10/25/2019

FIT5137 – Advanced Database Technology

Assignment 2
Neo4j

Tutor Name:
Chaluka Salgado

Submitted by:
Abhilash Kale – 30254140

MONASH University
Information Technology

ASSESSMENT COVER SHEET

Student ID number	30254140
Given Name	Abhilash Anil
Family name	Kale

Unit Name and Code:	**FIT5137 – Advanced Database Technology**		
Campus:	**Caulfield**		
Assignment Title:	**Assignment 2 - Neo4j**		
Name of Lecturer:	**Agnes Haryanto**		
Name of Tutor:	**Chaluka Salgado**		
Tutorial Day and Time:	**Monday, 4 PM**		
Phone Number:	**+61-433944038**		
Email Address:	**akal0009@student.monash.edu**		
Has any part of this assignment been previously submitted as part of another unit/course? ☐ Yes ☒ No			
Due Date:	**25/10/2019**	Date Submitted:	**25/10/2019**

All work must be submitted by the due date. If an extension of work is granted this must be specified with the signature of the lecturer/tutor.

Extension granted until (date)_____ Signature of lecturer/tutor _____

Please note that it is your responsibility to retain copies of your assessments.

Intentional plagiarism or collusion amounts to cheating under Part 7 of the Monash University (Council) Regulations

Plagiarism: Plagiarism means taking and using another person's ideas or manner of expressing them and passing them off as one's own. For example, by failing to give appropriate acknowledgement. The material used can be from any source (staff, students or the internet, published and unpublished works).

Collusion: Collusion means unauthorised collaboration with another person on assessable written, oral or practical work and includes paying another person to complete all or part of the work.

Where there are reasonable grounds for believing that intentional plagiarism or collusion has occurred, this will be reported to the Associate Dean (Education) or delegate, who may disallow the work concerned by prohibiting assessment or refer the matter to the Faculty Discipline Panel for a hearing.

Student Statement:

- I have read the university's Student Academic Integrity Policy and Procedures.
- I understand the consequences of engaging in plagiarism and collusion as described in Part 7 of the Monash University (Council) Regulations http://adm.monash.edu/legal/legislation/statutes
- have taken proper care to safeguard this work and made all reasonable efforts to ensure it could not be copied.
- No part of this assignment has been previously submitted as part of another unit/course.
- I acknowledge and agree that the assessor of this assignment may for the purposes of assessment, reproduce the assignment and:
  - i. provide to another member of faculty and any external marker; and/or
  - ii. submit it to a text matching software; and/or
  - iii. submit it to a text matching software which may then retain a copy of the assignment on its database for the purpose of future plagiarism checking.
- I certify that I have not plagiarised the work of others or participated in unauthorised collaboration when preparing this assignment.

Signature: Abhilash Kale Date: 25/10/2019

* delete (iii) if not applicable

The information on this form is collected for the primary purpose of assessing your assignment and ensuring the academic integrity requirements of the University are met. *Other purposes of collection include recording your plagiarism and collusion declaration, attending to course and administrative matters and statistical analyses. If you choose not to complete all the questions on this form it may not be possible for Monash University to assess your assignment. You have a right to access personal information that Monash University holds about you, subject to any exceptions in relevant legislation. If you wish to seek access to your personal information or inquire about the handling of your personal information, please contact the University Privacy Officer: privacyofficer@adm.monash.edu.au*

C.1. Database Design

// HOSTS

LOAD CSV WITH HEADERS FROM "file:///host_v2.csv" AS line

WITH line

WHERE line.host_id IS NOT NULL

MERGE (h: Hosts { hostId: toInt(line.host_id),

hostUrl: line.host_url,

hostName: line.host_name,

hostVerifications: line.host_verifications,

hostSince: Date(line.host_since),

hostLocation: line.host_location,

hostResponseTime: line.host_response_time,

hostIsSuperhost: (case line.host_is_superhost when 't' then true else false end)

})

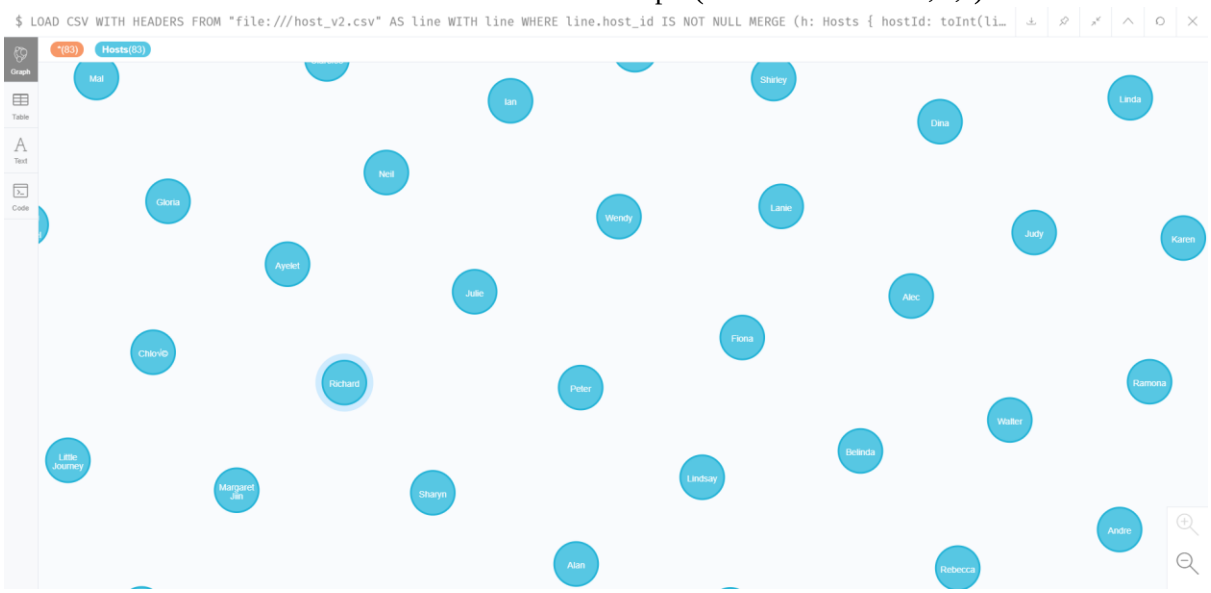
ON CREATE SET h.hostVerifications = replace(h.hostVerifications, "[", ""),

h.hostVerifications = replace(h.hostVerifications, "]", ""),

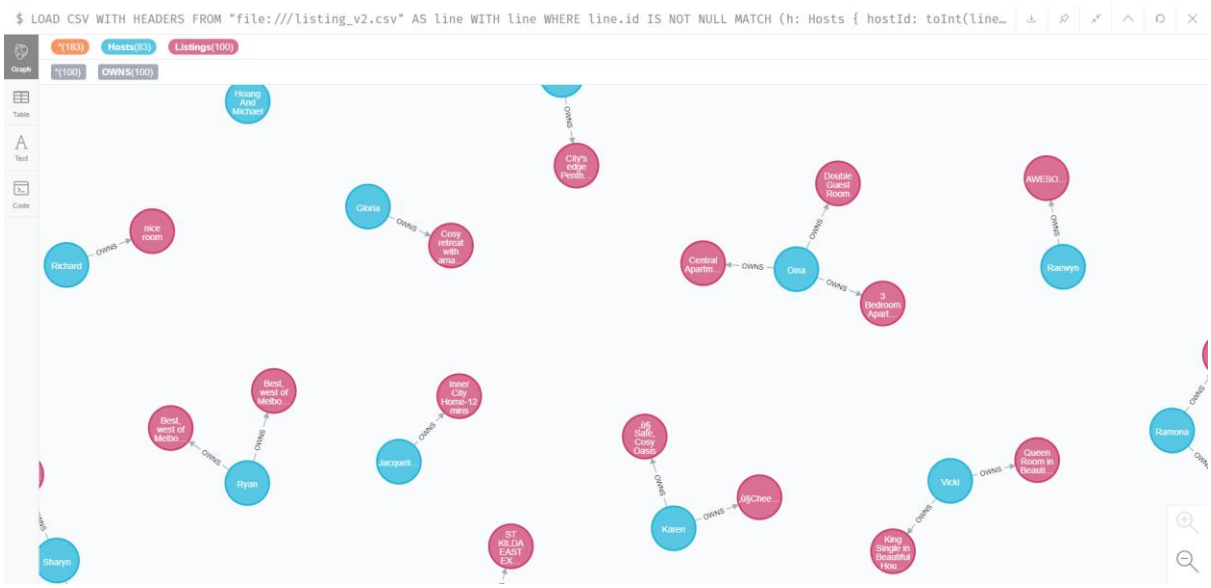
h.hostVerifications = replace(h.hostVerifications, " ", ""),

