FIT5137 Individual Assignment Report

Semester 2, 2019

Students Name: Dawei Gu

Students ID: 29910226

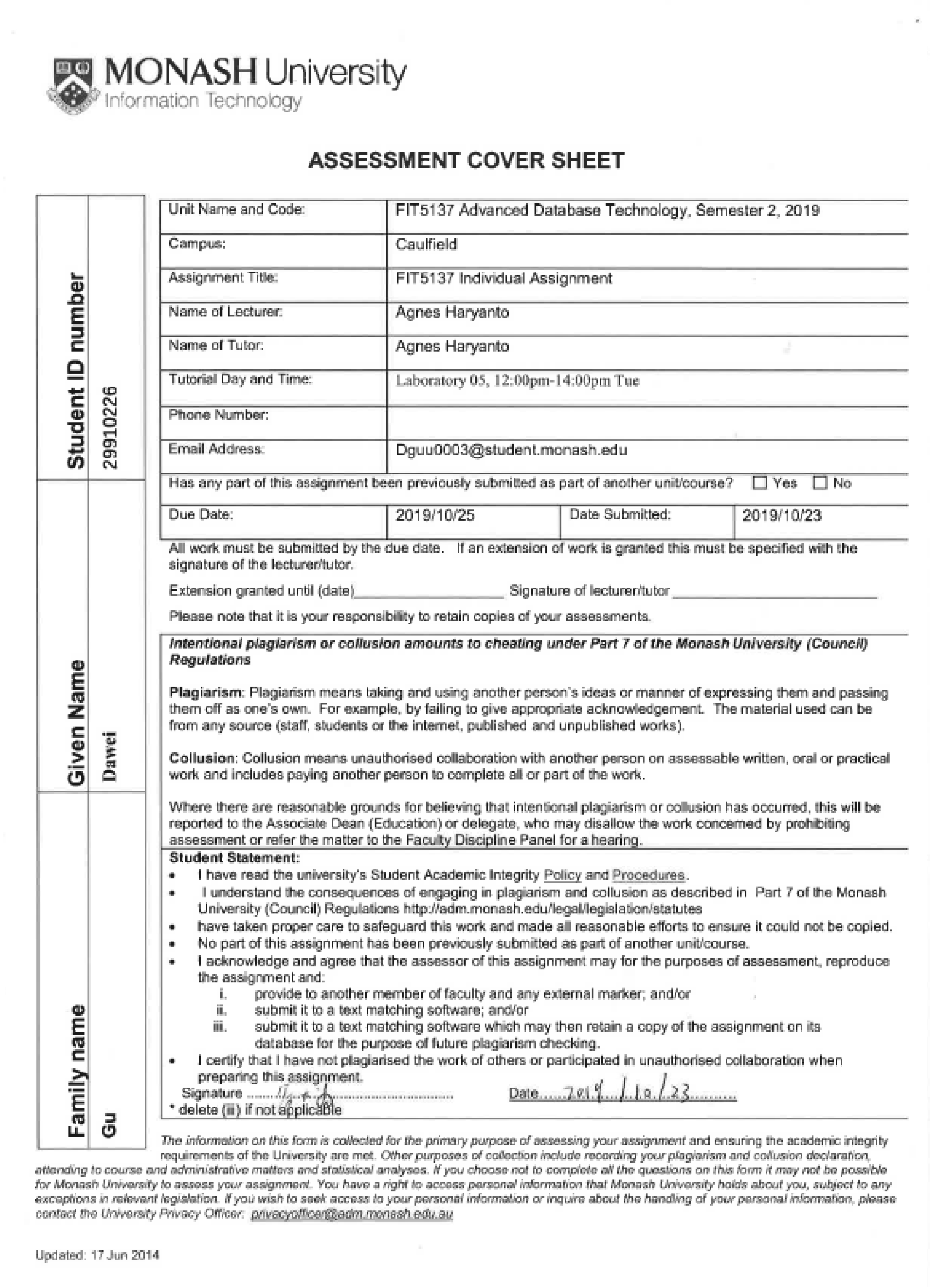
Tutorial Section: FIT5137 Laboratory 05,

12:00pm-14:00pm Tue

Tutor: Agnes Haryanto

Contents

Signed Cover Sheet



MONASH University
Information Technology

ASSESSMENT COVER SHEET

Student ID number	29910226
Given Name	Dawei
Family name	Gu

Unit Name and Code:	FIT5137 Advanced Database Technology, Semester 2, 2019		
Campus:	Caulfield		
Assignment Title:	FIT5137 Individual Assignment		
Name of Lecturer:	Agnes Haryanto		
Name of Tutor:	Agnes Haryanto		
Tutorial Day and Time:	Laboratory 05, 12:00pm-14:00pm Tue		
Phone Number:			
Email Address:	Dguu0003@student.monash.edu		
Has any part of this assignment been previously submitted as part of another unit/course? ☐ Yes ☐ No			
Due Date:	2019/10/25	Date Submitted:	2019/10/23

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 The material used can be from any source (staff, students or the internet, published and unpublished works).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 ......[illegible]...................... Date......2019/10/23.........

* delete (iii) if not applicable

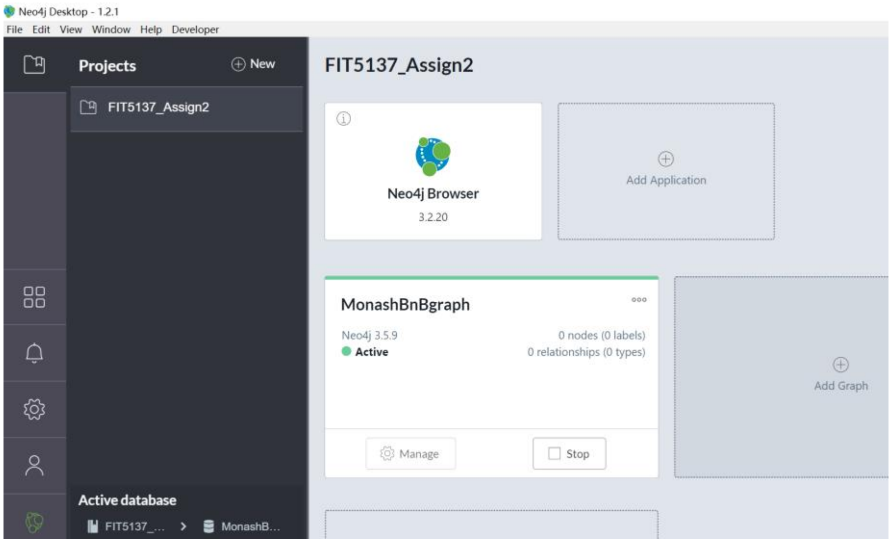
The information on this form is collected for the primary purpose of assessing your assignment and ensuring the academic integrity requirements of the University are met. *Other purposes of collection include recording your plagiarism and collusion declaration, attending to course and administrative 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

Updated: 17 Jun 2014

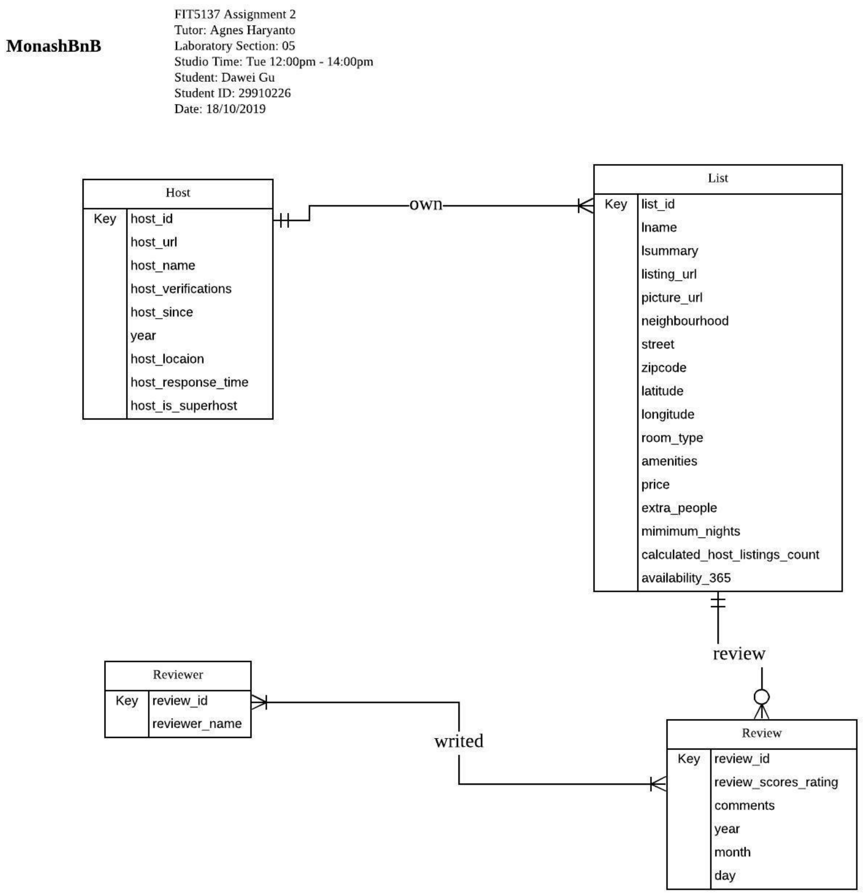
Report

C.1 Database Design

- **Create a new project called *FIT5137_Assign2*.**
- **Create a new graph called *MonashBnBgraph*.**



There will be 4 labels' nodes, they are Host, List, Reviewer, Review. There will be 3 labels' relation amount the 4 label's nodes, the host own list(house), the list(house) review by review, and the reviewer writed review. Following diagram show the design.



Cypher Script

#loading host nodes

```
LOAD CSV WITH HEADERS FROM "file:///host_v2.csv"
AS row
WITH row WHERE row.host_id IS NOT NULL
MERGE (h:Host {host_id: row.host_id})
ON CREATE SET h.host_url = row.host_url,
              h.host_name = row.host_name,
               h.host_verifications = split(split(split(row.host_verifications, '[')[1], ']')[0],
', '),
               h.host_since = row.host_since,
               h.year = toInteger(split(row.host_since,'-')[0]),
               h.host_location = row.host_location,
               h.host_response_time = row.host_response_time,
               h.host_is_superhost = row.host_is_superhost;
```

#load list

```
LOAD CSV WITH HEADERS FROM "file:///listing_v2.csv" AS row
WITH row WHERE row.id IS NOT NULL
MERGE (l:List {list_id: row.id})
ON CREATE SET l.name = row.name,
              l.summary = row.summary,
              l.listing_url = row.listing_url,
              l.picture_url = row.picture_url,
              l.neighbourhood = row.neighbourhood,
              l.street= row.street,
              l.zipcode = row.zipcode,
              l.latitude = toFloat(row.latitude),
              l.longitude = toFloat(row.longitude),
              l.room_type = row.room_type,
              l.amenities = split(split(split(row.amenities, '{')[1], '}')[0], ','),
              l.price = toInteger(row.price),
              l.extra_people = toFloat(replace(row.extra_people,'$', '')),
              l.minimum_nights = toInteger(row.minimum_nights),
              l.calculated_host_listings_count                          =
toInteger(row.calculated_host_listings_count),
              l.availability_365 = toInteger(row.availability_365);
```

#loading reviewer nodes

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

```
AS row
WITH row WHERE row.reviewer_id IS NOT NULL
MERGE (r:Reviewer {reviewer_id: row.reviewer_id})
ON CREATE SET r.reviewer_name = row.reviewer_name;
```

#loading review nodes

```
LOAD CSV WITH HEADERS FROM "file:///review_v2.csv"
AS row
WITH row WHERE row.id IS NOT NULL
MERGE (r:Review {review_id: row.id})
ON CREATE SET r.year = toInteger(split(row.date,'-')[0]),
              r.month = toInteger(split(row.date,'-')[1]),
              r.day = toInteger(split(row.date,'-')[2]),
              r.review_scores_rating = toInteger(row.review_scores_rating),
              r.comments = row.comments;
```

#add relationship between list and host

```
LOAD CSV WITH HEADERS FROM "file:///listing_v2.csv" AS csvLine
MATCH (h:Host {host_id: csvLine.host_id})
MATCH (l:List {list_id: csvLine.id})
CREATE (h)-[:own]->(l);
```

#add relationship between review and reviewer

```
LOAD CSV WITH HEADERS FROM "file:///review_v2.csv" AS csvLine
MATCH (p:Reviewer {reviewer_id: csvLine.reviewer_id})
MATCH (r:Review {review_id: csvLine.id})
CREATE (p)-[:write]->(r);
```

