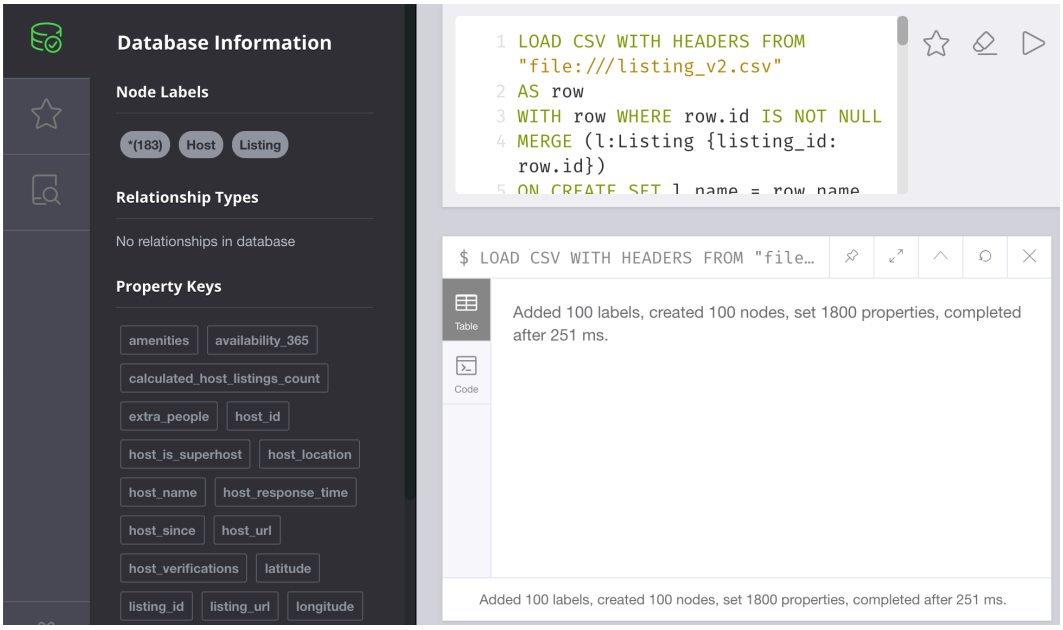
To create the graph database in Neo4j, we first create a project and a new graph in the project of Neo4j Desktop.



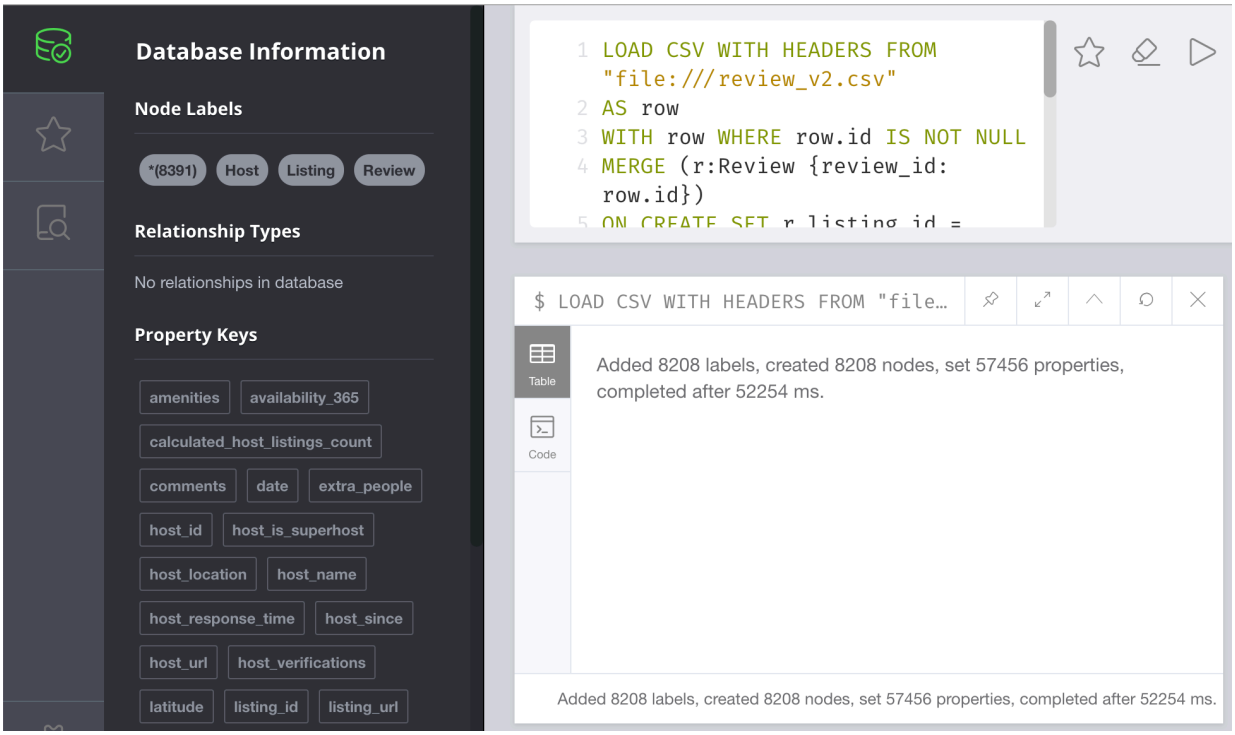
After creating the graph project, we can create the graph database includes nodes and relationships. Firstly, we create a node of host in the Neo4j by using Cypher to import the csv data file into the database. There are 83 nodes and 664 properties stored in the host.