h.hostVerifications = replace(h.hostVerifications, "'", ""),

h.hostVerifications = split(h.hostVerifications, ",")



```
// LISTINGS

LOAD CSV WITH HEADERS FROM "file:///listing_v2.csv" AS line
WITH line
WHERE line.id IS NOT NULL
MATCH (h: Hosts { hostId: toInt(line.host_id) })
MERGE (l: Listings {    listingId: toInt(line.id),
                                                name: line.name,
                                                summary: line.summary,
                                                listingUrl: line.listing_url,
                                                pictureUrl: line.picture_url,
                                                neighbourhood: line.neighbourhood,
                                                street: line.street,
                                                zipcode: toInt(line.zipcode),
                                                latitude: toFloat(line.latitude),
                                                longitude: toFloat(line.longitude),
                                                roomType: line.room_type,
                                                amenities: line.amenities,
                                                price: toFloat(line.price),
                                                extraPeople: line.extra_people,
                                                minimumNights: toInt(line.minimum_nights),
                                                calculatedHostListingsCount:
toInt(line.calculated_host_listings_count),
                                                availability365: toInt(line.availability_365)
                                                })
ON CREATE SET       l.amenities = replace(l.amenities, "{", "),
                                l.amenities = replace(l.amenities, "}", "),
                                l.amenities = replace(l.amenities, "", "),
                                l.amenities = split(l.amenities, ","),
                                l.extraPeople = replace(l.extraPeople, "$", ""),
                                l.extraPeople = replace(l.extraPeople, " ", ""),
                                l.extraPeople = toFloat(l.extraPeople)
MERGE (h)-[:OWNS]->(l)
```



//REVIEWS

LOAD CSV WITH HEADERS FROM "file:///review_v2.csv" AS line

WITH line

WHERE line.id IS NOT NULL

MATCH (l: Listings { listingId: toInt(line.listing_id) })

MERGE (r: Reviewers { reviewerId: toInt(line.reviewer_id),

reviewerName: line.reviewer_name

})

MERGE (r)-[:REVIEWED { id: toInt(line.id),

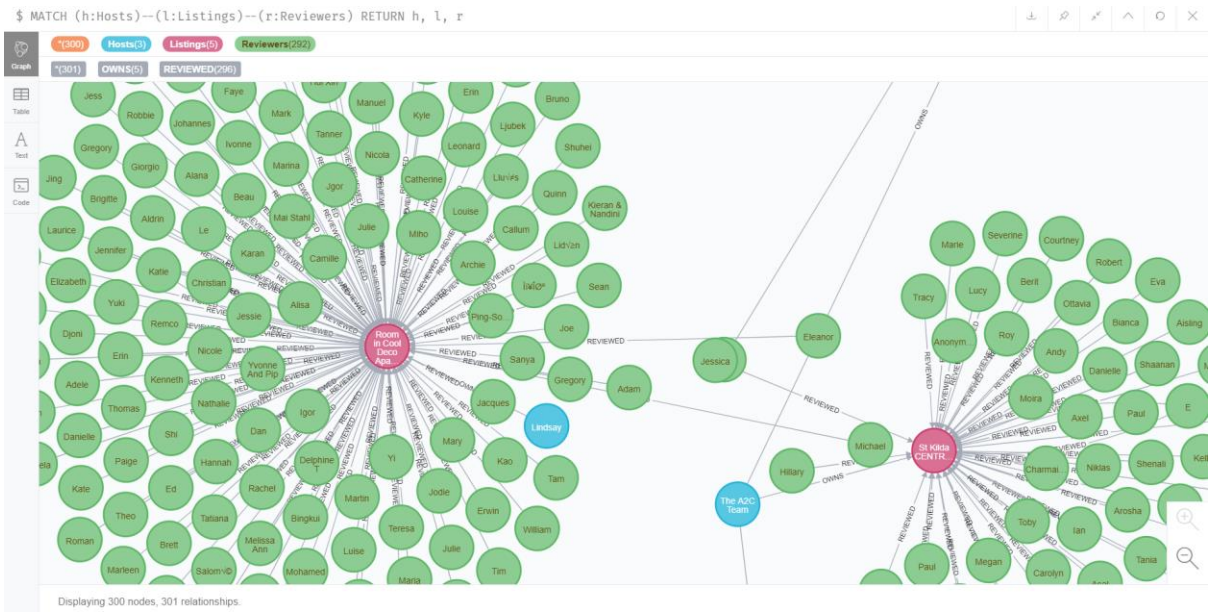
date: Date(line.date),

reviewScoresRating:

toInt(line.review_scores_rating),

comments: line.comments

}]->(l)



Explanation

MonashBnB have provided us with a set of data split into 3 parts, namely, Hosts, Listings and Reviews.

This data has been represented using nodes, Hosts, Listings and Reviewers, with details stored as properties using their appropriate data types.

These nodes are connected by relationships, Hosts-'OWNS'->Listings and Reviewers-'REVIEWED'->Listings with all the review details stored in the 'REVIEWED' relationship.

These sets of edges and vertices together comprise of the whole graph database of the MonashBnB system.

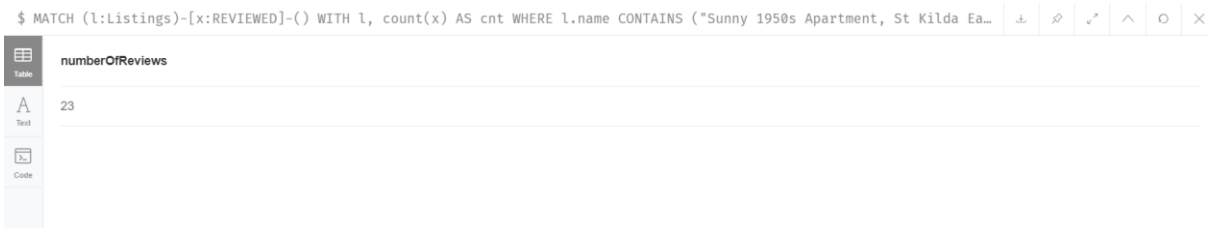
The nodes are linked to each other using their respective ID properties, which makes it easy to run queries leading us to the desired results.