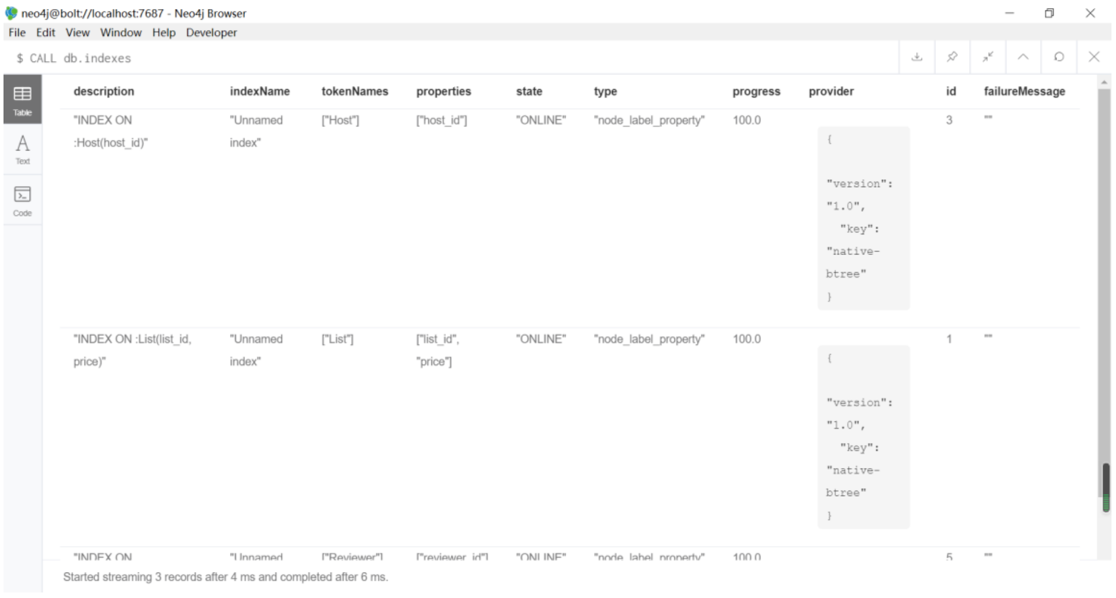
#add relationship between review and list

```
LOAD CSV WITH HEADERS FROM "file:///review_v2.csv" AS csvLine
MATCH (l:List {list_id: csvLine.listing_id})
MATCH (r:Review {review_id: csvLine.id})
CREATE (r)-[:review]->(l);
```

C.2 Queries

#Create index

CREATE INDEX ON :List(list_id, price);
CREATE INDEX ON :Host(host_id);
CREATE INDEX ON :Reviewer(reviewer_id);


neo4j@bolt://localhost:7687 - Neo4j Browser
File Edit View Window Help Developer
Table
Text
Code

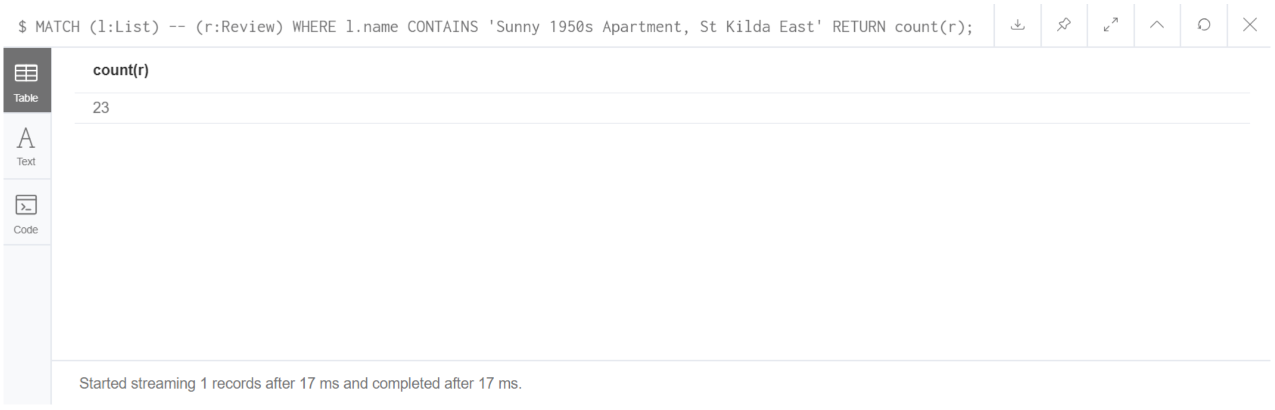
```
$ CALL db.indexes
```

description	indexName	tokenNames	properties	state	type	progress	provider	id	failureMessage
"INDEX ON :Host(host_id)"	"Unnamed index"	["Host"]	["host_id"]	"ONLINE"	"node_label_property"	100.0	{ "version": "1.0", "key": "native-btree" }	3	""
"INDEX ON :List(list_id, price)"	"Unnamed index"	["List"]	["list_id", "price"]	"ONLINE"	"node_label_property"	100.0	{ "version": "1.0", "key": "native-btree" }	1	""
"INDEX ON	"Unnamed	["Reviewer"]	["reviewer_id"]	"ONLINE"	"node_label_property"	100.0		5	""

Started streaming 3 records after 4 ms and completed after 6 ms.

1. How many reviews does "Sunny 1950s Apartment, St Kilda East" have?

MATCH (l:List) -- (r:Review)
WHERE l.name CONTAINS 'Sunny 1950s Apartment, St Kilda East'
RETURN count(r);



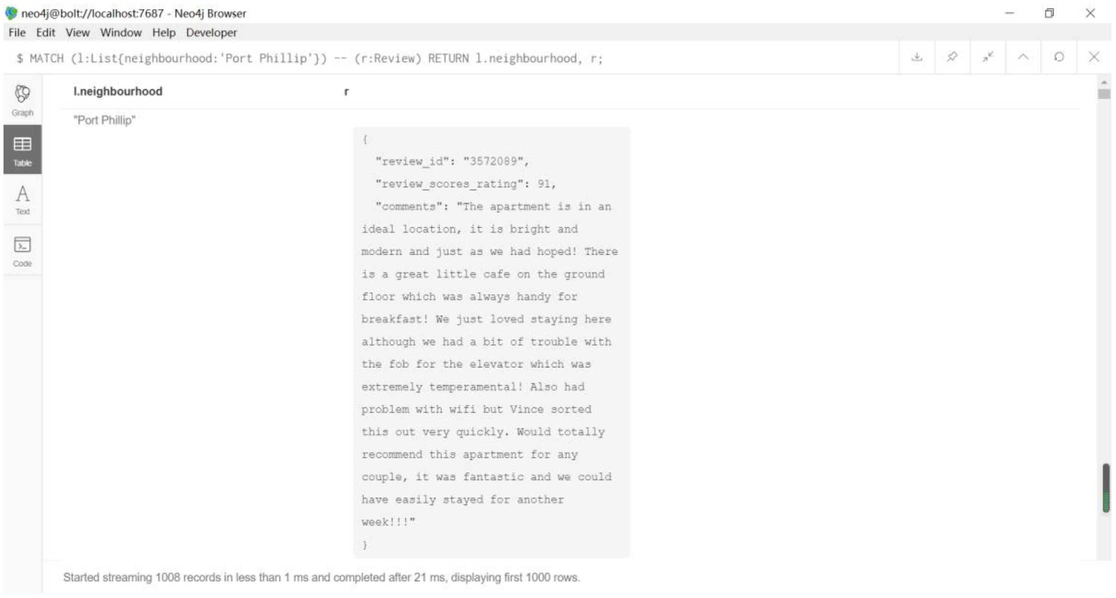
```
$ MATCH (l:List) -- (r:Review) WHERE l.name CONTAINS 'Sunny 1950s Apartment, St Kilda East' RETURN count(r);
```

count(r)
23

Started streaming 1 records after 17 ms and completed after 17 ms.

2. Show all reviews in Port Phillip.

MATCH (l:List{neighbourhood:'Port Phillip'}) -- (r:Review)
RETURN l.neighbourhood, r;



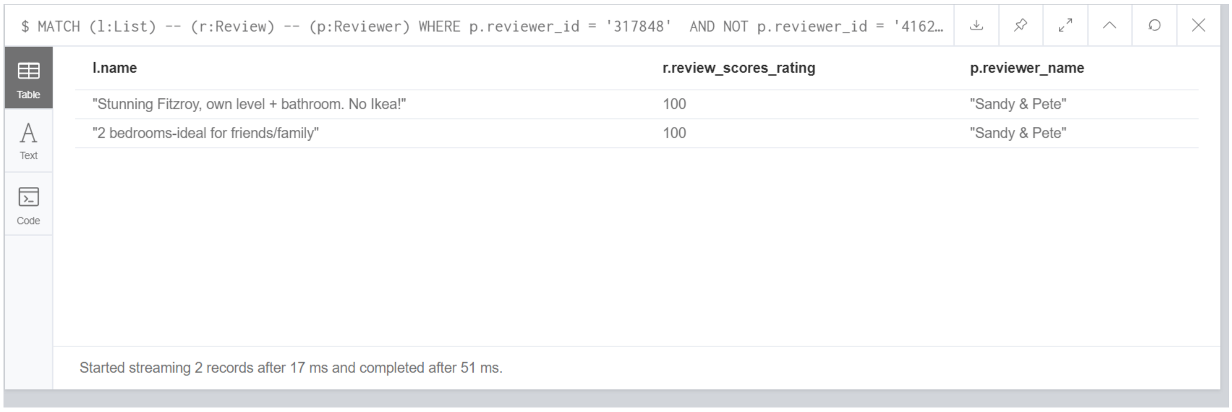
```
$ MATCH (l:List{neighbourhood:'Port Phillip'}) -- (r:Review) RETURN l.neighbourhood, r;
```

l.neighbourhood	r
"Port Phillip"	{ "review_id": "3572089", "review_scores_rating": 91, "comments": "The apartment is in an ideal location, it is bright and modern and just as we had hoped! There is a great little cafe on the ground floor which was always handy for breakfast! We just loved staying here although we had a bit of trouble with the fob for the elevator which was extremely temperamental! Also had problem with wifi but Vince sorted this out very quickly. Would totally recommend this apartment for any couple, it was fantastic and we could have easily stayed for another week!!!" }

Started streaming 1008 records in less than 1 ms and completed after 21 ms, displaying first 1000 rows.

3. Can you recommend accommodations that Jerome (reviewer 4162110) has never been but Sandy & Pete (reviewer 317848) have stayed and gave ratings above 90?

MATCH (l:List) -- (r:Review) -- (p:Reviewer)
WHERE p.reviewer_id = '317848'
    AND NOT p.reviewer_id = '4162110'
    AND r.review_scores_rating > 90
RETURN l.name, r.review_scores_rating, p.reviewer_name;



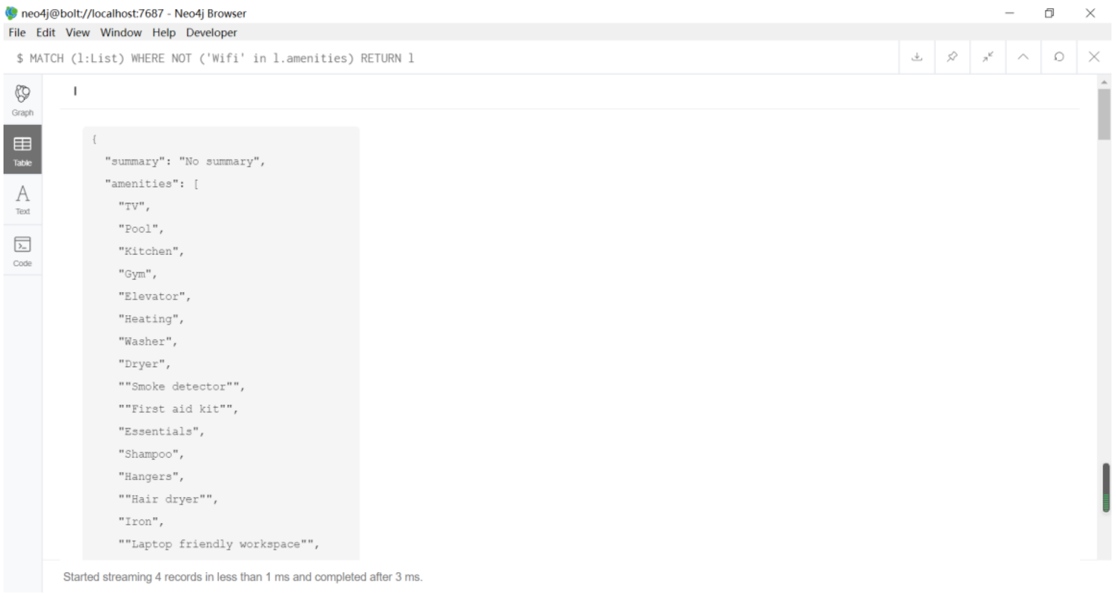
```
$ MATCH (l:List) -- (r:Review) -- (p:Reviewer) WHERE p.reviewer_id = '317848'  AND NOT p.reviewer_id = '4162…
```

l.name	r.review_scores_rating	p.reviewer_name
"Stunning Fitzroy, own level + bathroom. No Ikea!"	100	"Sandy & Pete"
"2 bedrooms-ideal for friends/family"	100	"Sandy & Pete"

Started streaming 2 records after 17 ms and completed after 51 ms.