Then, we create the node of Listing in the graph database using Cypher to import the csv data file into the database. There are 100 nodes, 100 labels and 1800 properties stored in the listing.

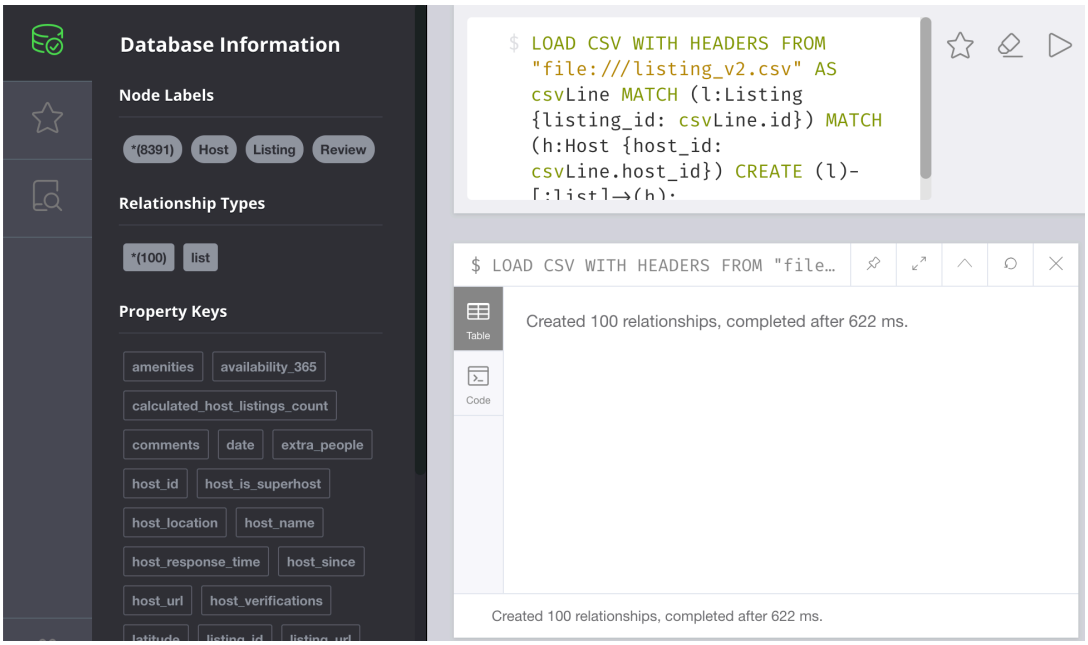


Finally, we create the node of Review in the graph database using Cypher to import the csv data file into the database. There are 8208 nodes. 8208 labels and 57456 properties stored in the review.