C.2 Queries

C.2.1 Answered Queries

// 1

```
MATCH (l:Listings)-[x:REVIEWED]-()
WITH l, count(x) AS cnt
WHERE l.name CONTAINS ("Sunny 1950s Apartment, St Kilda East")
RETURN cnt AS numberOfReviews
```



// 2

```
MATCH (l:Listings {neighbourhood:'Port Phillip'})-[x:REVIEWED]-(r:Reviewers)
RETURN r AS reviewers, x AS reviews
```

// 3

```
MATCH (r:Reviewers)-[x:REVIEWED]-(l:Listings)
WHERE NOT r.reviewerId = 4162110 AND r.reviewerId = 317848 AND x.reviewScoresRating > 90
RETURN l AS recommendedAccommodations
```

// 4

```
MATCH (l:Listings)

WITH l, collect ( {neighbourhood:l.neighbourhood, street:l.street, zipcode:l.zipcode} ) AS listingLocation

WHERE NOT 'Wifi' IN l.amenities

RETURN l.name AS listingName, listingLocation
```

// 5

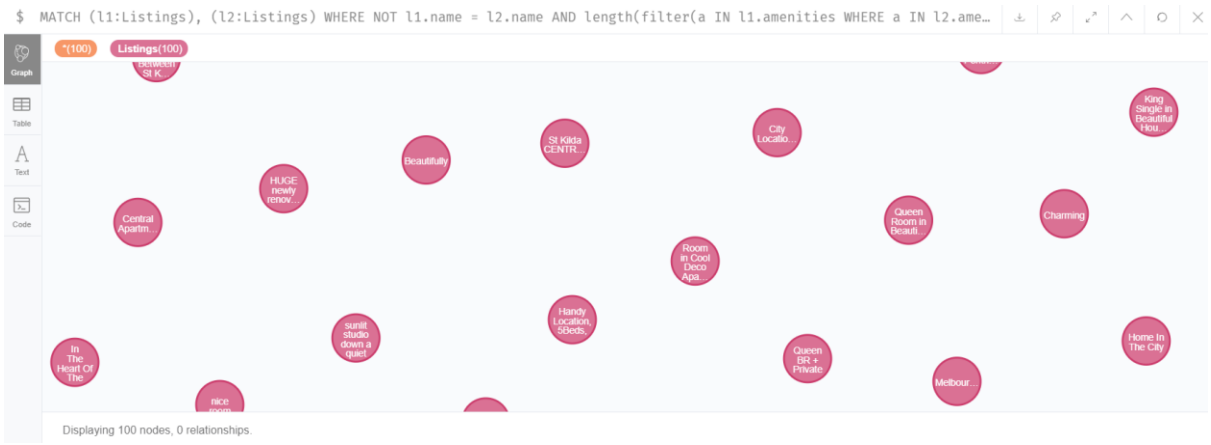
```
MATCH (r:Reviewers)-[x:REVIEWED]-()

RETURN r AS reviewer, count(x) AS numberOfReviews
```

// 6

MATCH (l1:Listings), (l2:Listings)

WHERE NOT l1.name = l2.name AND length(filter(a IN l1.amenities WHERE a IN l2.amenities)) > 3

RETURN l1 AS accommodation1, l2 AS accommodation2



// 7

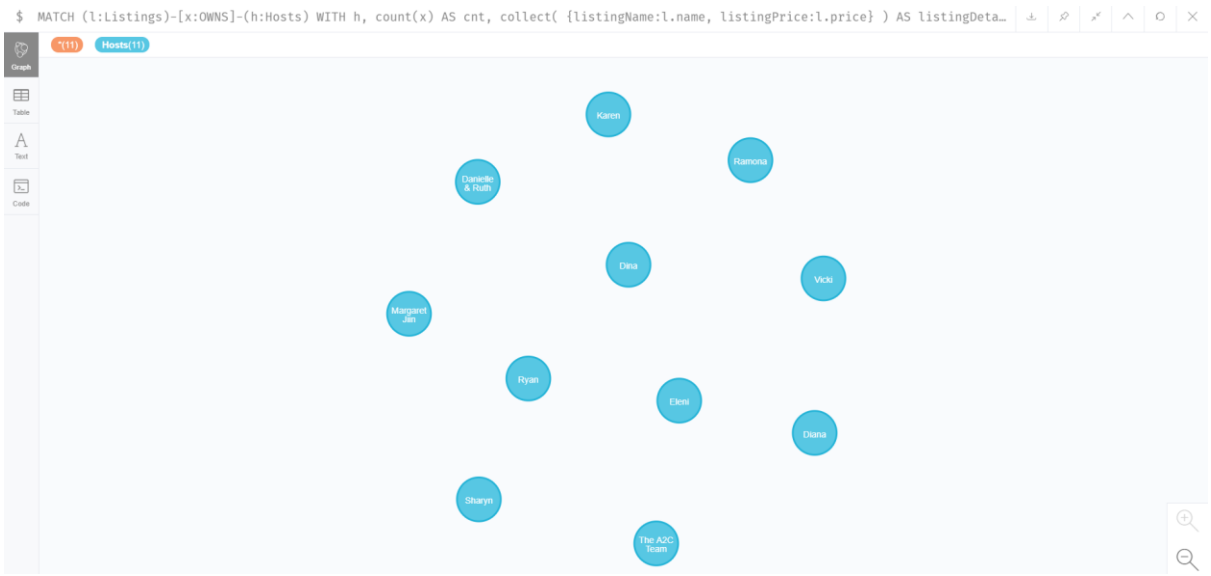
MATCH (l:Listings)

WHERE NOT ()-[:REVIEWED]->(l)

RETURN l AS listings

// 8

```
MATCH (l:Listings)-[x:OWNS]-(h:Hosts)
WITH h, count(x) AS cnt, collect( {listingName:l.name, listingPrice:l.price} ) AS listingDetails
WHERE cnt > 1
RETURN h AS hostDetails, listingDetails
```

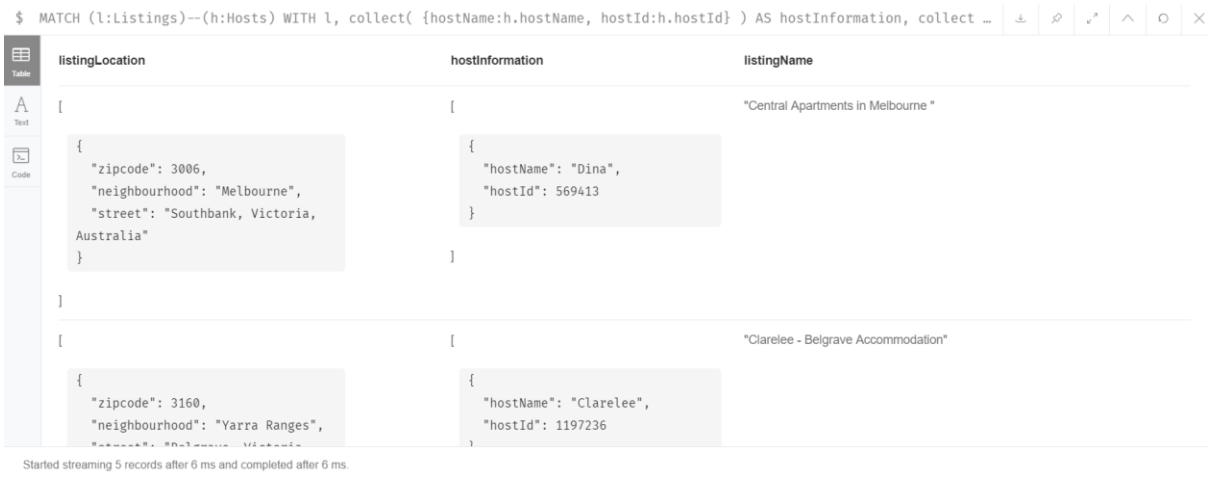


// 9

```
MATCH ( l:Listings {neighbourhood:'Melbourne'} )
RETURN l.neighbourhood AS neighbourhood, avg(l.price) AS averagePrice
```

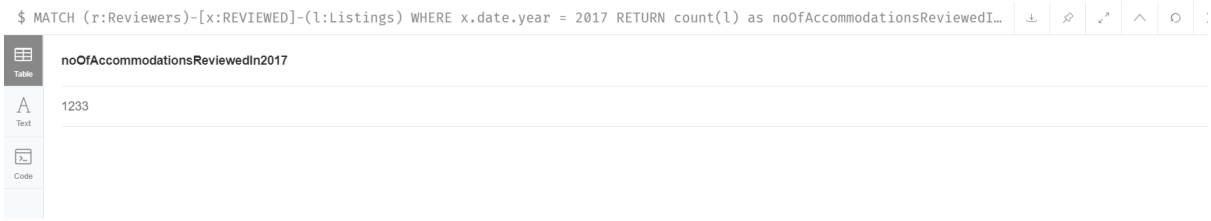
// 10

```
MATCH (l:Listings)--(h:Hosts)
WITH l, collect( {hostName:h.hostName, hostId:h.hostId} ) AS hostInformation, collect ( {neighbourhood:l.neighbourhood, street:l.street, zipcode:l.zipcode} ) AS listingLocation
RETURN listingLocation, hostInformation, l.name AS listingName
ORDER BY l.price DESC
LIMIT 5
```