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

```
MATCH (l:List)
WHERE NOT ('Wifi' in l.amenities)
RETURN l
```



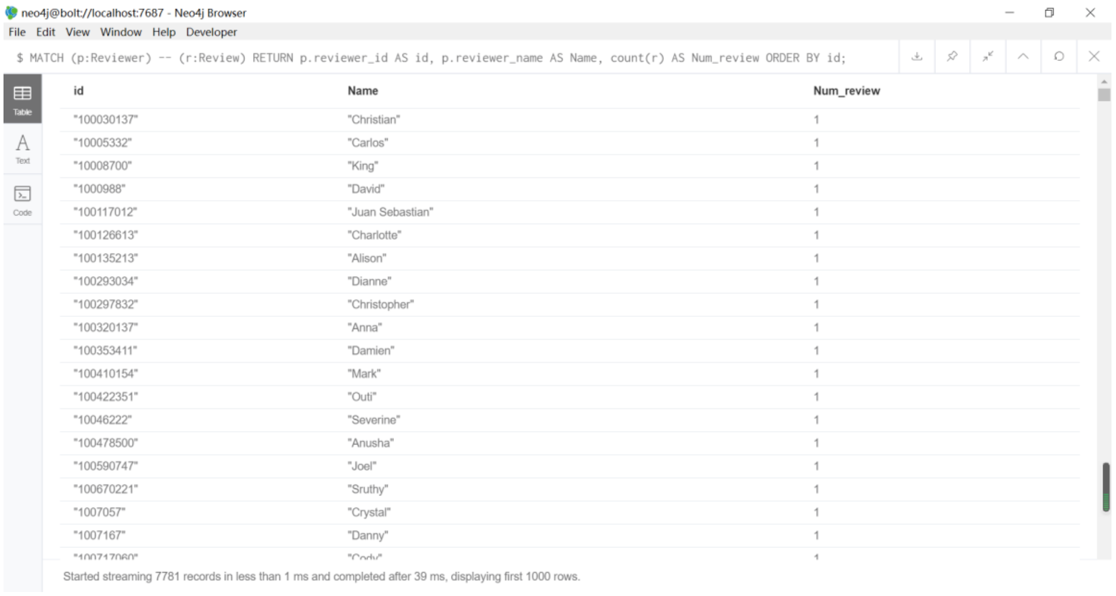
```
$ MATCH (l:List) WHERE NOT ('Wifi' in l.amenities) RETURN l
```

```
l

{
  "summary": "No summary",
  "amenities": [
    "TV",
    "Pool",
    "Kitchen",
    "Gym",
    "Elevator",
    "Heating",
    "Washer",
    "Dryer",
    ""Smoke detector"",
    ""First aid kit"",
    "Essentials",
    "Shampoo",
    "Hangers",
    ""Hair dryer"",
    "Iron",
    ""Laptop friendly workspace"",
```

Started streaming 4 records in less than 1 ms and completed after 3 ms.

5. Count how many times a reviewer left reviews.

```
MATCH (p:Reviewer) -- (r:Review)
RETURN p.reviewer_id AS id, p.reviewer_name AS Name, count(r) AS Num_review;
```



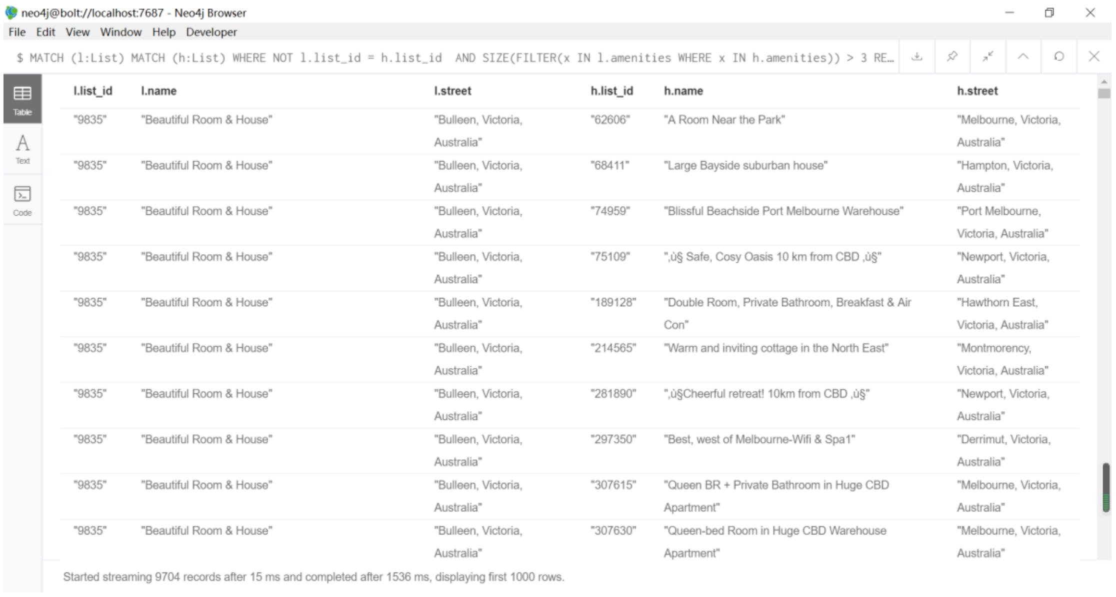
```
$ MATCH (p:Reviewer) -- (r:Review) RETURN p.reviewer_id AS id, p.reviewer_name AS Name, count(r) AS Num_review ORDER BY id;
```

id	Name	Num_review
"100030137"	"Christian"	1
"10005332"	"Carlos"	1
"10008700"	"King"	1
"1000988"	"David"	1
"100117012"	"Juan Sebastian"	1
"100126613"	"Charlotte"	1
"100135213"	"Alison"	1
"100293034"	"Dianne"	1
"100297832"	"Christopher"	1
"100320137"	"Anna"	1
"100353411"	"Damien"	1
"100410154"	"Mark"	1
"100422351"	"Outi"	1
"10046222"	"Severine"	1
"100478500"	"Anusha"	1
"100590747"	"Joel"	1
"100670221"	"Sruthy"	1
"1007057"	"Crystal"	1
"1007167"	"Danny"	1
"100717060"	"Cody"	1

Started streaming 7781 records in less than 1 ms and completed after 39 ms, displaying first 1000 rows.

6. Display a list of pairs of accommodations having more than three amenities in common.

```
MATCH (l:List)
MATCH (h:List)
WHERE NOT l.list_id = h.list_id
    AND SIZE(FILTER(x IN l.amenities WHERE x IN h.amenities)) > 3
RETURN l.list_id, l.name, l.street, h.list_id, h.name, h.street;
```


neo4j@bolt://localhost:7687 - Neo4j Browser
File Edit View Window Help Developer
Table
Text
Code

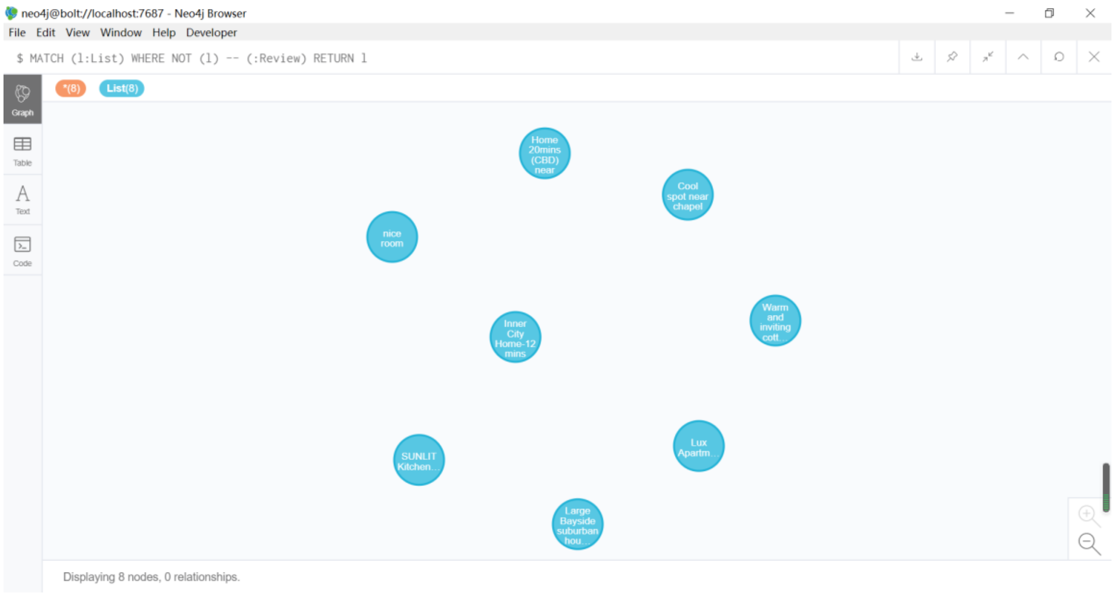
```
$ MATCH (l:List) MATCH (h:List) WHERE NOT l.list_id = h.list_id  AND SIZE(FILTER(x IN l.amenities WHERE x IN h.amenities)) > 3 RE…
```

l.list_id	l.name	l.street	h.list_id	h.name	h.street
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"62606"	"A Room Near the Park"	"Melbourne, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"68411"	"Large Bayside suburban house"	"Hampton, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"74959"	"Blissful Beachside Port Melbourne Warehouse"	"Port Melbourne, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"75109"	",ù§ Safe, Cosy Oasis 10 km from CBD ,ù§"	"Newport, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"189128"	"Double Room, Private Bathroom, Breakfast & Air Con"	"Hawthorn East, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"214565"	"Warm and inviting cottage in the North East"	"Montmorency, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"281890"	",ù§Cheerful retreat! 10km from CBD ,ù§"	"Newport, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"297350"	"Best, west of Melbourne-Wifi & Spa1"	"Derrimut, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"307615"	"Queen BR + Private Bathroom in Huge CBD Apartment"	"Melbourne, Victoria, Australia"
"9835"	"Beautiful Room & House"	"Bulleen, Victoria, Australia"	"307630"	"Queen-bed Room in Huge CBD Warehouse Apartment"	"Melbourne, Victoria, Australia"

Started streaming 9704 records after 15 ms and completed after 1536 ms, displaying first 1000 rows.

7. Which listings do not have any review?

```
MATCH (l:List)
WHERE NOT (l) -- (:Review)
RETURN l
```

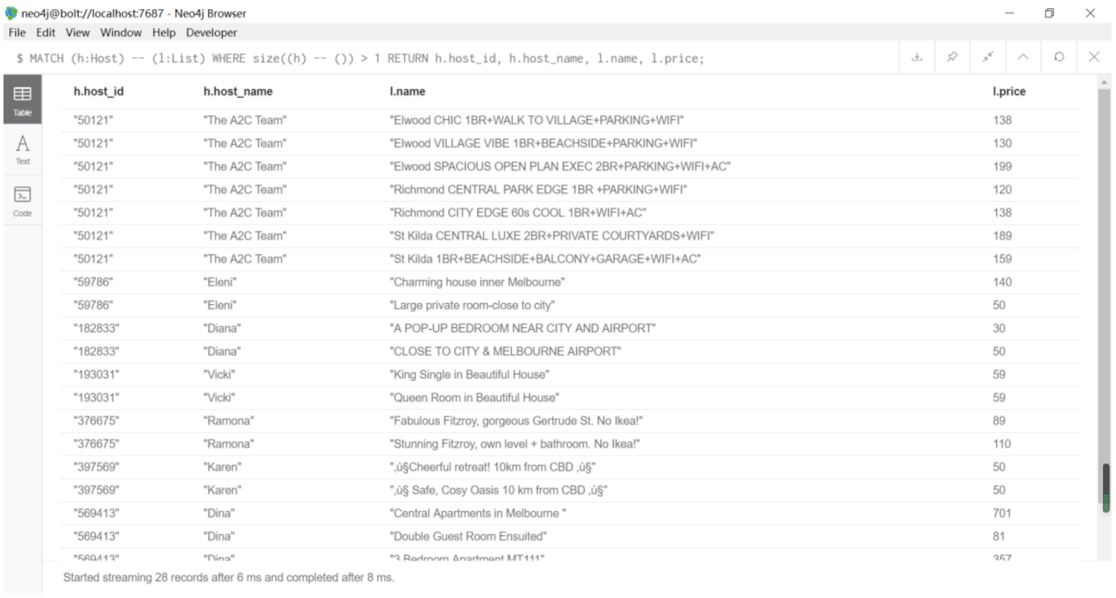


```
$ MATCH (l:List) WHERE NOT (l) -- (:Review) RETURN l
```

Displaying 8 nodes, 0 relationships.