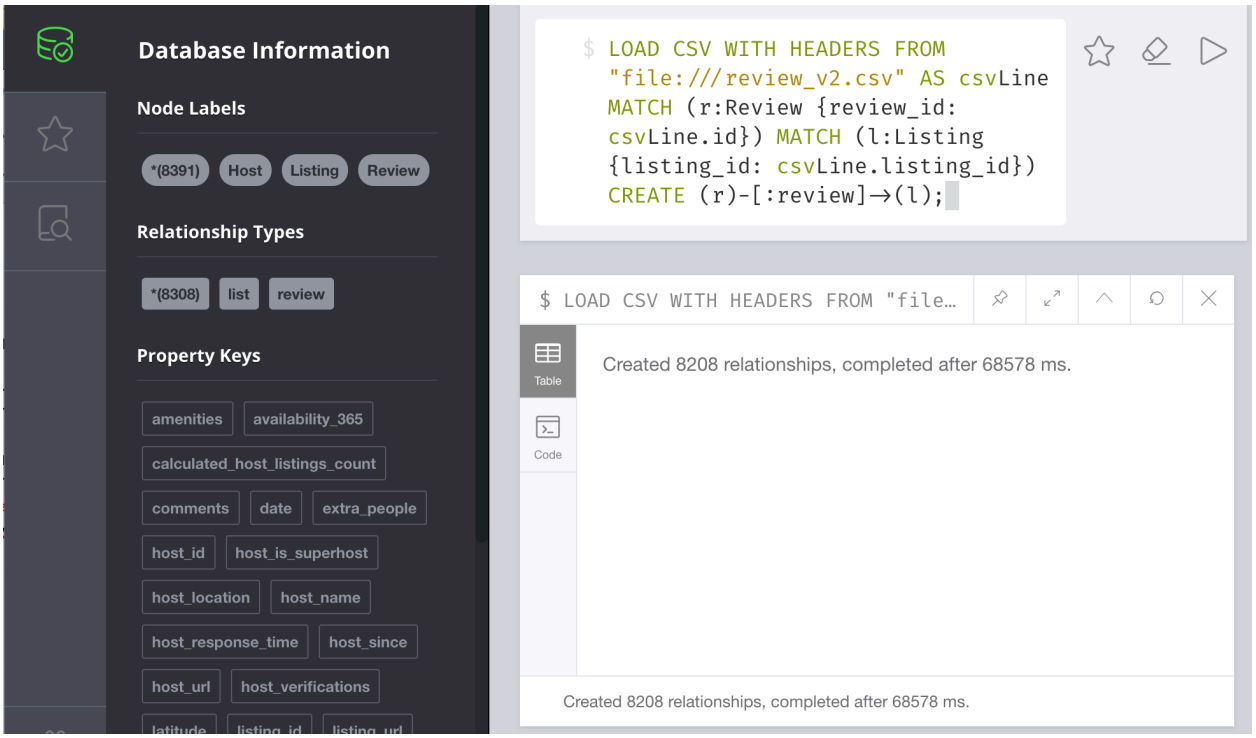


Now, we create the basic relationships between the nodes.

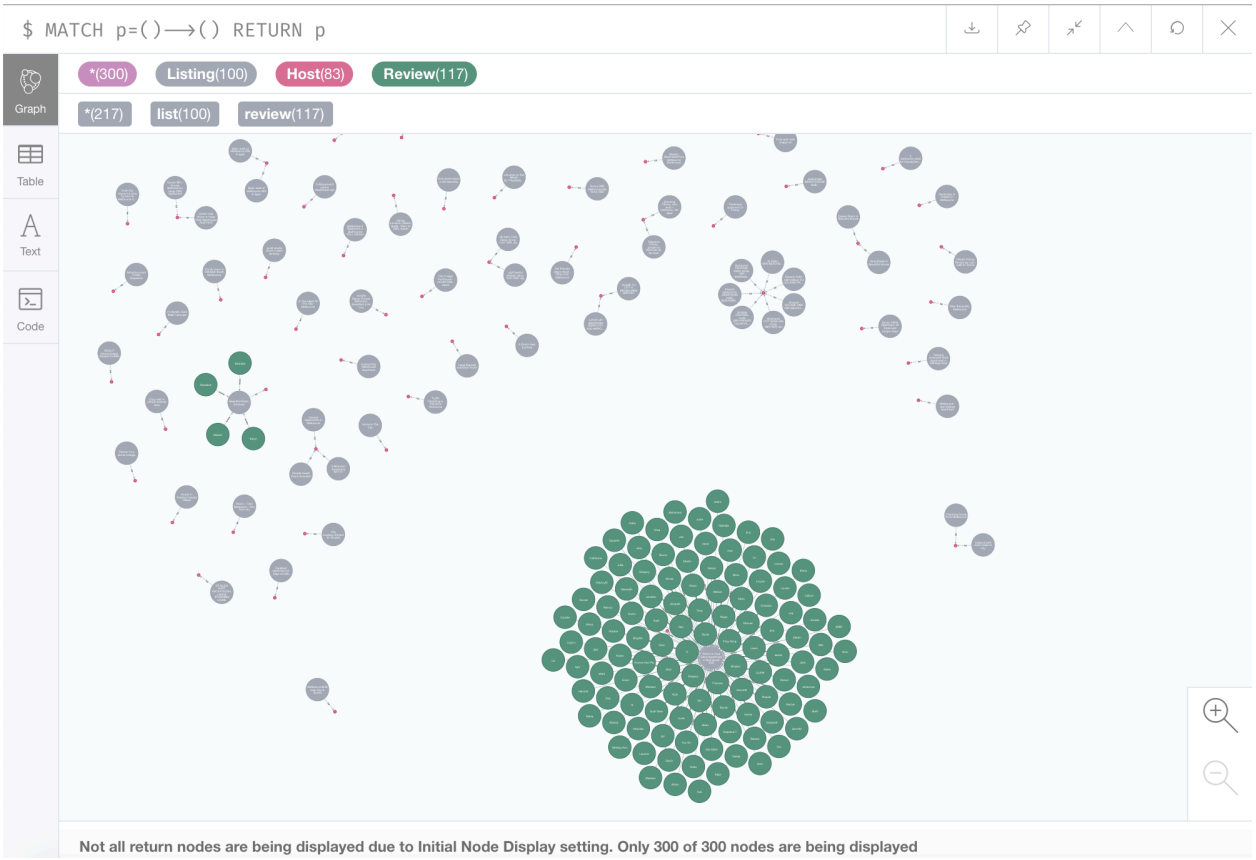
The first relationships were created between the host and listing nodes. There are 100 relationships between each node we created for host and listing.