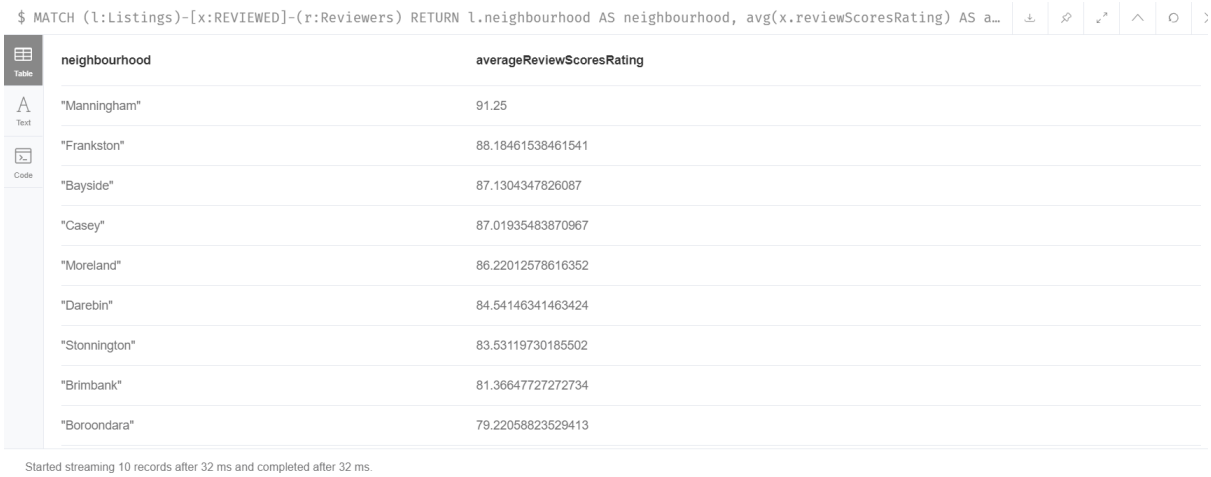
// 11

```
MATCH (r:Reviewers)-[x:REVIEWED]-(l:Listings)
WHERE x.date.year = 2017
RETURN count(l) as noOfAccommodationsReviewedIn2017
```



// 12

```
MATCH (l:Listings)-[x:REVIEWED]-(r:Reviewers)
RETURN l.neighbourhood AS neighbourhood, avg(x.reviewScoresRating) AS
averageReviewScoresRating
ORDER BY averageReviewScoresRating DESC
LIMIT 10
```



// 13

```
MATCH (h:Hosts)--(l:Listings)

WITH h, l, collect ( {neighbourhood:l.neighbourhood, street:l.street, zipcode:l.zipcode} ) AS
listingLocation

WHERE NOT h.hostLocation = l.street

RETURN h.hostName AS hostName, h.hostLocation AS hostLocation, l.name AS listingName,
listingLocation
```

// 14

```
MATCH (l:Listings)

WITH l, collect ( {neighbourhood:l.neighbourhood, street:l.street, zipcode:l.zipcode} ) AS
listingLocation

RETURN l.name AS listingName, listingLocation, l.price AS pricePerNight, l.extraPeople AS
extraPeopleCharge, 5 * (l.price + (2 * l.extraPeople)) AS totalPrice

ORDER BY totalPrice
```

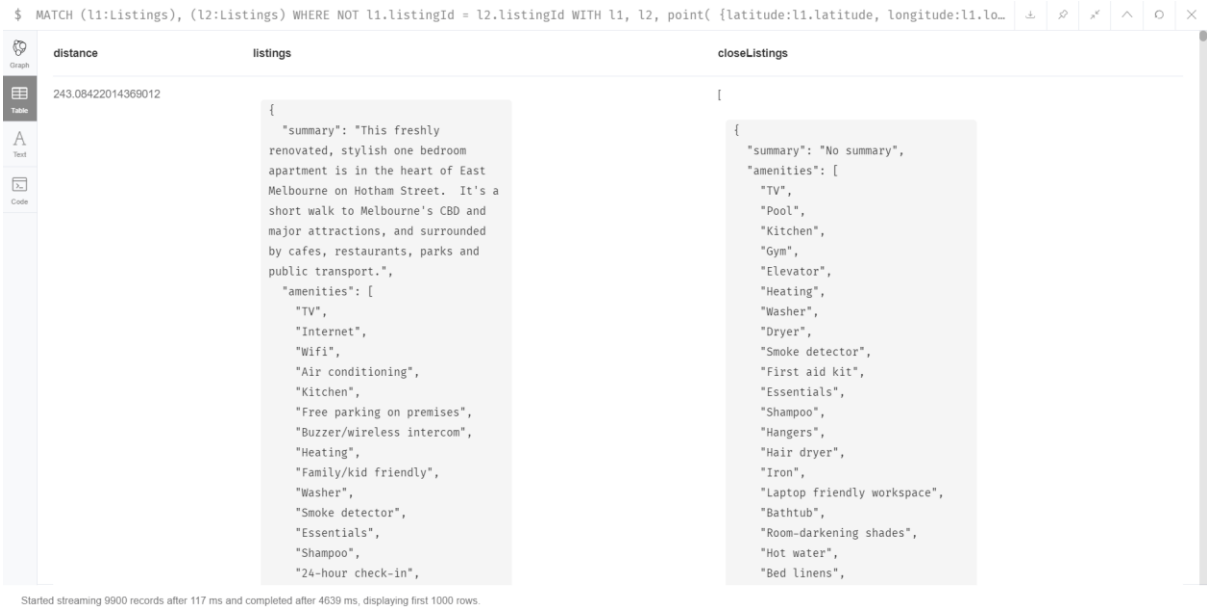
$ MATCH (l:Listings) WITH l, collect ({neighbourhood:l.neighbourhood, street:l.street, zipcode:l.zipcode}) AS listingLocation RETUR…

listingName	listingLocation	pricePerNight	extraPeopleCharge	totalPrice
"Warm and inviting cottage in the North East"	[{ "zipcode": 3094, "neighbourhood": "Banyule", "street": "Montmorency, Victoria, Australia" }]	30.0	0.0	150.0
"Kew Tranquility, Melbourne"	[{ "zipcode": 3101, "neighbourhood": "Boroondara", "street": "Kew, Victoria, Australia" }]	45.0	0.0	225.0
"Convenient Spot in Mt Waverley"	[{ "zipcode": 3149,	45.0	0.0	225.0

Started streaming 100 records after 4 ms and completed after 5 ms.