8. Show all hosts who have multiple listings. Display both the host details and the listing name and price.

```
MATCH (h:Host) -- (l:List)
WHERE size((h) -- ()) > 1
RETURN h.host_id, h.host_name, l.name, l.price;
```


neo4j@bolt://localhost:7687 - Neo4j Browser
File Edit View Window Help Developer
Table
Text
Code

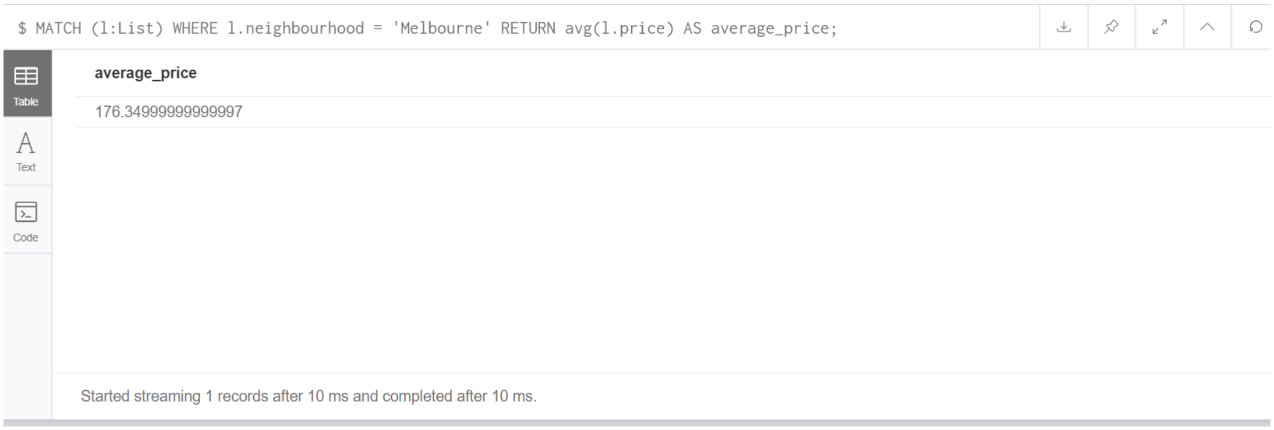
```
$ MATCH (h:Host) -- (l:List) WHERE size((h) -- ()) > 1 RETURN h.host_id, h.host_name, l.name, l.price;
```

h.host_id	h.host_name	l.name	l.price
"50121"	"The A2C Team"	"Elwood CHIC 1BR+WALK TO VILLAGE+PARKING+WIFI"	138
"50121"	"The A2C Team"	"Elwood VILLAGE VIBE 1BR+BEACHSIDE+PARKING+WIFI"	130
"50121"	"The A2C Team"	"Elwood SPACIOUS OPEN PLAN EXEC 2BR+PARKING+WIFI+AC"	199
"50121"	"The A2C Team"	"Richmond CENTRAL PARK EDGE 1BR +PARKING+WIFI"	120
"50121"	"The A2C Team"	"Richmond CITY EDGE 60s COOL 1BR+WIFI+AC"	138
"50121"	"The A2C Team"	"St Kilda CENTRAL LUXE 2BR+PRIVATE COURTYARDS+WIFI"	189
"50121"	"The A2C Team"	"St Kilda 1BR+BEACHSIDE+BALCONY+GARAGE+WIFI+AC"	159
"59786"	"Eleni"	"Charming house inner Melbourne"	140
"59786"	"Eleni"	"Large private room-close to city"	50
"182833"	"Diana"	"A POP-UP BEDROOM NEAR CITY AND AIRPORT"	30
"182833"	"Diana"	"CLOSE TO CITY & MELBOURNE AIRPORT"	50
"193031"	"Vicki"	"King Single in Beautiful House"	59
"193031"	"Vicki"	"Queen Room in Beautiful House"	59
"376675"	"Ramona"	"Fabulous Fitzroy, gorgeous Gertrude St. No Ikea!"	89
"376675"	"Ramona"	"Stunning Fitzroy, own level + bathroom. No Ikea!"	110
"397569"	"Karen"	",ù§Cheerful retreat! 10km from CBD ,ù§"	50
"397569"	"Karen"	",ù§ Safe, Cosy Oasis 10 km from CBD ,ù§"	50
"569413"	"Dina"	"Central Apartments in Melbourne "	701
"569413"	"Dina"	"Double Guest Room Ensuited"	81
"569413"	"Dina"	"3 Bedroom Apartment MT111"	357

```
Started streaming 28 records after 6 ms and completed after 8 ms.
```

9. What is the average price for accommodations in Melbourne neighbourhood?

```
MATCH (l:List)
WHERE l.neighbourhood = 'Melbourne'
RETURN avg(l.price) AS average_price;
```



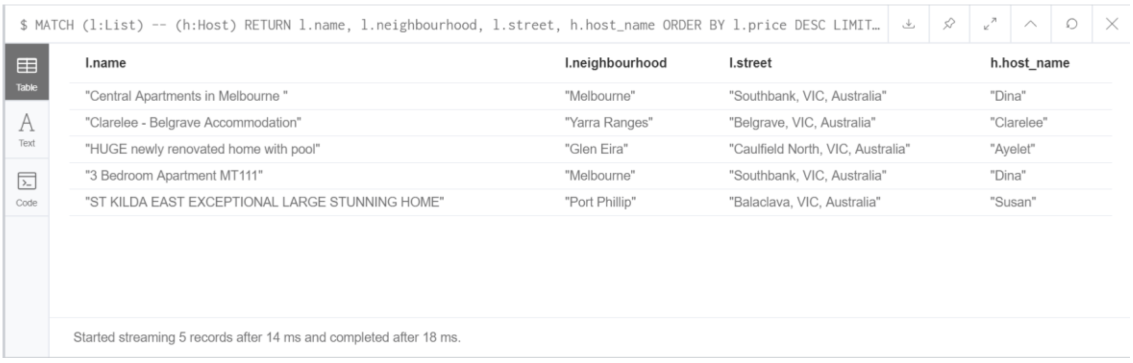
```
$ MATCH (l:List) WHERE l.neighbourhood = 'Melbourne' RETURN avg(l.price) AS average_price;
```

average_price
176.34999999999997

```
Started streaming 1 records after 10 ms and completed after 10 ms.
```

10. Where are the top 5 most expensive accommodations? Display the locations, host information, and names of those accommodation.

MATCH (l:List) -- (h:Host)
RETURN l.name, l.neighbourhood, l.street, h.host_name
ORDER BY l.price DESC
LIMIT 5;



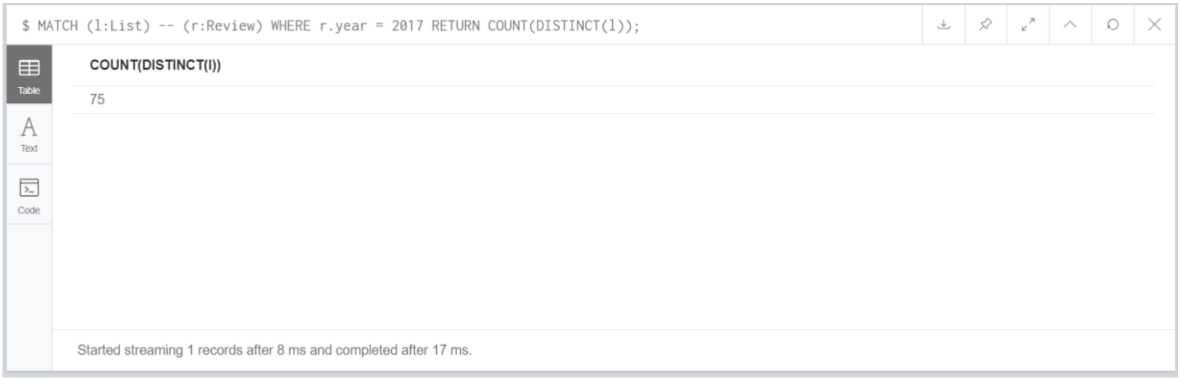
```
$ MATCH (l:List) -- (h:Host) RETURN l.name, l.neighbourhood, l.street, h.host_name ORDER BY l.price DESC LIMIT…
```

l.name	l.neighbourhood	l.street	h.host_name
"Central Apartments in Melbourne "	"Melbourne"	"Southbank, VIC, Australia"	"Dina"
"Clarelee - Belgrave Accommodation"	"Yarra Ranges"	"Belgrave, VIC, Australia"	"Clarelee"
"HUGE newly renovated home with pool"	"Glen Eira"	"Caulfield North, VIC, Australia"	"Ayelet"
"3 Bedroom Apartment MT111"	"Melbourne"	"Southbank, VIC, Australia"	"Dina"
"ST KILDA EAST EXCEPTIONAL LARGE STUNNING HOME"	"Port Phillip"	"Balaclava, VIC, Australia"	"Susan"

Started streaming 5 records after 14 ms and completed after 18 ms.

11. How many accommodations were reviewed in 2017?

MATCH (l:List) -- (r:Review)
WHERE r.year = 2017
RETURN COUNT(DISTINCT(l));



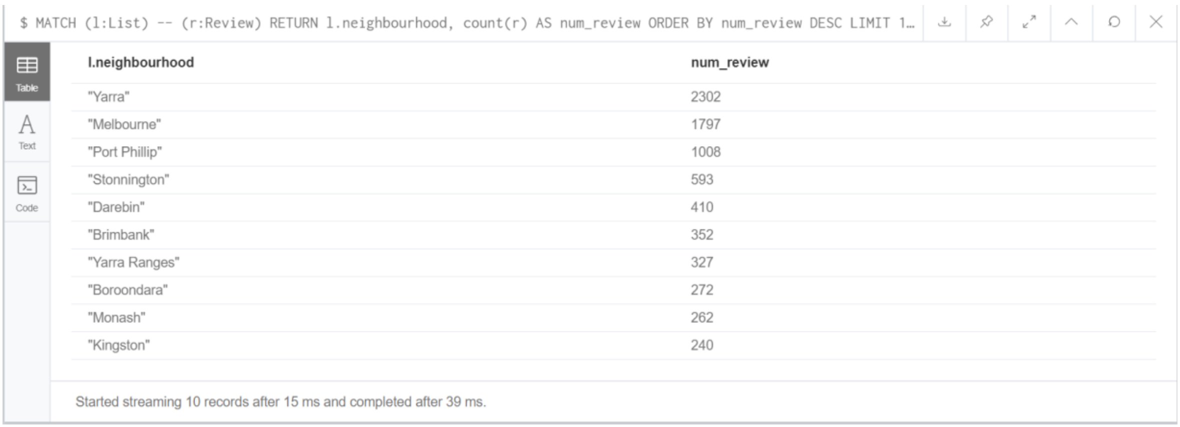
```
$ MATCH (l:List) -- (r:Review) WHERE r.year = 2017 RETURN COUNT(DISTINCT(l));
```

COUNT(DISTINCT(l))
75

Started streaming 1 records after 8 ms and completed after 17 ms.

12. What are the top 10 most popular neighbourhoods based on the total average reviews?

MATCH (l:List) -- (r:Review)
RETURN l.neighbourhood, count(r) AS num_review
ORDER BY num_review DESC
LIMIT 10;



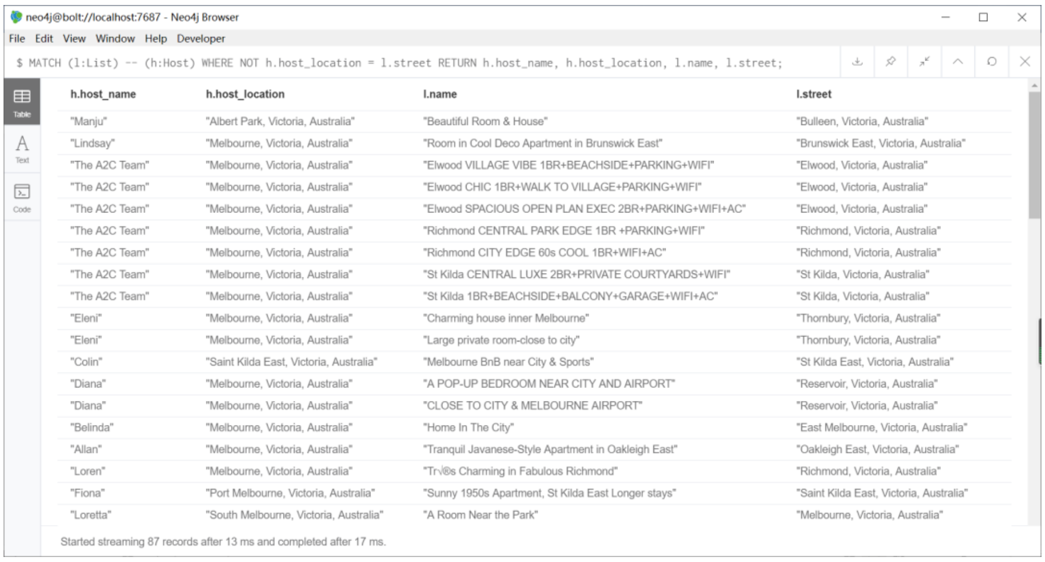
```
$ MATCH (l:List) -- (r:Review) RETURN l.neighbourhood, count(r) AS num_review ORDER BY num_review DESC LIMIT 1…
```

l.neighbourhood	num_review
"Yarra"	2302
"Melbourne"	1797
"Port Phillip"	1008
"Stonnington"	593
"Darebin"	410
"Brimbank"	352
"Yarra Ranges"	327
"Boroondara"	272
"Monash"	262
"Kingston"	240

Started streaming 10 records after 15 ms and completed after 39 ms.