The next relationships were created between review and listing nodes, there are 8208 relationships between nodes of review and listing.

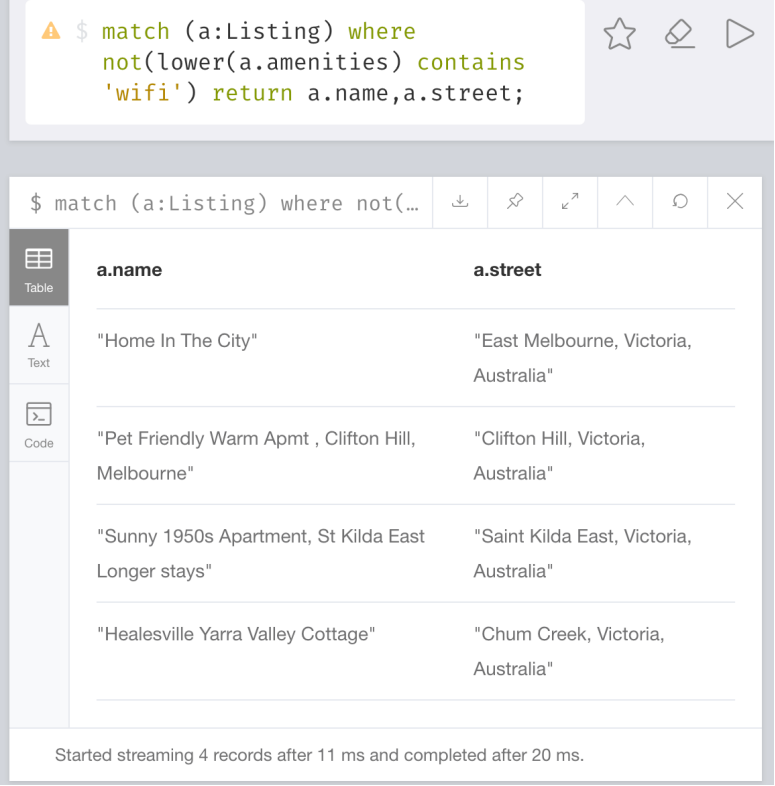


After creating the nodes and relationships of the data, we can see the general graph of the database (because the graph database is big, and there are 300 nodes displayed):