// 15

```
MATCH (l1:Listings), (l2:Listings)
WHERE NOT l1.listingId = l2.listingId
WITH l1, l2, point( {latitude:l1.latitude, longitude:l1.longitude} ) AS p1, point( {latitude:l2.latitude, longitude:l2.longitude} ) AS p2
RETURN distance(p1, p2) AS distance, l1 AS listings, collect(l2) AS closeListings
ORDER BY l1.name, distance
```

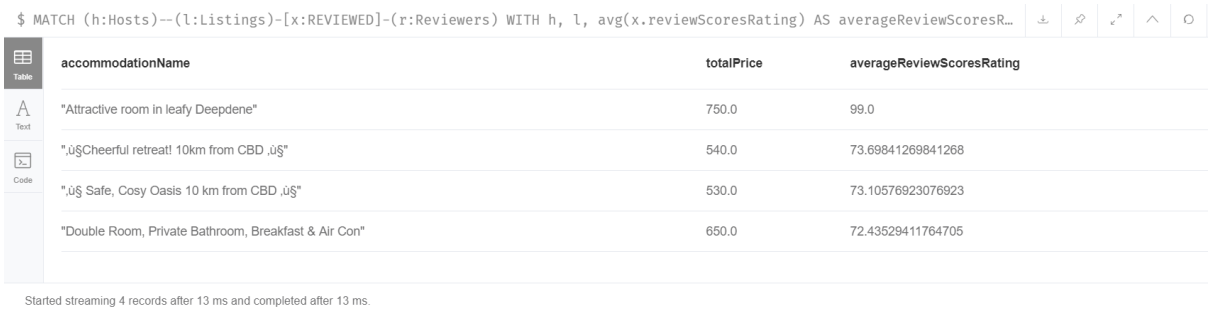


C.2.2 Additional Queries

// 1

// Assuming that the accommodation price mentioned is for 2 people per night, find out all the accommodation names whose hosts are from Fitzroy and calculate the price for each accommodation for 2 people staying for 10 nights, with 1 extra person joining and staying for the last 2 days. Sort the accommodations with their average review ratings score, displaying the highest one first.

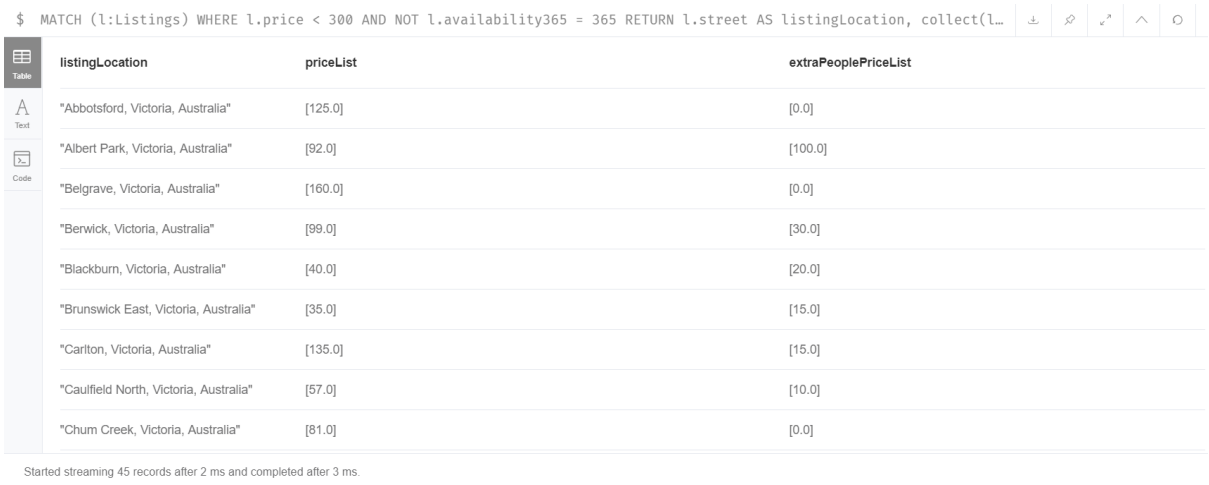
```
MATCH (h:Hosts)--(l:Listings)-[x:REVIEWED]-(r:Reviewers)
WITH h, l, avg(x.reviewScoresRating) AS averageReviewScoresRating
WHERE h.hostLocation STARTS WITH 'Fitzroy'
RETURN l.name AS accommodationName, (8 * l.price) + (2 * (l.price + l.extraPeople)) as totalPrice, averageReviewScoresRating
ORDER BY averageReviewScoresRating DESC
```



// 2

// Display the unique listing locations whose prices per night are less than $300 and are not available for 365 days. Display all the prices (per night and extra people charge) for each location. Sort alphabetically with the location.

```
MATCH (l:Listings)
WHERE l.price < 300 AND NOT l.availability365 = 365
RETURN l.street AS listingLocation, collect(l.price) AS priceList, collect(l.extraPeople) AS extraPeoplePriceList
ORDER BY listingLocation
```



```
$ MATCH (l:Listings) WHERE l.price < 300 AND NOT l.availability365 = 365 RETURN l.street AS listingLocation, collect(l…
```

Table | Text | Code

listingLocation	priceList	extraPeoplePriceList
"Abbotsford, Victoria, Australia"	[125.0]	[0.0]
"Albert Park, Victoria, Australia"	[92.0]	[100.0]
"Belgrave, Victoria, Australia"	[160.0]	[0.0]
"Berwick, Victoria, Australia"	[99.0]	[30.0]
"Blackburn, Victoria, Australia"	[40.0]	[20.0]
"Brunswick East, Victoria, Australia"	[35.0]	[15.0]
"Carlton, Victoria, Australia"	[135.0]	[15.0]
"Caulfield North, Victoria, Australia"	[57.0]	[10.0]
"Chum Creek, Victoria, Australia"	[81.0]	[0.0]

Started streaming 45 records after 2 ms and completed after 3 ms.

// 3

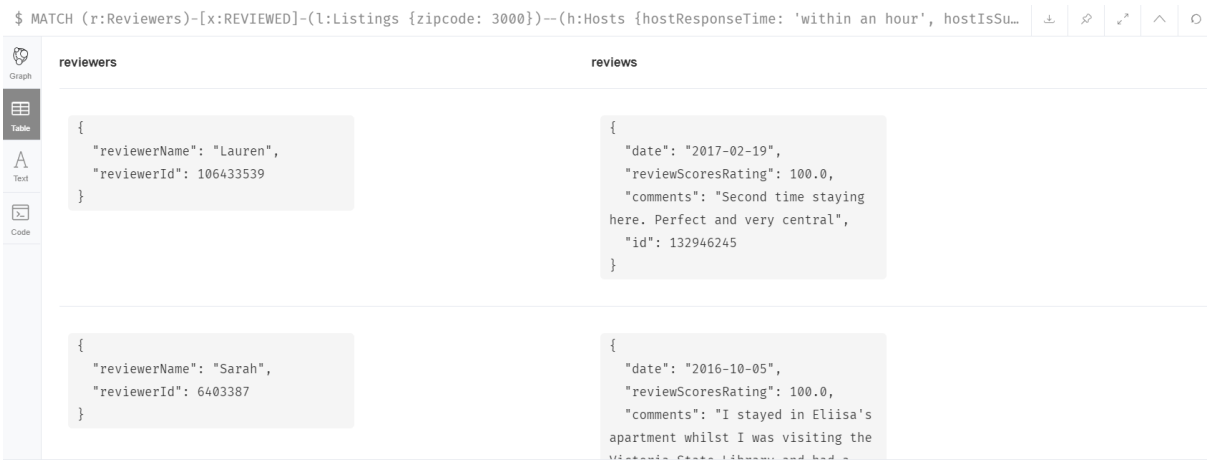
// Find the host details of the listings having either 'Oven' or 'TV' as one of their amenities. Also display the listing name along with its location and price per night.