13. Find hosts whose location are different from their listings. Show the host name, host location, listing name, and listing location.

MATCH (l:List) -- (h:Host)
WHERE NOT h.host_location = l.street
RETURN h.host_name, h.host_location, l.name, l.street;



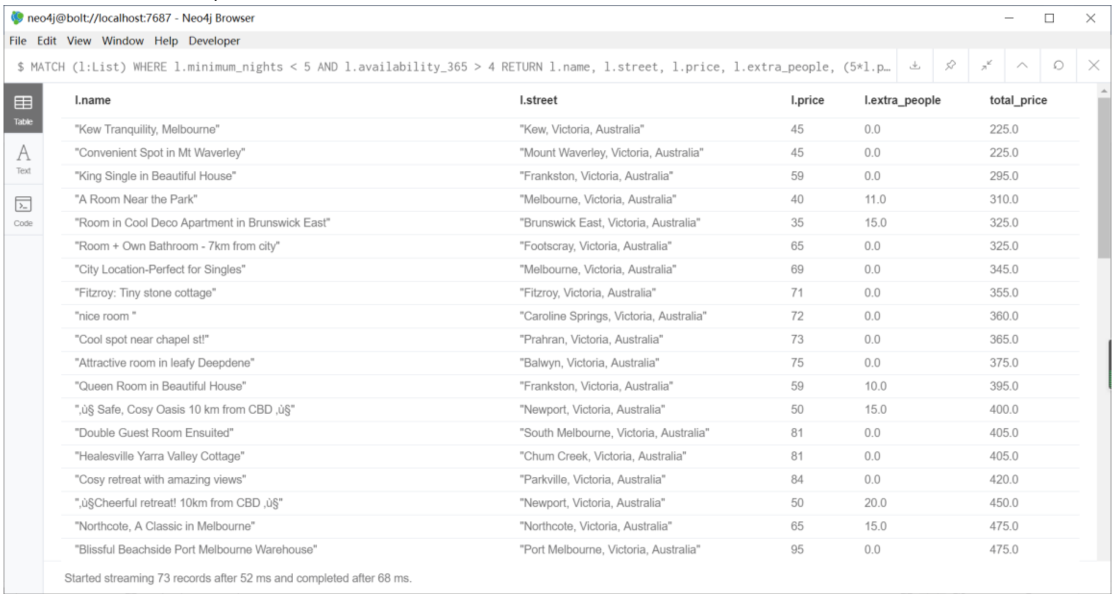
```
$ MATCH (l:List) -- (h:Host) WHERE NOT h.host_location = l.street RETURN h.host_name, h.host_location, l.name, l.street;
```

h.host_name	h.host_location	l.name	l.street
"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"	"Elwood VILLAGE VIBE 1BR+BEACHSIDE+PARKING+WIFI"	"Elwood, Victoria, Australia"
"The A2C Team"	"Melbourne, Victoria, Australia"	"Elwood CHIC 1BR+WALK TO VILLAGE+PARKING+WIFI"	"Elwood, Victoria, Australia"
"The A2C Team"	"Melbourne, Victoria, Australia"	"Elwood SPACIOUS OPEN PLAN EXEC 2BR+PARKING+WIFI+AC"	"Elwood, Victoria, Australia"
"The A2C Team"	"Melbourne, Victoria, Australia"	"Richmond CENTRAL PARK EDGE 1BR +PARKING+WIFI"	"Richmond, Victoria, Australia"
"The A2C Team"	"Melbourne, Victoria, Australia"	"Richmond CITY EDGE 60s COOL 1BR+WIFI+AC"	"Richmond, Victoria, Australia"
"The A2C Team"	"Melbourne, Victoria, Australia"	"St Kilda CENTRAL LUXE 2BR+PRIVATE COURTYARDS+WIFI"	"St Kilda, Victoria, Australia"
"The A2C Team"	"Melbourne, Victoria, Australia"	"St Kilda 1BR+BEACHSIDE+BALCONY+GARAGE+WIFI+AC"	"St Kilda, Victoria, Australia"
"Eleni"	"Melbourne, Victoria, Australia"	"Charming house inner Melbourne"	"Thornbury, Victoria, Australia"
"Eleni"	"Melbourne, Victoria, Australia"	"Large private room-close to city"	"Thornbury, Victoria, Australia"
"Colin"	"Saint Kilda East, Victoria, Australia"	"Melbourne BnB near City & Sports"	"St Kilda East, Victoria, Australia"
"Diana"	"Melbourne, Victoria, Australia"	"A POP-UP BEDROOM NEAR CITY AND AIRPORT"	"Reservoir, Victoria, Australia"
"Diana"	"Melbourne, Victoria, Australia"	"CLOSE TO CITY & MELBOURNE AIRPORT"	"Reservoir, Victoria, Australia"
"Belinda"	"Melbourne, Victoria, Australia"	"Home In The City"	"East Melbourne, Victoria, Australia"
"Allan"	"Melbourne, Victoria, Australia"	"Tranquil Javanese-Style Apartment in Oakleigh East"	"Oakleigh East, Victoria, Australia"
"Loren"	"Melbourne, Victoria, Australia"	"Tr√®s Charming in Fabulous Richmond"	"Richmond, Victoria, Australia"
"Fiona"	"Port Melbourne, Victoria, Australia"	"Sunny 1950s Apartment, St Kilda East Longer stays"	"Saint Kilda East, Victoria, Australia"
"Loretta"	"South Melbourne, Victoria, Australia"	"A Room Near the Park"	"Melbourne, Victoria, Australia"

Started streaming 87 records after 13 ms and completed after 17 ms.

14. Assuming that each accommodation only accepts two guests, calculate the price of each accommodation for four people staying for five nights. Display only the accommodation name, location, price per night, extra people charge, and total price. Rank the accommodation from the cheapest price.

MATCH (l:List)
WHERE l.minimum_nights < 6
AND l.availability_365 > 4
RETURN l.name, l.street, l.price, l.extra_people, (5*l.price + 5*2*l.extra_people) AS total_price
ORDER BY total_price;



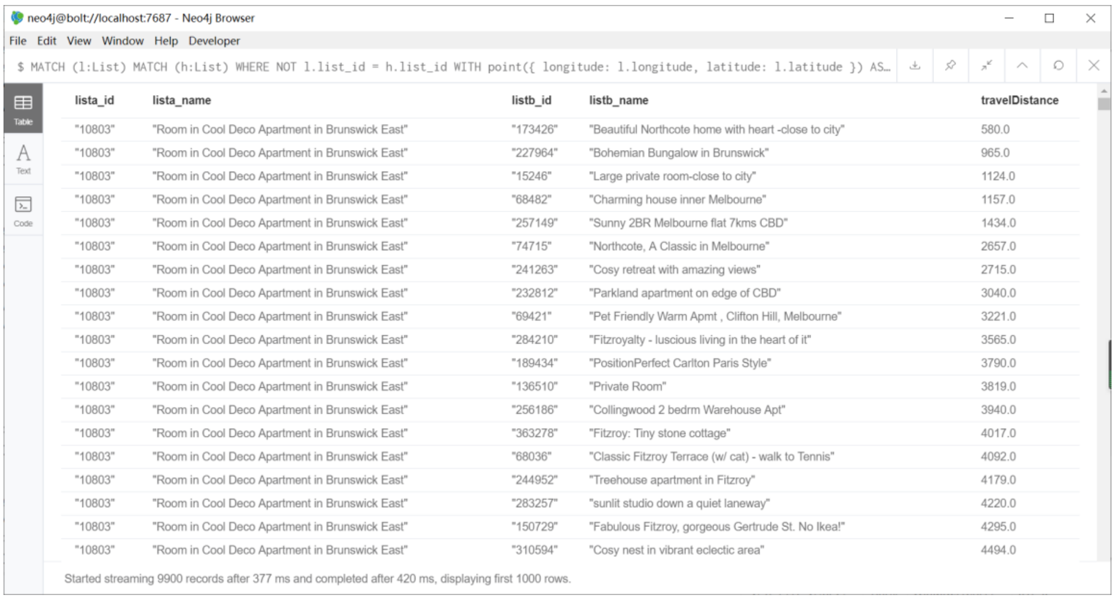
```
$ MATCH (l:List) WHERE l.minimum_nights < 5 AND l.availability_365 > 4 RETURN l.name, l.street, l.price, l.extra_people, (5*l.p…
```

l.name	l.street	l.price	l.extra_people	total_price
"Kew Tranquility, Melbourne"	"Kew, Victoria, Australia"	45	0.0	225.0
"Convenient Spot in Mt Waverley"	"Mount Waverley, Victoria, Australia"	45	0.0	225.0
"King Single in Beautiful House"	"Frankston, Victoria, Australia"	59	0.0	295.0
"A Room Near the Park"	"Melbourne, Victoria, Australia"	40	11.0	310.0
"Room in Cool Deco Apartment in Brunswick East"	"Brunswick East, Victoria, Australia"	35	15.0	325.0
"Room + Own Bathroom - 7km from city"	"Footscray, Victoria, Australia"	65	0.0	325.0
"City Location-Perfect for Singles"	"Melbourne, Victoria, Australia"	69	0.0	345.0
"Fitzroy: Tiny stone cottage"	"Fitzroy, Victoria, Australia"	71	0.0	355.0
"nice room "	"Caroline Springs, Victoria, Australia"	72	0.0	360.0
"Cool spot near chapel st!"	"Prahran, Victoria, Australia"	73	0.0	365.0
"Attractive room in leafy Deepdene"	"Balwyn, Victoria, Australia"	75	0.0	375.0
"Queen Room in Beautiful House"	"Frankston, Victoria, Australia"	59	10.0	395.0
",ù§ Safe, Cosy Oasis 10 km from CBD ,ù§"	"Newport, Victoria, Australia"	50	15.0	400.0
"Double Guest Room Ensuited"	"South Melbourne, Victoria, Australia"	81	0.0	405.0
"Healesville Yarra Valley Cottage"	"Chum Creek, Victoria, Australia"	81	0.0	405.0
"Cosy retreat with amazing views"	"Parkville, Victoria, Australia"	84	0.0	420.0
",ù§Cheerful retreat! 10km from CBD ,ù§"	"Newport, Victoria, Australia"	50	20.0	450.0
"Northcote, A Classic in Melbourne"	"Northcote, Victoria, Australia"	65	15.0	475.0
"Blissful Beachside Port Melbourne Warehouse"	"Port Melbourne, Victoria, Australia"	95	0.0	475.0

Started streaming 73 records after 52 ms and completed after 68 ms.