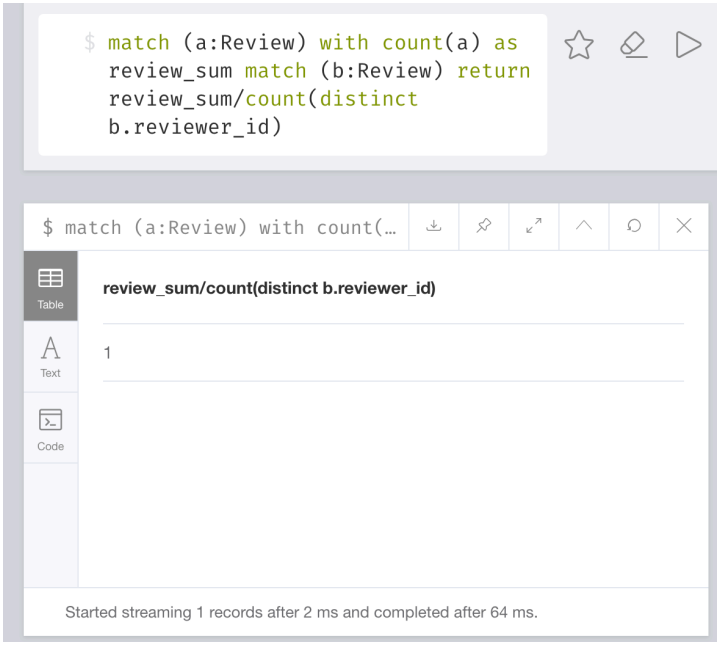
Analysis of Airbnb Data

1. List all accommodation names and locations that do not provide Wi-Fi.



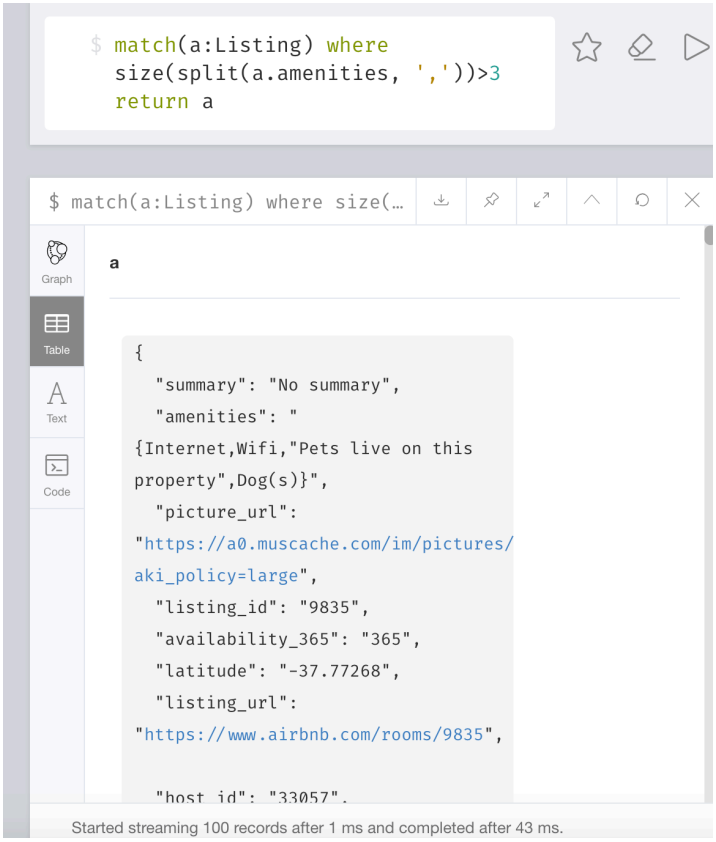
Analysis this task is using for the guests who may want to choose an accommodation has or has not Wi-Fi as amenity. From the analysis result, we can see there are 4 records of the accommodations do not provide Wi-Fi, we can conclude that all most accommodations have Wi-Fi as their amenity. For the guests who want to book a host provides Wi-Fi, they could choose the accommodations except these 4 hosts in the result.

2. How many times a reviewer left reviews:



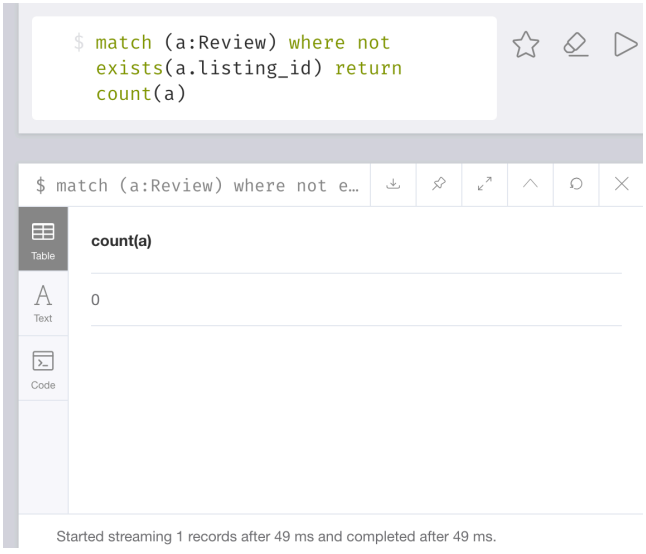
From the analysis, we can conclude that typically one reviewer will leave only one review for the accommodation. In this case, we could see that the reviews from the reviewers already include all of their own experiences of staying in the host.

3. List the pairs of accommodations that have more than three amenities in common



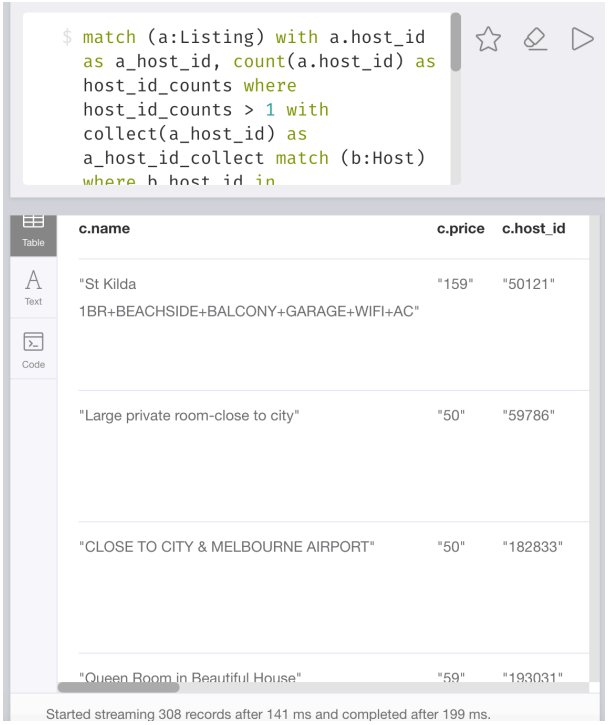
According to this analysis, we could get the results of the accommodations which have comparably perfect amenities for their guests. When an accommodation has at least three amenities, it may have the good living standards for their guests.