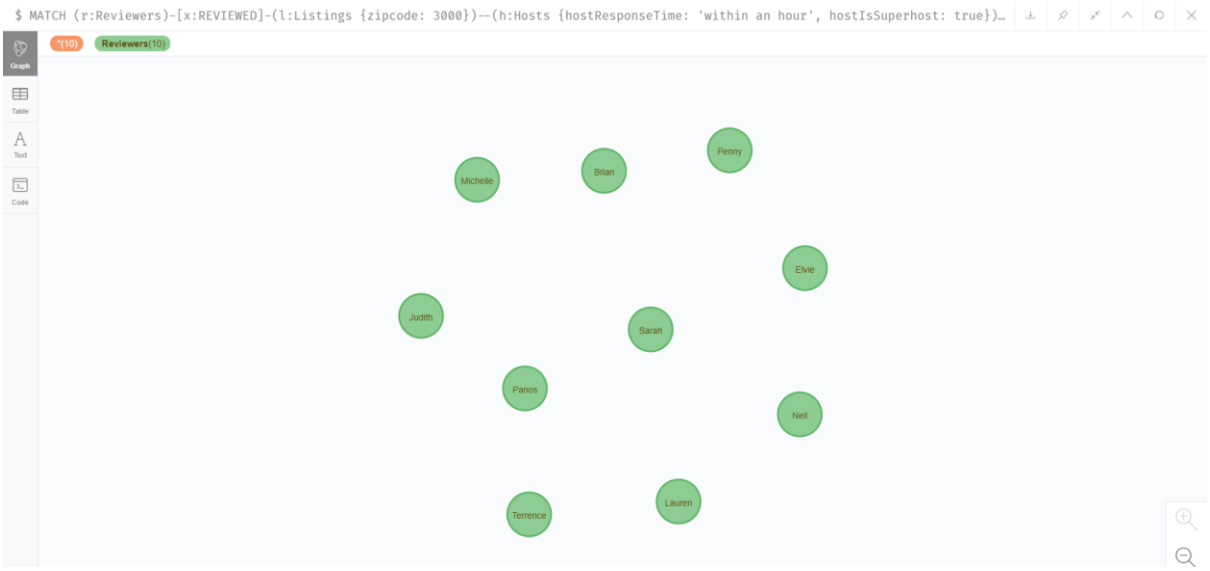
```
MATCH (l:Listings)--(h:Hosts)

UNWIND l.amenities AS amenities1

WITH l, h, amenities1, ['Oven', 'TV'] AS amenities2, collect( {hostId: h.hostId, hostName: h.hostName} ) AS hostDetails, collect ( {neighbourhood:l.neighbourhood, street:l.street, zipcode:l.zipcode} ) AS listingLocation

WHERE amenities1 IN amenities2

RETURN DISTINCT hostDetails, l.name AS listingName, listingLocation, l.price AS pricePerNight
```

// 4

// Extract the top 10 reviews based on the score rating received, which are 2 to 4 years old whose listing's zipcode is 3000, and its host responds within an hour and also is a superhost.

```
MATCH (r:Reviewers)-[x:REVIEWED]-(l:Listings {zipcode: 3000})--(h:Hosts {hostResponseTime: 'within an hour', hostIsSuperhost: true})

WITH r, x, date() AS currentDate

WHERE (currentDate.year - x.date.year) > 1 AND (currentDate.year - x.date.year) < 5

RETURN r AS reviewers, x AS reviews

ORDER BY x.reviewScoresRating DESC

LIMIT 10
```

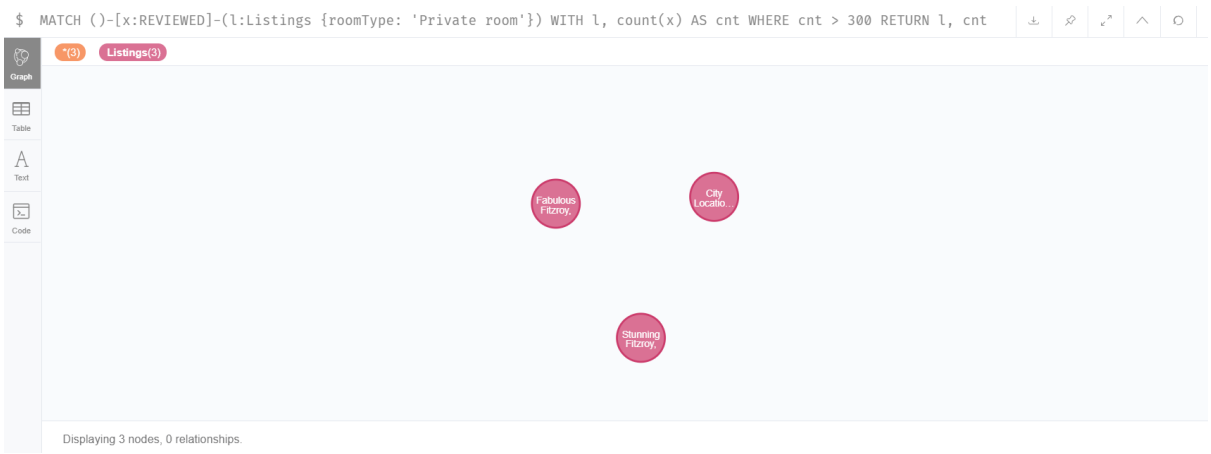




// 5

// Find the listings with room type as private room and have more than 300 reviews each.

```
MATCH ()-[x:REVIEWED]-(l:Listings {roomType: 'Private room'})
WITH l, count(x) AS cnt
WHERE cnt > 300
RETURN l, cnt
```



C.2.3 Indices

CREATE INDEX ON: Listings(neighbourhood)

CREATE INDEX ON: Listings(amenities)

CREATE INDEX ON: Listings(street, price)

Justification

Indices have been chosen according to the occurrences of the properties of the nodes in the queries.

Being a travel accommodation booking system, the major search criteria runs based on the neighbourhood, amenities and review ratings that listings have received.

To begin a search query for travel booking, location and price, together are key properties of the listings or accommodations, and hence, are used to create a composite index.

C.3 Database Modifications

// 1

//I

MERGE (h: Hosts { hostId: 111111,

hostUrl: "https://www.airbnb.com.au/users/show/106130712",

hostName: 'Frank and Justine',

hostVerifications: ['Government ID', 'Email address', 'Phone number'],

hostSince: Date("2016-10-24"),

hostLocation: 'Melbourne, Victoria, Australia',

hostResponseTime: 'within an hour',

hostIsSuperhost: true

})

-[:OWNS]-> (l: Listings { listingId: 1111111,

name: 'Executive Luxury Condo with City and Lake Views',

summary: "Take in the sprawling views of Albert Park Lake and Port Phillip Bay from the balcony of this spacious and tastefully styled apartment. Unwind in a contemporary design space with contrasting black and white throughout and access to a building gym.",

listingUrl: "https://www.airbnb.com.au/rooms/plus/17914380?check_in=2019-10-30&check_out=2019-10-31&source_impression_id=p3_1571818039_AdscHndShYsAkLAt",

pictureUrl: "https://www.airbnb.com.au/rooms/17914380?s=67&shared_item_type=1&virality_entry_point=1",

neighbourhood: 'Melbourne',

street: 'Melbourne, Victoria, Australia',

zipcode: 3000,

latitude: -37.8155,

longitude: 144.95894,

roomType: 'Entire home/apt',

amenities: ['Self check_in', 'Gym', 'Kitchen', 'Washing machine', 'Dryer', 'Wifi', 'Coffee maker'],

price: 328.87,

extraPeople: 50.0,

minimumNights: 1,

calculatedHostListingsCount: 1,

availability365: 365

})

<-[:REVIEWED { id: 12345,

date: Date("2019-09-27"),

reviewScoresRating: 99,

comments: "This apartment has everything you need for a short or long stay. The views are magnificent and peaceful. Located on St Kilda Road with easy access and parking. The tram stop is just across the street making it extremely easy to travel to city or St Kilda."

}]- (r: Reviewers { reviewerId: 11111,

reviewerName: 'Lyn'

})

//II

MATCH (l: Listings { listingId: 1111111 })

MERGE (r: Reviewers { reviewerId: 11112,

reviewerName: 'Monika'

})

-[:REVIEWED { id: 12346,

date: Date("2019-08-15"),

reviewScoresRating: 81,

comments: "Beautiful apartment with all that you need for a weekend stay (or longer). Amazing location and breathtaking view!! Wanted to stay longer and will absolutely be back."