15. For each listing, rank other listings that are close to each other by their locations. You will need to use the longitude and latitude to calculate the distance between listings.

```
MATCH (l:List)
MATCH (h:List)
WHERE NOT l.list_id = h.list_id
WITH point({ longitude: l.longitude, latitude: l.latitude }) AS pointa,
        l.list_id AS lista_id,
        l.name AS lista_name,
        point({ longitude: h.longitude, latitude: h.latitude }) AS pointb,
        h.list_id AS listb_id,
        h.name AS listb_name
RETURN lista_id, lista_name, listb_id, listb_name, round(distance(pointa, pointb)) AS
travelDistance
ORDER BY lista_id, travelDistance;
```


neo4j@bolt://localhost:7687 - Neo4j Browser
File Edit View Window Help Developer
Table
Text
Code

```
$ MATCH (l:List) MATCH (h:List) WHERE NOT l.list_id = h.list_id WITH point({ longitude: l.longitude, latitude: l.latitude }) AS…
```

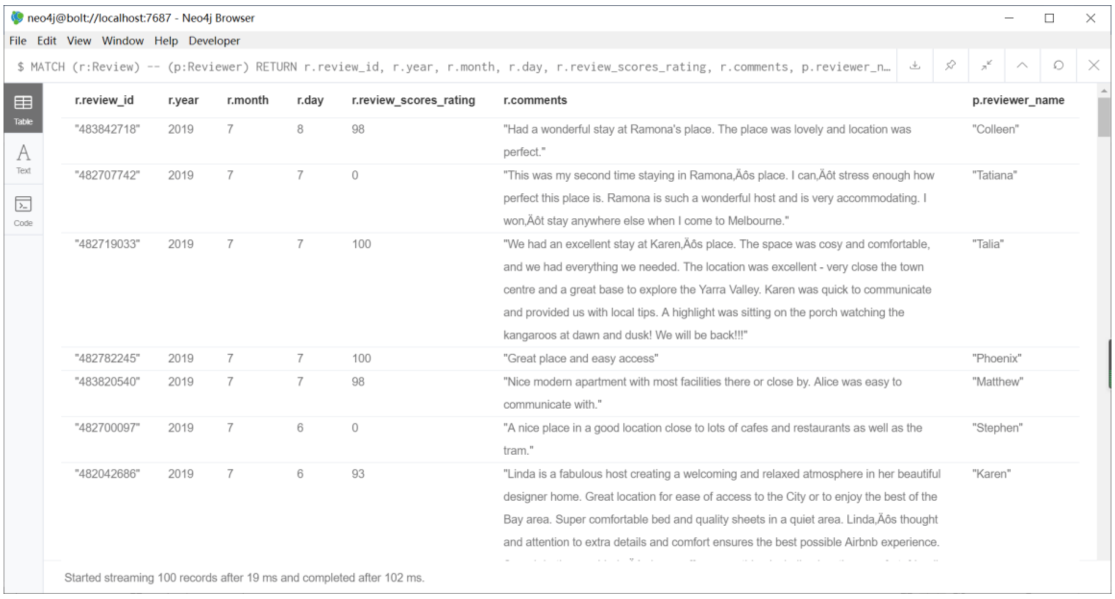
lista_id	lista_name	listb_id	listb_name	travelDistance
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"173426"	"Beautiful Northcote home with heart -close to city"	580.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"227964"	"Bohemian Bungalow in Brunswick"	965.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"15246"	"Large private room-close to city"	1124.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"68482"	"Charming house inner Melbourne"	1157.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"257149"	"Sunny 2BR Melbourne flat 7kms CBD"	1434.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"74715"	"Northcote, A Classic in Melbourne"	2657.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"241263"	"Cosy retreat with amazing views"	2715.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"232812"	"Parkland apartment on edge of CBD"	3040.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"69421"	"Pet Friendly Warm Apmt , Clifton Hill, Melbourne"	3221.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"284210"	"Fitzroyalty - luscious living in the heart of it"	3565.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"189434"	"PositionPerfect Carlton Paris Style"	3790.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"136510"	"Private Room"	3819.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"256186"	"Collingwood 2 bedrm Warehouse Apt"	3940.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"363278"	"Fitzroy: Tiny stone cottage"	4017.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"68036"	"Classic Fitzroy Terrace (w/ cat) - walk to Tennis"	4092.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"244952"	"Treehouse apartment in Fitzroy"	4179.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"283257"	"sunlit studio down a quiet laneway"	4220.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"150729"	"Fabulous Fitzroy, gorgeous Gertrude St. No Ikea!"	4295.0
"10803"	"Room in Cool Deco Apartment in Brunswick East"	"310594"	"Cosy nest in vibrant eclectic area"	4494.0

```
Started streaming 9900 records after 377 ms and completed after 420 ms, displaying first 1000 rows.
```

Additional 5 Queries:

Addional Q1. What is most recent 100 reviews, and how is reviewer?

MATCH (r:Review) – (p:Reviewer)
RETURN r.review_id, r.year, r.month, r.day, r.review_scores_rating, r.comments, p.reviewer_name
ORDER BY r.year DESC, r.month DESC, r.day DESC
LIMIT 100;



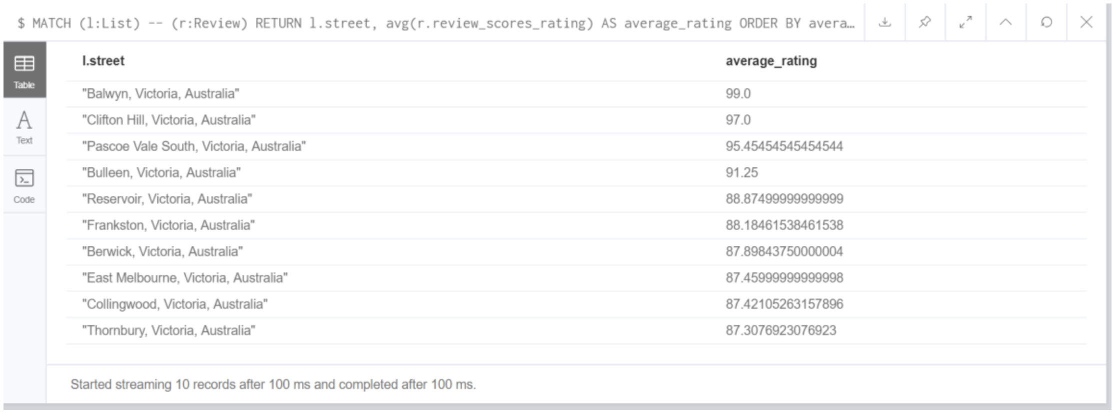
```
$ MATCH (r:Review) -- (p:Reviewer) RETURN r.review_id, r.year, r.month, r.day, r.review_scores_rating, r.comments, p.reviewer_n…
```

r.review_id	r.year	r.month	r.day	r.review_scores_rating	r.comments	p.reviewer_name
"483842718"	2019	7	8	98	"Had a wonderful stay at Ramona's place. The place was lovely and location was perfect."	"Colleen"
"482707742"	2019	7	7	0	"This was my second time staying in Ramona‚Äôs place. I can‚Äôt stress enough how perfect this place is. Ramona is such a wonderful host and is very accommodating. I won‚Äôt stay anywhere else when I come to Melbourne."	"Tatiana"
"482719033"	2019	7	7	100	"We had an excellent stay at Karen‚Äôs place. The space was cosy and comfortable, and we had everything we needed. The location was excellent - very close the town centre and a great base to explore the Yarra Valley. Karen was quick to communicate and provided us with local tips. A highlight was sitting on the porch watching the kangaroos at dawn and dusk! We will be back!!!"	"Talia"
"482782245"	2019	7	7	100	"Great place and easy access"	"Phoenix"
"483820540"	2019	7	7	98	"Nice modern apartment with most facilities there or close by. Alice was easy to communicate with."	"Matthew"
"482700097"	2019	7	6	0	"A nice place in a good location close to lots of cafes and restaurants as well as the tram."	"Stephen"
"482042686"	2019	7	6	93	"Linda is a fabulous host creating a welcoming and relaxed atmosphere in her beautiful designer home. Great location for ease of access to the City or to enjoy the best of the Bay area. Super comfortable bed and quality sheets in a quiet area. Linda‚Äôs thought and attention to extra details and comfort ensures the best possible Airbnb experience.	"Karen"

Started streaming 100 records after 19 ms and completed after 102 ms.

Addional Q2. What is the top 10 cities have the highest review rating?

MATCH (l:List) -- (r:Review)
RETURN l.street, avg(r.review_scores_rating) AS average_rating
ORDER BY average_rating DESC
LIMIT 10;



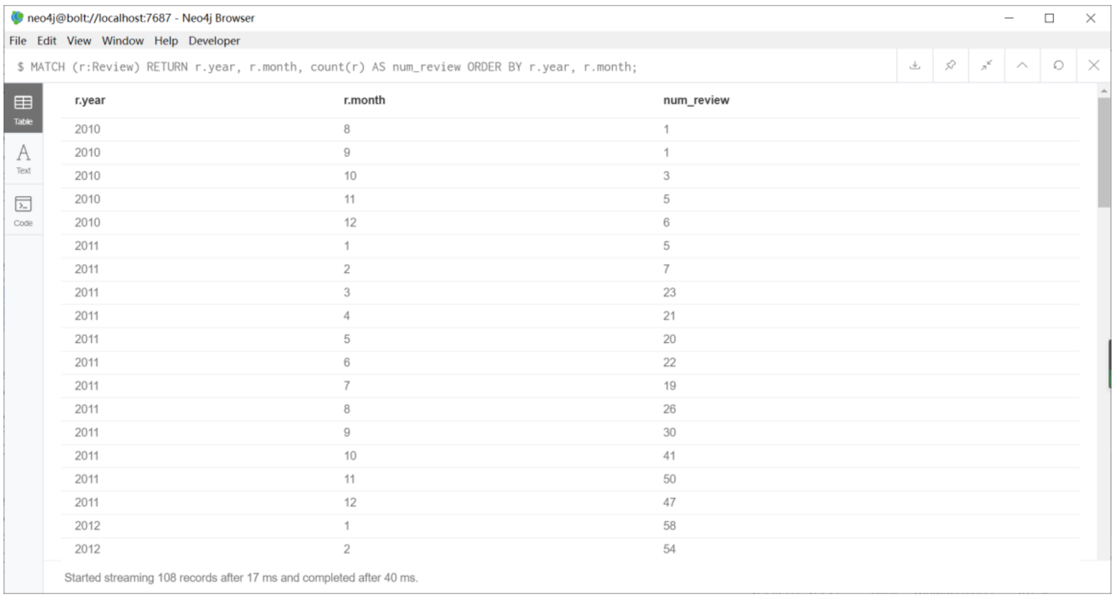
```
$ MATCH (l:List) -- (r:Review) RETURN l.street, avg(r.review_scores_rating) AS average_rating ORDER BY avera…
```

l.street	average_rating
"Balwyn, Victoria, Australia"	99.0
"Clifton Hill, Victoria, Australia"	97.0
"Pascoe Vale South, Victoria, Australia"	95.45454545454544
"Bulleen, Victoria, Australia"	91.25
"Reservoir, Victoria, Australia"	88.87499999999999
"Frankston, Victoria, Australia"	88.18461538461538
"Berwick, Victoria, Australia"	87.89843750000004
"East Melbourne, Victoria, Australia"	87.45999999999998
"Collingwood, Victoria, Australia"	87.42105263157896
"Thornbury, Victoria, Australia"	87.3076923076923

Started streaming 10 records after 100 ms and completed after 100 ms.

Addional Q3. What is number of reviews per month?

```
MATCH (r:Review)
RETURN r.year, r.month, count(r) AS num_review
ORDER BY r.year, r.month;
```



neo4j@bolt://localhost:7687 - Neo4j Browser

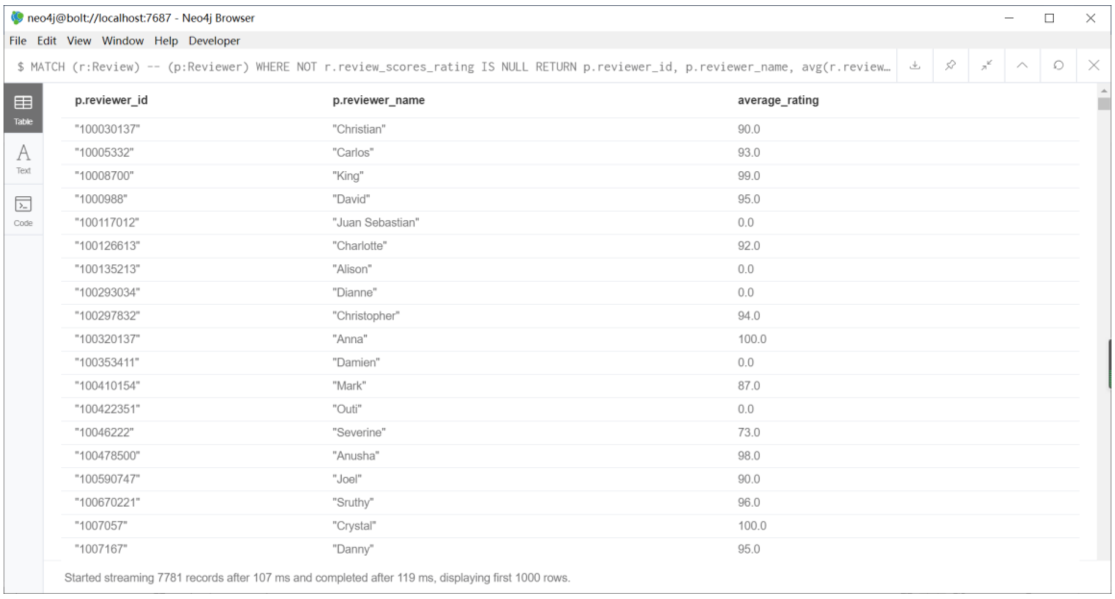
```
$ MATCH (r:Review) RETURN r.year, r.month, count(r) AS num_review ORDER BY r.year, r.month;
```

r.year	r.month	num_review
2010	8	1
2010	9	1
2010	10	3
2010	11	5
2010	12	6
2011	1	5
2011	2	7
2011	3	23
2011	4	21
2011	5	20
2011	6	22
2011	7	19
2011	8	26
2011	9	30
2011	10	41
2011	11	50
2011	12	47
2012	1	58
2012	2	54

Started streaming 108 records after 17 ms and completed after 40 ms.