4. The listings do not have any review:

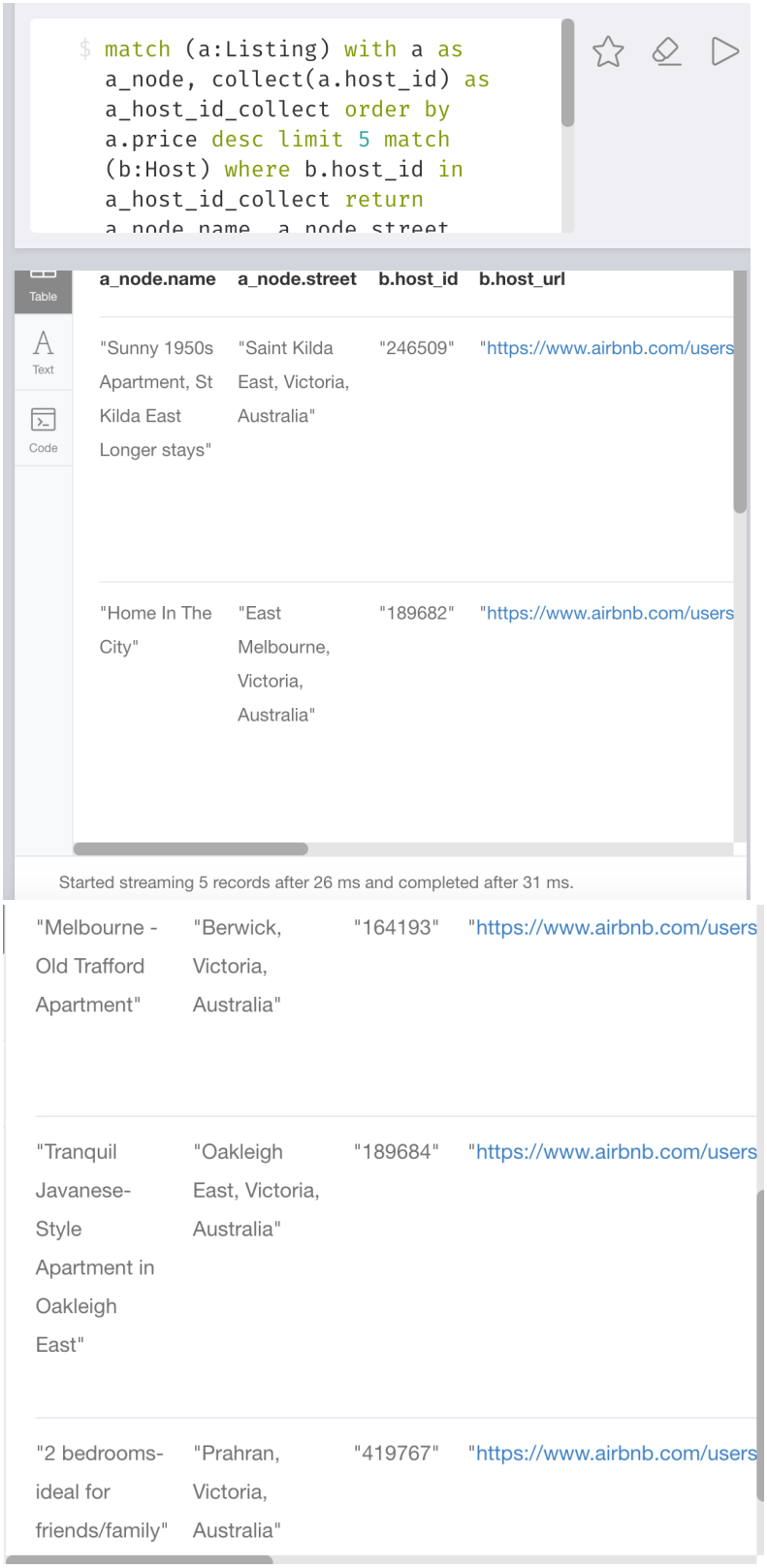


This analysis is used to find out how many listings do not have any review. When we got a “0” here, which mean every listing has reviews and make sure guests could view some reviews of the host they like before they book online.

5. Hosts have multiple listings, include host and listing details, such as name and price: This result analysis the details of the hosts who have multiple listings.



6. The top 5 most expensive accommodations, include locations, host information, and names of the accommodation:



According to this analysis, we could get the detailed results of the top 5 expensive accommodations.

7. The top 10 most popular neighborhoods based on the total average review score ratings:

```
$ match (a:Listing) with
 collect(distinct(a.neighbourhood
)) as a_neighbourhood_collect
match (b:Listing) where
b.neighbourhood in
a_neighbourhood_collect return
b.neighbourhood
```

$ match (a:Listing) with colle…

Table | Text | Code

b.neighbourhood	avg(tointeger(b.price))
"Yarra Ranges"	238.0
"Glen Eira"	191.66666666666669
"Melbourne"	176.34999999999997
"Bayside"	169.5
"Port Phillip"	134.5
"Yarra"	131.1875
"Hobsons Bay"	86.33333333333334
"Stonnington"	85.5
"Casey"	79.0
"Moreland"	72.75

Started streaming 10 records after 41 ms and completed after 41 ms.