}]->(l)

//III

```
MERGE (h: Hosts {       hostId: 222222,
                                        hostUrl: "https://www.airbnb.com.au/users/show/15874351",
                                        hostName: 'Donna',
                                        hostVerifications: ['Government ID', 'Selfie', 'Email address',
'Phone number'],
                                        hostSince: Date("2014-06-20"),
                                        hostLocation: 'Melbourne, Victoria, Australia',
                                        hostResponseTime: 'within few hours',
                                        hostIsSuperhost: true
                                        })
-[:OWNS]->
                (l: Listings {      listingId: 2222222,
                                                name: 'Designer Suite in Central Deco Icon',
                                                summary: "Commanding views across historic City
Hall and Paris End Skyscrapers, this centrally located apartment sits within the famous 1924 early
modernist/Art Deco Capitol Theatre Building, designed by Frank Lloyd Wright trained architect,
Walter Burley Griffin. Griffin is renowned for his design for Australia's capital city, Canberra.",
                                                listingUrl:
"https://www.airbnb.com.au/rooms/3125217?check_in=2019-10-30&check_out=2019-10-
31&source_impression_id=p3_1571824450_ApuC%2F77nrJQBAazT",
                                                pictureUrl:
"https://www.airbnb.com.au/rooms/3125217/slideshow/25339875?check_in=2019-10-
30&check_out=2019-10-31&adults=1&children=0&infants=0",
                                                neighbourhood: 'CBD',
                                                street: 'Melbourne, Victoria, Australia',
                                                zipcode: 3000,
                                                latitude: -37.7985,
                                                longitude: 144.97883,
                                                roomType: 'Entire home/apt',
                                                amenities: ['Lift', 'Breakfast', 'Wifi', 'Kitchen'],
                                                price: 216.96,
                                                extraPeople: 20.0,
                                                minimumNights: 2,
                                                calculatedHostListingsCount: 2,
                                                availability365: 365
```

```
                                                    })
<-[:REVIEWED {          id: 23456,
                                    date: Date("2019-10-06"),
                                    reviewScoresRating: 100,
                                    comments: "Location, location. Donna's apartment is a wonderful,
clean, quiet place right in the heart of Melbourne. We spent a comfortable week in the stylish
apartment with all the amenities you could require. We will definitely stay again."
}]-         (r: Reviewers {  reviewerId: 22222,
                                            reviewerName: 'Karen'
                                            })



//IV


MATCH (l: Listings { listingId: 2222222 })
MERGE (r: Reviewers {           reviewerId: 22223,
                                                reviewerName: 'Victor'
                                            })
-[:REVIEWED {           id: 23457,
                                    date: Date("2019-09-19"),
                                    reviewScoresRating: 100,
                                    comments: '10 out of 10'
                                    }]->(l)



//V


MATCH (h: Hosts {hostId: 222222 })
MERGE (l: Listings {    listingId: 3333333,
                                            name: 'Stylish Room in the City Heart',
                                            summary: "LOCATION AND COMFORT! Your
own private room in my home in Melbourne's Centre - only a 10-12 minute walk from the city's main
transport terminal and shopping. The apartment is in the classy financial district of Melbourne's
downtown - with historical character with the convenience of clean, modern interior. Some of the
```

city's best cafes close by, and free tram transport at the doorstep. You will be right at the centre of the action in this vibrant city!",

listingUrl: "https://www.airbnb.com.au/rooms/3163595?source_impression_id=p3_1571825313_%2BAy5b2zgNGVoKQje",

pictureUrl: "https://www.airbnb.com.au/rooms/3163595/slideshow/653308813?adults=1&children=0&infants=0",

neighbourhood: 'Richmond',

street: 'Richmond, Victoria, Australia',

zipcode: 3121,

latitude: -37.81973,

longitude: 145.00078,

roomType: 'Entire home/apt',

amenities: ['Lift', 'Breakfast', 'Wifi', 'Kitchen', 'Air conditioning', 'Hot water', 'Dishwasher', 'Bed and bath'],

price: 79.0,

extraPeople: 0.0,

minimumNights: 1,

calculatedHostListingsCount: 2,

availability365: 365

})<-[:OWNS]-(h)

//VI

MATCH (l: Listings { listingId: 3333333 })

MERGE (r: Reviewers { reviewerId: 33333,

reviewerName: 'Alice'

})

-[:REVIEWED { id: 34567,

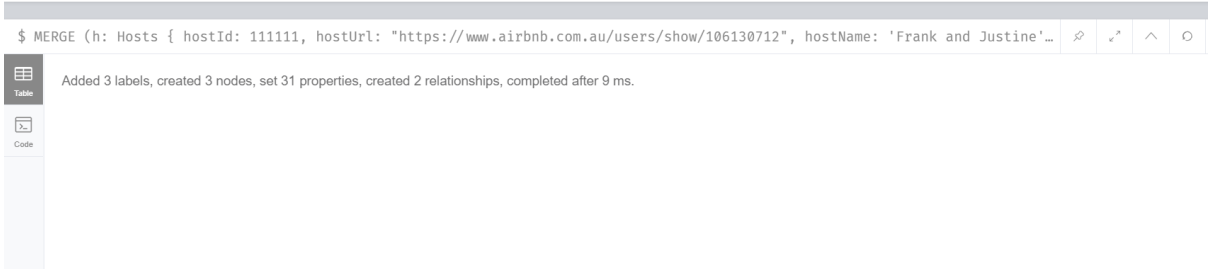
date: Date("2019-08-18"),

reviewScoresRating: 95,

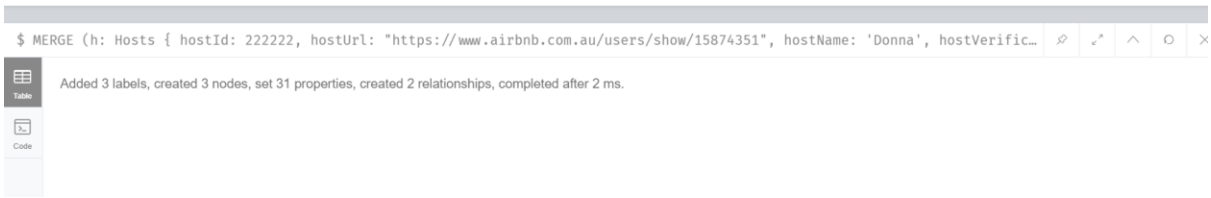
comments: "Located in the heart of the city, stylish comfy room in a great apartment, and an excellent host. What more could you wish for!"

```
                                        }]->(l)


//VII


MATCH (l: Listings { listingId: 3333333 })
MERGE (r: Reviewers {           reviewerId: 33334,
                                                reviewerName: 'Cindy'
                                                })
-[:REVIEWED {           id: 34568,
                                        date: Date("2019-08-04"),
                                        reviewScoresRating: 100,
                                        comments: "10/10 would recommend. Easy communication, excellent location and nothing but amazing."
                                        }]->(l)


//VIII


MATCH (l: Listings { listingId: 3333333 })
MERGE (r: Reviewers {           reviewerId: 33335,
                                                reviewerName: 'Alana'
                                                })
-[:REVIEWED {           id: 34569,
                                        date: Date("2019-07-07"),
                                        reviewScoresRating: 99,
                                        comments: "Could not recommend Donna’s apartment more. The hospitality, location, unit and small touches make this a very special place to stay. Donna is beyond helpful and thoughtful. She made our stay so easy and effortless. The helpful tips around the city and good restaurant and bar recommendations were the perfect finishing touch. Thank you so much Donna!"
                                        }]->(l)
```


$ MATCH (l: Listings { listingId: 1111111 }) MERGE (r: Reviews { id: 12346, date: Date("2019-08-15"), reviewerId: 11112, rev…
Table
Code
Added 1 label, created 1 node, set 6 properties, created 1 relationship, completed after 1 ms.
Added 1 label, created 1 node, set 6 properties, created 1 relationship, completed after 1 ms.


$ MERGE (h: Hosts { hostId: 111111, hostUrl: "https://www.airbnb.com.au/users/show/106130712", hostName: 'Frank and Justine'…
Table
Code
Added 3 labels, created 3 nodes, set 31 properties, created 2 relationships, completed after 9 ms.