Addional Q4. What the average rate that each reviewer give?

```
MATCH (r:Review) -- (p:Reviewer)
WHERE NOT r.review_scores_rating IS NULL
RETURN p.reviewer_id, p.reviewer_name, avg(r.review_scores_rating) AS average_rating
ORDER BY p.reviewer_id;
```



neo4j@bolt://localhost:7687 - Neo4j Browser

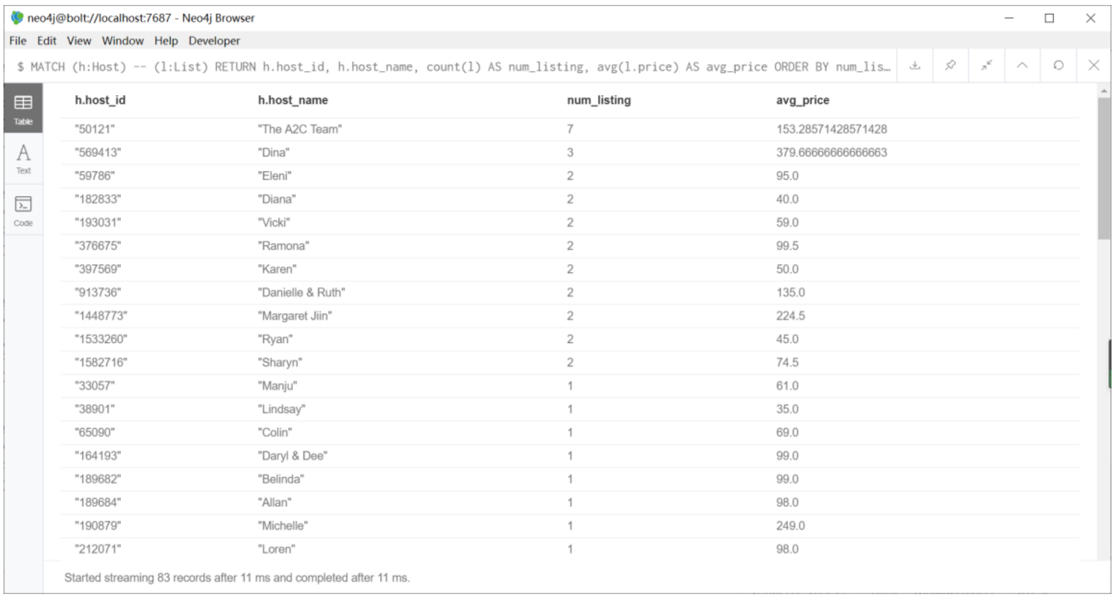
```
$ MATCH (r:Review) -- (p:Reviewer) WHERE NOT r.review_scores_rating IS NULL RETURN p.reviewer_id, p.reviewer_name, avg(r.review…
```

p.reviewer_id	p.reviewer_name	average_rating
"100030137"	"Christian"	90.0
"10005332"	"Carlos"	93.0
"10008700"	"King"	99.0
"1000988"	"David"	95.0
"100117012"	"Juan Sebastian"	0.0
"100126613"	"Charlotte"	92.0
"100135213"	"Alison"	0.0
"100293034"	"Dianne"	0.0
"100297832"	"Christopher"	94.0
"100320137"	"Anna"	100.0
"100353411"	"Damien"	0.0
"100410154"	"Mark"	87.0
"100422351"	"Outi"	0.0
"10046222"	"Severine"	73.0
"100478500"	"Anusha"	98.0
"100590747"	"Joel"	90.0
"100670221"	"Sruthy"	96.0
"1007057"	"Crystal"	100.0
"1007167"	"Danny"	95.0

Started streaming 7781 records after 107 ms and completed after 119 ms, displaying first 1000 rows.

Addional Q5. What is number of accommodation and avgerage price of accommodation of each host?

```
MATCH (h:Host) -- (l:List)
RETURN h.host_id, h.host_name, count(l) AS num_listing, avg(l.price) AS avg_price
ORDER BY num_listing DESC;
```



neo4j@bolt://localhost:7687 - Neo4j Browser

```
$ MATCH (h:Host) -- (l:List) RETURN h.host_id, h.host_name, count(l) AS num_listing, avg(l.price) AS avg_price ORDER BY num_lis…
```

h.host_id	h.host_name	num_listing	avg_price
"50121"	"The A2C Team"	7	153.28571428571428
"569413"	"Dina"	3	379.66666666666663
"59786"	"Eleni"	2	95.0
"182833"	"Diana"	2	40.0
"193031"	"Vicki"	2	59.0
"376675"	"Ramona"	2	99.5
"397569"	"Karen"	2	50.0
"913736"	"Danielle & Ruth"	2	135.0
"1448773"	"Margaret Jiin"	2	224.5
"1533260"	"Ryan"	2	45.0
"1582716"	"Sharyn"	2	74.5
"33057"	"Manju"	1	61.0
"38901"	"Lindsay"	1	35.0
"65090"	"Colin"	1	69.0
"164193"	"Daryl & Dee"	1	99.0
"189682"	"Belinda"	1	99.0
"189684"	"Allan"	1	98.0
"190879"	"Michelle"	1	249.0
"212071"	"Loren"	1	98.0

Started streaming 83 records after 11 ms and completed after 11 ms.

C.3 Database Modifications

1. Go to AirBnB website and add three new listings, including the hosts details and some related reviews of the listings you chose. The IDs in this case can be assigned manually by yourself.

Links of 3 listing:

1.https://www.airbnb.com.au/rooms/17465305?location=Melbourne%2C%20Victoria&source_impression_id=p3_1571471684_x5DzWDLotJ0wpaON

2.https://www.airbnb.com.au/rooms/28907077?location=Melbourne%2C%20Victoria&source_impression_id=p3_1571472315_yRtC4GK5bctqyn7A

3.https://www.airbnb.com.au/rooms/28630130?source_impression_id=p3_1571566656_GNiZmhPKNPlp5wNS

Links of 2 host (listing 1 belong to host 1, and listing 2 belong to host 2, list 3 belong to host 3)

1.https://www.airbnb.com.au/users/show/98798133

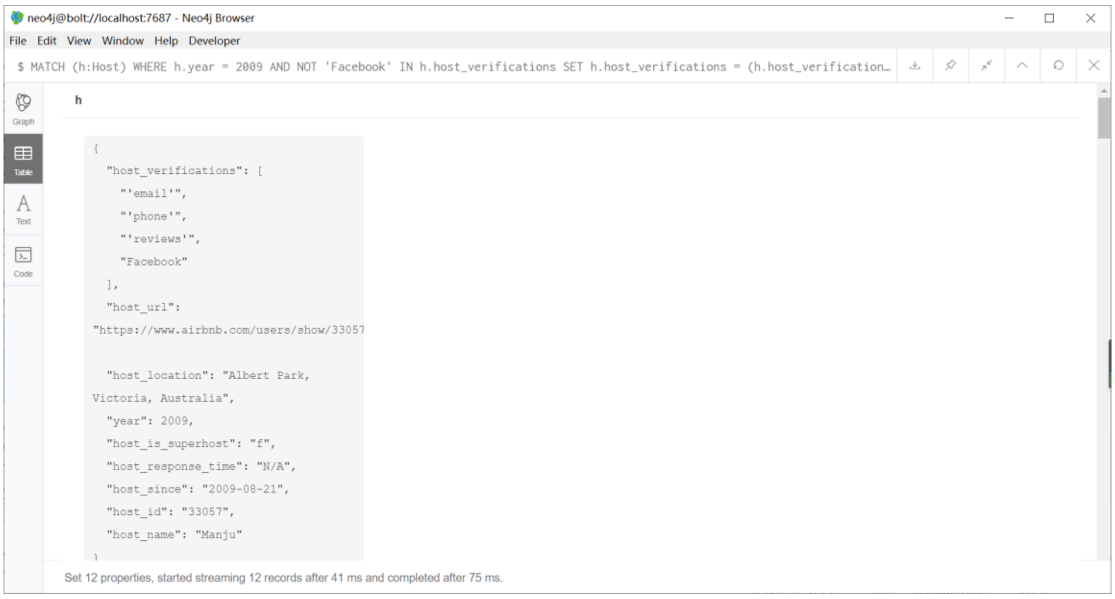
2.https://www.airbnb.com.au/users/show/41369546

3. https://www.airbnb.com.au/users/show/153057309

```
CREATE (h:Host {host_id: '98798133',
                    host_url: 'https://www.airbnb.com.au/users/show/98798133',
                    host_name: 'Hadi',
                    host_verifications: ['Government ID', 'Email address', 'Phone number'],
                    host_since: '2016/10/08',
                    year: 2016,
                    host_location: 'Melbourne, Australia',
                    host_response_time: 'N/A',
                    host_is_superhost: 'f'});

CREATE (h:Host {host_id: '41369546',
                   host_url: 'https://www.airbnb.com.au/users/show/41369546',
                   host_name: 'Wendy',
                   host_verifications: ['Government ID', 'Selfie', 'Email address', 'Phone
number'],
                   host_since: '2015/08/13',
                   year: 2015,
                   host_location: 'Melbourne, Australia',
                   host_response_time: 'N/A',
                   host_is_superhost: 'f'});

CREATE (h:Host {host_id: '153057309',
                   host_url: 'https://www.airbnb.com.au/users/show/153057309',
                   host_name: 'Bryan And Soraida',
                   host_verifications: ["Government ID","Selfie","Email address","Phone
number"],
                   host_since: '2017/10/03',
                   year: 2017,
                        host_location: 'Melbourne, Australia',
                        host_response_time: 'N/A',
                        host_is_superhost: 't'});


CREATE (l:List {list_id: '1',
                      name: 'Cityview Master Bedroom in The Green Abode',
                      summary: 'Located in the heart of Melbourne with a vibrant and thriving
lifestyle. Guest(s) will stay in the master bedroom furnished with king size mattress
complete with 2 pillows and bed sheets. The common spaces such as living room and
kitchen area are fully furnished dominated by minimalist style furnitures and greeneries.
Perfect for solo nomad or couple. This unit opens its door to any types of couple or
individual.',
                      listing_url: 'https://www.airbnb.com.au/rooms/17465305',
                      picture_url: 'N/A',
                      neighbourhood: 'Melbourne',
                      street: 'Melbourne, VIC, Australia',
                      zipcode: '3000',
                      latitude: -37.816114,
                      longitude: 144.953123,
                      room_type: 'Private room',
                      amenities: ['Wifi', 'Dryer', 'Air conditioning', 'Washing mashine',
'Essentials', 'TV', 'Heating', 'Hot water', 'Lift', 'Gym', 'Pool', 'Paid parking off premisses',
'Free street parking', 'Microwave', 'Refrigerator', 'Oven', 'Stove'],
                      price: 50,
                      extra_people: 0,
                      minimum_nights: 1,
                      calculated_host_listings_count: 1,
                      availability_365: 120});

CREATE (l:List {list_id: '2',
                      name: 'Eco-friendly Studio*Private bathroom*Wifi*Pool*Gym',
                      summary: 'A cozy and homey studio located in the CBD, perfect for
tourists and professionals. Easily accessible by public transports (free tram zone, 100m
Melbourne Central Station). Close to some top rated attractions (State Library, Royal
Exhibition Building, Parliament House), two steps away from QV Center, China Town,
restaurants, cafes, pubs. Swimming pool, sauna, gym are available in the building.' ,
                      listing_url: 'https://www.airbnb.com.au/rooms/28907077',
                      picture_url: 'N/A',
                      neighbourhood: 'Melbourne',
                      street: 'Melbourne, VIC, Australia',
                      zipcode: '3000',
                      latitude: -37.810159,
                      longitude: 144.967390,
                      room_type: 'Entire home/apt',
                      amenities: ['Wifi', 'Essentials', 'TV', 'Heating', 'Hot water', 'Lift', 'Gym',
'Pool', 'Paid parking off premisses', 'Microwave', 'Refrigerator', 'Stove'],
                      price: 67,
                      extra_people: 20,
                      minimum_nights: 1,
                      calculated_host_listings_count: 2,
                      availability_365: 90});

CREATE (l:List {list_id: '3',
                      name: 'Studio~Best Memories in CBD ♡ *Free Tram Zone!',
                      summary: 'N/A',
                      listing_url: 'https://www.airbnb.com.au/rooms/28630130',
                      picture_url: 'N/A',
                      neighbourhood: 'Melbourne',
                      street: 'Melbourne, VIC, Australia',
                      zipcode: '3000',
                      latitude: -37.809941,
                      longitude: 144.958056,
                      room_type: 'Entire home/apt',
                      amenities: ['Wifi', 'Essentials', 'TV', 'Heating', 'Hot water', 'Lift', 'Gym',
'Pool', 'Paid parking off premisses', 'Microwave', 'Refrigerator', 'Stove'],
                      price: 50,
                      extra_people: 0,
                      minimum_nights: 2,
                      calculated_host_listings_count: 2,
                      availability_365: 120});

CREATE (r:Review {review_id: '1',
                         year: 2019,
                         month: 8,
                         day: 1,
                         review_scores_rating: 95,
                         comments: 'Very clean place and supe close to everything. So much
great food and bars around'});

CREATE (r:Reviewer {reviewer_id: '1', reviewer_name: 'Joel'});

CREATE (r:Review {review_id: '2',
                         year: 2019,
                         month: 10,
                         day: 1,
                         review_scores_rating: 96,
                         comments: "It's a good place for a short trip to Melbourne. Very
convenient to get around everywhere."});

CREATE (r:Reviewer {reviewer_id: '2', reviewer_name: 'Chia Jane'});
CREATE (r:Review {review_id: '3',
                          year: 2019,
                          month: 10,
                          day: 04,
                          review_scores_rating: 98,
                          comments: 'Absolutely amazing spot for our trip :D'});

CREATE (r:Reviewer {reviewer_id: '3', reviewer_name: 'Alex'});

MATCH (l:List{list_id: '1'})
MATCH (h:Host{host_id: '98798133'})
CREATE (h)-[:own]->(l);

MATCH (l:List{list_id: '2'})
MATCH (h:Host{host_id: '41369546'})
CREATE (h)-[:own]->(l);

MATCH (l:List{list_id: '3'})
MATCH (h:Host{host_id: '153057309'})
CREATE (h)-[:own]->(l);

MATCH (l:List{list_id: '1'})
MATCH (r:Review{review_id: '1'})
CREATE (r)-[:review]->(l);

MATCH (l:List{list_id: '2'})
MATCH (r:Review{review_id: '2'})
CREATE (r)-[:review]->(l);

MATCH (l:List{list_id: '3'})
MATCH (r:Review{review_id: '3'})
CREATE (r)-[:review]->(l);

MATCH (p:Reviewer {reviewer_id: '1'})
MATCH (r:Review {review_id: '1'})
CREATE (p)-[:write]->(r);

MATCH (p:Reviewer {reviewer_id: '2'})
MATCH (r:Review {review_id: '2'})
CREATE (p)-[:write]->(r);

MATCH (p:Reviewer {reviewer_id: '3'})
MATCH (r:Review {review_id: '3'})
CREATE (p)-[:write]->(r);
```