This analysis includes the top 10 most popular neighborhoods who have comparable high average reviewing score. We can conclude that which areas are the most popular places.

8. Hosts whose location are different from their listings:

```
$ match (a:Listing) with collect(distinct(a.street)) as
a_street_collect match  (b:Listing) where b.street in
a_street_collect with b.street as b_street_collect,
collect(b.name) as b_name_collect , collect(b.host_id) as
b_host_id_collect match (c:Host) where c.host_id in
b_host_id_collect and not (c.host_location =
b.street collect) return c.host_name, c.host_location
```

$ match (a:Listing) with collect(distinct(a.street)) as a_st…

Table | Text | Code

c.host_name	c.host_location	b_name_collect	b_street_collect
"Manju"	"Albert Park, Victoria, Australia"	["Beautiful Room & House"]	"Bulleen, Victoria, Australia"
"Lindsay"	"Melbourne, Victoria, Australia"	["Room in Cool Deco Apartment in Brunswick East"]	"Brunswick East, Victoria, Australia"
"The A2C Team"	"Melbourne, Victoria, Australia"	["St Kilda 1BR+BEACHSIDE+BALCONY+GARAGE+WIFI+AC", "St Kilda CENTRAL LUXE 2BR+PRIVATE COURTYARDS+WIFI", "Large room in quiet Victorian home"]	"St Kilda, Victoria, Australia"
"Adrian"	"Melbourne, Victoria, Australia"	["St Kilda 1BR+BEACHSIDE+BALCONY+GARAGE+WIFI+AC", "St Kilda CENTRAL LUXE 2BR+PRIVATE COURTYARDS+WIFI",	"St Kilda, Victoria, Australia"

Started streaming 76 records after 8 ms and completed after 68 ms.

This analysis uses for finding the hosts who have different locations from their listings. According to this analysis, the management of Airbnb could make some business decision such as modification for these hosts who have different address.

Analysis for specific case

1. How many reviews does the host “Sunny 1950s Apartment, St Kilda East” have:

```
match (a:Listing {name:"Sunny 1950s Apartment, St Kilda East
Longer stays"}) -[:review]-(b:Review) return
count(b.review_id)
```

$ match (a:Listing {name:"Sunny 1950s Apartment, St Kilda Ea…

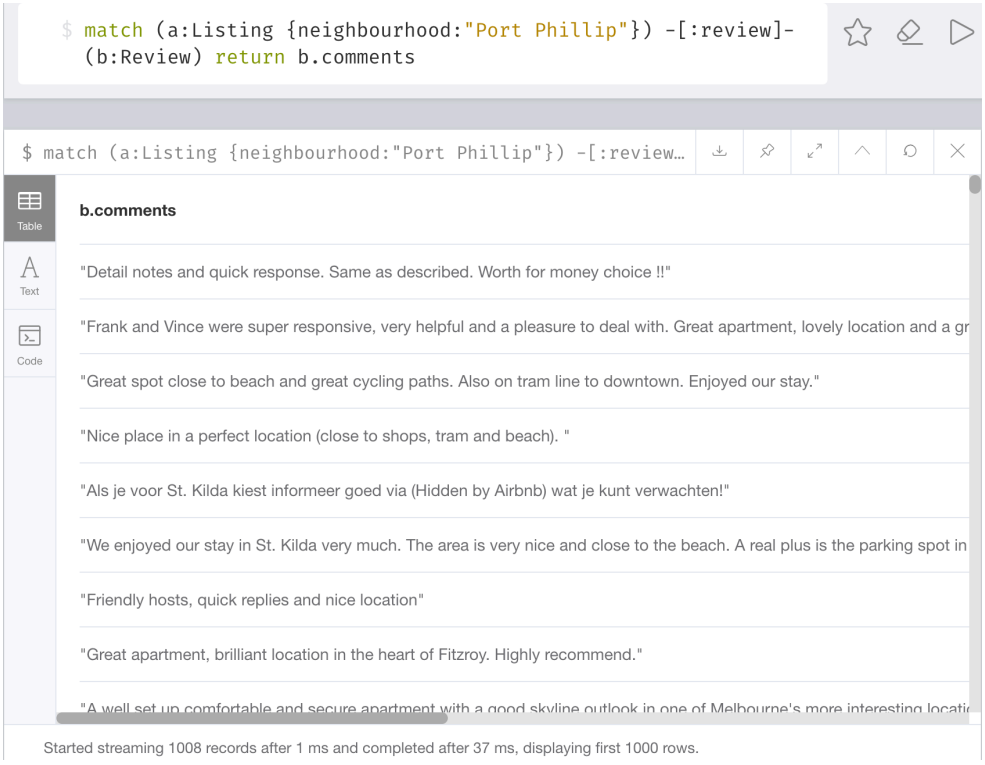
Table | Text | Code

count(b.review_id)
23

Started streaming 1 records after 2 ms and completed after 12 ms.