$ MATCH (l: Listings { listingId: 2222222 }) MERGE (r: Reviews { id: 23457, date: Date("2019-09-19"), reviewerId: 22223, rev…
Table
Code
Added 1 label, created 1 node, set 6 properties, created 1 relationship, completed after 2 ms.
Added 1 label, created 1 node, set 6 properties, created 1 relationship, completed after 2 ms.


$ MERGE (h: Hosts { hostId: 222222, hostUrl: "https://www.airbnb.com.au/users/show/15874351", hostName: 'Donna', hostVerific…
Table
Code
Added 3 labels, created 3 nodes, set 31 properties, created 2 relationships, completed after 2 ms.

// 2

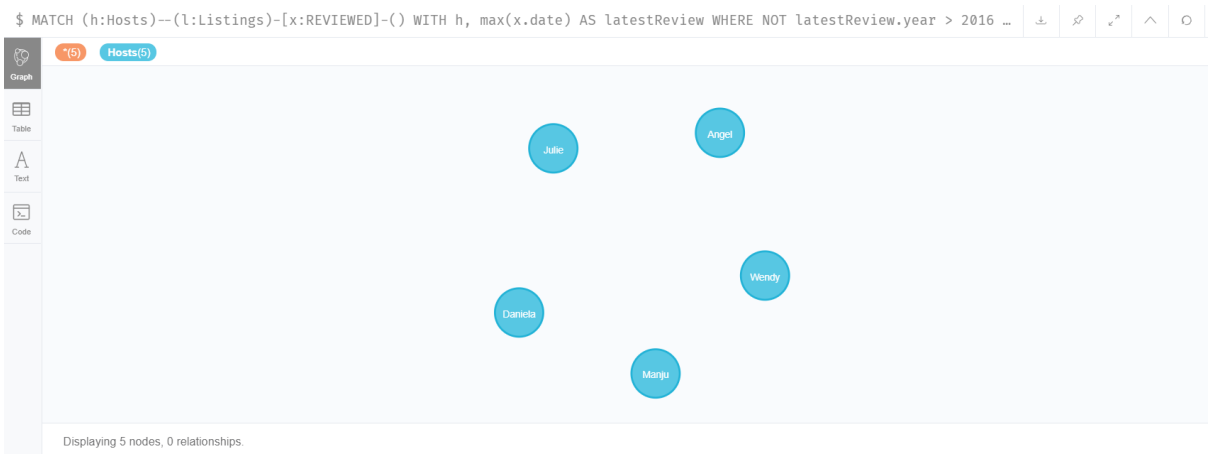
```
MATCH (h:Hosts)
WHERE h.hostSince.year = 2009
SET h.hostVerifications = h.hostVerifications + 'Facebook'
```

// 3

```
MATCH (h:Hosts {hostResponseTime: 'within an hour'})
SET h.hostIsSuperhost = true
```

// 4

```
MATCH (h:Hosts)--(l:Listings)-[x:REVIEWED]-()
WITH h, max(x.date) AS latestReview
WHERE NOT latestReview.year > 2016
SET h.active = false
```



// 5

```
MATCH (l:Listings)
WHERE NOT (l)-[:REVIEWED]-() AND l.availability365 = 0
DETACH DELETE l
```

C.4 Advanced Topic

The accommodation recommendation system has been created on the existing graph database.

The system works on K-Nearest Neighbours (kNN) and Cosine Similarity techniques.

//1 - Create the cosine similarity relationships among all the reviewers based on their review scores ratings on the accommodations.

```
MATCH       (r1:Reviewers)-[x:REVIEWED]->(l:Listings)<-[y:REVIEWED]-(r2:Reviewers)
WITH  SUM(x.reviewScoresRating * y.reviewScoresRating) AS xyDotProduct,
            SQRT(REDUCE(xDot = 0.0, a IN COLLECT(x.reviewScoresRating) | xDot + a^2))
AS xLength,
            SQRT(REDUCE(yDot = 0.0, b IN COLLECT(y.reviewScoresRating) | yDot + b^2))
AS yLength,
            r1, r2
MERGE       (r1)-[s:SIMILARITY]-(r2)
SET     s.similarity = xyDotProduct / (xLength * yLength)
```

// 2 - View the 5 nearest neighbours for a reviewer based on their similarities.

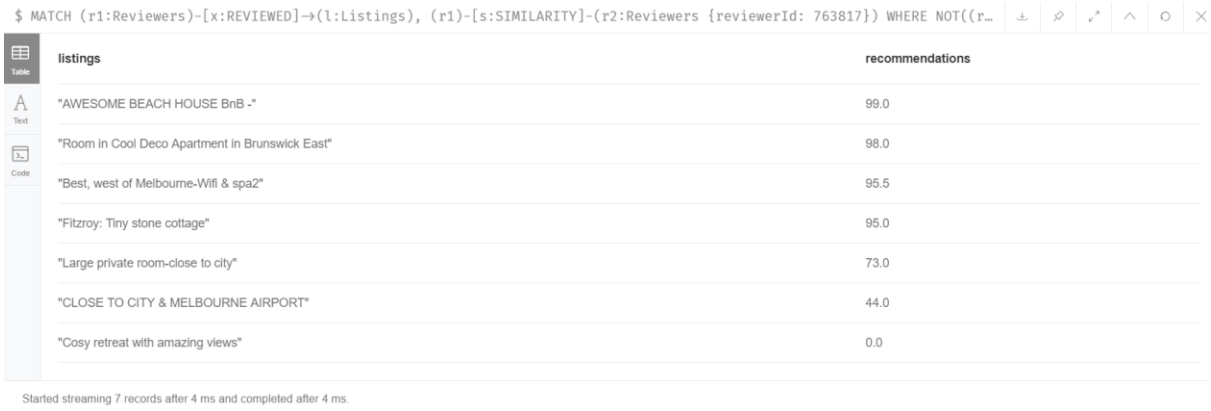
// Here, reviewerId = 763817

```
MATCH    (r1:Reviewers {reviewerId: 763817})-[s:SIMILARITY]-(r2:Reviewers)
WHERE            s.similarity <= 1
WITH     r2, s.similarity AS sim
RETURN   r2.reviewerName AS Neighbour, sim AS Similarity
ORDER BY sim
```

// 3 - Get the accommodation recommendations for any reviewer using the reviewer ID.

//EXAMPLE 1: Here, reviewerId = 763817 and k = 2

```
MATCH    (r1:Reviewers)-[x:REVIEWED]->(l:Listings), (r1)-[s:SIMILARITY]-(r2:Reviewers
{reviewerId: 763817})
WHERE    NOT((r2)-[:REVIEWED]->(l))
WITH     l, s.similarity AS similarity, x.reviewScoresRating AS rating
ORDER BY l.name, similarity
WITH     l.name AS listing, COLLECT(rating)[0..2] AS ratings
WITH     listing, REDUCE(s = 0, i IN ratings | s + i) * 1.0 / SIZE(ratings) AS recommend
ORDER BY recommend DESC
RETURN   listing AS listings, recommend AS recommendations
```



//EXAMPLE 2: Here, reviewerId = 321095 and k = 5

```
MATCH    (r1:Reviewers)-[x:REVIEWED]->(l:Listings), (r1)-[s:SIMILARITY]-(r2:Reviewers
{reviewerId: 321095})
WHERE    NOT((r2)-[:REVIEWED]->(l))
WITH     l, s.similarity AS similarity, x.reviewScoresRating AS rating
ORDER BY l.name, similarity
WITH     l.name AS listing, COLLECT(rating)[0..5] AS ratings
WITH     listing, REDUCE(s = 0, i IN ratings | s + i) * 1.0 / SIZE(ratings) AS recommend
ORDER BY recommend DESC
RETURN   listing AS listings, recommend AS recommendations
```