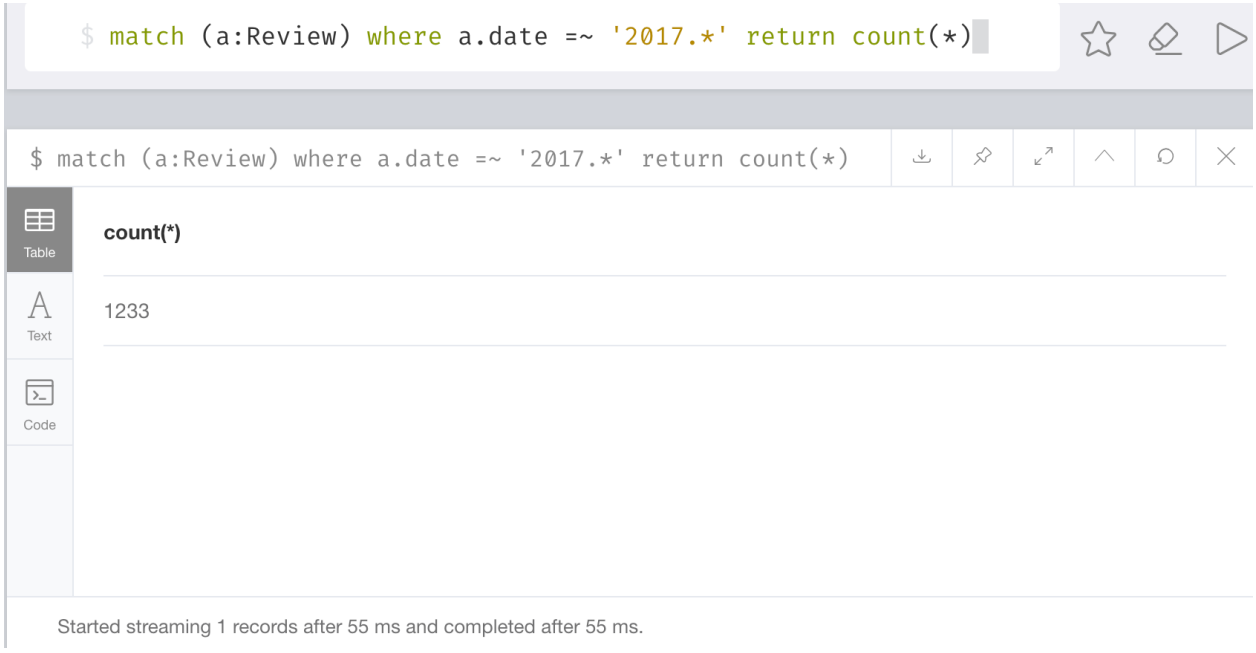
By analysis the specific case like this one, we could analysis how many reviews does the specific host has and conclude the popularity the hosts have.

2. All reviews of the neighborhood of “Port Phillip”:



This query uses for analyzing the reviews of one specific neighborhood. From the result, we could conclude that the review of the neighborhood of "Port Phillip" was almost good review, which means the hosts of this neighborhood and tourism experience of the guests were enjoyable.

3. Accommodations were reviewed in 2017:



By using this query, we could get a count for all reviews in one specific year. We could not only compare the reviews annually but also evaluate the percentage of the host reviewed. For example, we totally have 1233 reviews in 2017 for all hosts and the host “Sunny 1950s Apartment, St Kilda East” have 23 reviews, which means the percentage of review for this host was 1.88%, so this host may not have really good popularity around this neighborhood.

Conclusion

According to the researches, I have basic knowledge on graph database, Neo4j and Cypher query language. Neo4j is a high-performance, NoSQL graph database with vertically structured data stored on the network rather than in tables. Work on an object-oriented, flexible network structure instead of strict, static tables but could also get all the benefits of a completely transactional, enterprise-grade database.

By analyzing the datasets of Airbnb, I have learned how the recommendation engines of Neo4j works for the customers of the business in real-time. Neo4j is not only an advanced database management tool, but also an engine, a search and analysis tool for the business. By using Neo4j, the business could offer user with real-time recommendation to meet their needs, manage the customer relationship to keep the existing customers, and process all of the data into the master data with faster speed.

Reference:

Depeau, Joe. (2019. May). The Seven Bridges of Konigsberg: A Dog's Eye View. Retrieved Nov. 18, 2019

Bruggen, Rik Van. (2014. August). Learning Neo4j. Retrieved Nov.18, 2019

Fernandes, D., Bernardino, Jorge. (2018). Graph Database Comparison: AllegroGraph, ArangoDB, InfiniteGraph, Neo4j, and OrientDB

Rathle, Philip. Marzi, De. Max. (2017). Driving innovation in Retail with Neo4j

Hunger, Michael. Boyd, Ryan. Lyon, William. (2016. March) RDBMS & Graphs: SQL vs. Cypher Query Languages.