2. Update the host verification for those who registered in 2009 and add Facebook to the list of existing verifications.

```
MATCH (h:Host)
WHERE h.year = 2009
        AND NOT 'Facebook' IN  h.host_verifications
SET h.host_verifications = (h.host_verifications + 'Facebook')
RETURN h;
```


neo4j@bolt://localhost:7687 - Neo4j Browser
File Edit View Window Help Developer
Graph
Table
Text
Code

```
$ MATCH (h:Host) WHERE h.year = 2009 AND NOT 'Facebook' IN h.host_verifications SET h.host_verifications = (h.host_verification…
```

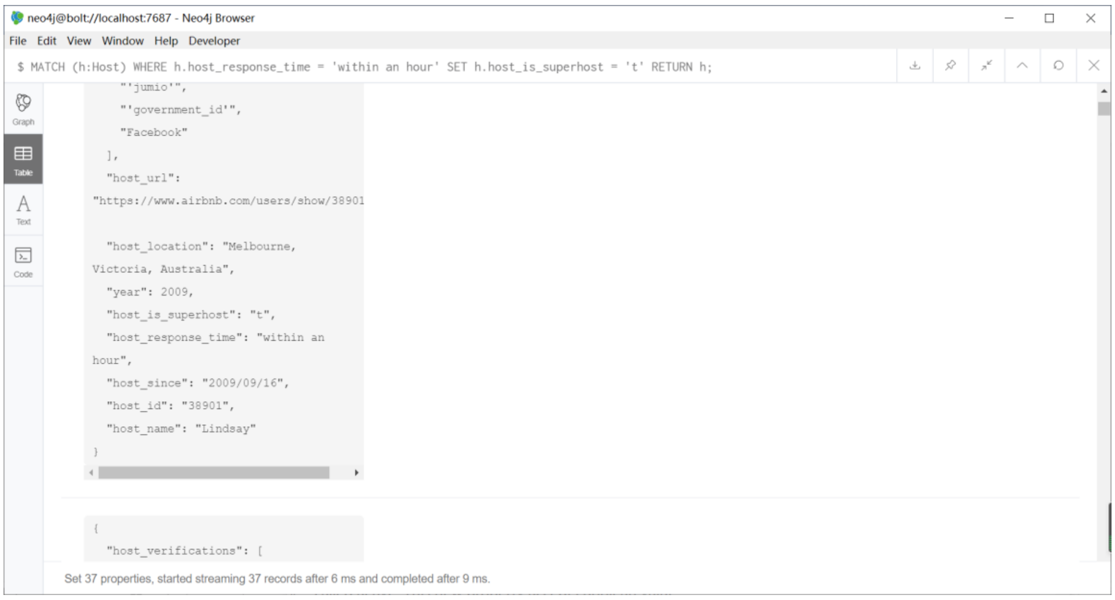
h

```
{
  "host_verifications": [
    "'email'",
    "'phone'",
    "'reviews'",
    "Facebook"
  ],
  "host_url":
"https://www.airbnb.com/users/show/33057

  "host_location": "Albert Park,
Victoria, Australia",
  "year": 2009,
  "host_is_superhost": "f",
  "host_response_time": "N/A",
  "host_since": "2009-08-21",
  "host_id": "33057",
  "host_name": "Manju"
}
```

Set 12 properties, started streaming 12 records after 41 ms and completed after 75 ms.

3. Update hosts who respond "within an hour" to a superhost. For this update you may only use the "host response time" and "host is a super host" information.

```
MATCH (h:Host)
WHERE h.host_response_time = 'within an hour'
SET h.host_is_superhost = 't'
RETURN h;
```


neo4j@bolt://localhost:7687 - Neo4j Browser
File Edit View Window Help Developer
Graph
Table
Text
Code

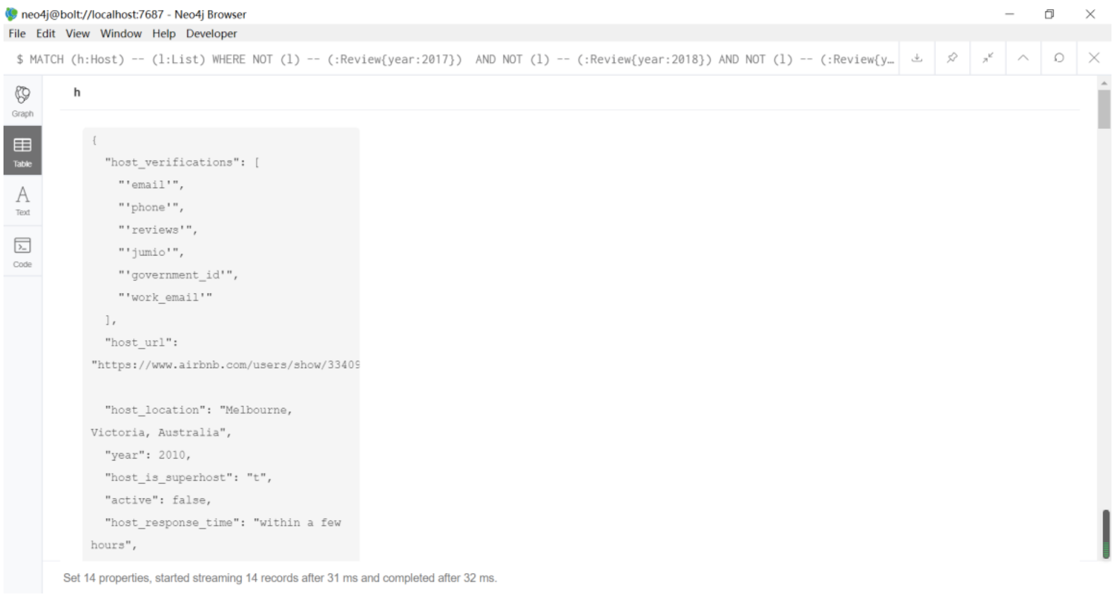
```
$ MATCH (h:Host) WHERE h.host_response_time = 'within an hour' SET h.host_is_superhost = 't' RETURN h;
```

```
    "'jumio'",
    "'government_id'",
    "Facebook"
  ],
  "host_url":
"https://www.airbnb.com/users/show/38901

  "host_location": "Melbourne,
Victoria, Australia",
  "year": 2009,
  "host_is_superhost": "t",
  "host_response_time": "within an
hour",
  "host_since": "2009/09/16",
  "host_id": "38901",
  "host_name": "Lindsay"
}

{
  "host_verifications": [
```

Set 37 properties, started streaming 37 records after 6 ms and completed after 9 ms.

4. Update hosts who do not receive any reviews for their accommodation since 2017 and add a new property called active. This new property accepts Boolean value.

```
MATCH (h:Host) -- (l:List)
WHERE NOT (l) -- (:Review{year:2017})
        AND NOT (l) -- (:Review{year:2018})
        AND NOT (l) -- (:Review{year:2019})
SET h.active = false
RETURN h;
```



neo4j@bolt://localhost:7687 - Neo4j Browser

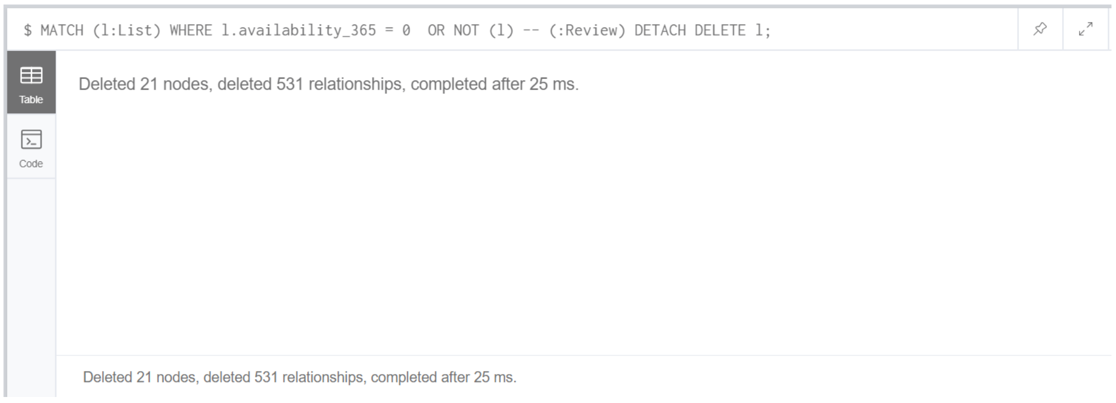
```
$ MATCH (h:Host) -- (l:List) WHERE NOT (l) -- (:Review{year:2017})  AND NOT (l) -- (:Review{year:2018}) AND NOT (l) -- (:Review{y…
```

```
h

{
  "host_verifications": [
    "'email'",
    "'phone'",
    "'reviews'",
    "'jumio'",
    "'government_id'",
    "'work_email'"
  ],
  "host_url":
"https://www.airbnb.com/users/show/33409

  "host_location": "Melbourne,
Victoria, Australia",
  "year": 2010,
  "host_is_superhost": "t",
  "active": false,
  "host_response_time": "within a few
hours",
```

Set 14 properties, started streaming 14 records after 31 ms and completed after 32 ms.

5. Delete all listings with zero availability and have no reviews.

```
MATCH (l:List)
WHERE l.availability_365 = 0
        OR NOT (l) -- (:Review)
DETACH DELETE l
RETURN l;
```



```
$ MATCH (l:List) WHERE l.availability_365 = 0  OR NOT (l) -- (:Review) DETACH DELETE l;
```

Deleted 21 nodes, deleted 531 relationships, completed after 25 ms.

Deleted 21 nodes, deleted 531 relationships, completed after 25 ms.

C.4 Advanced Topic

As we design the MoanshBnB, the most similar, successful and famous travel accommodation booking systems is AirBnB. AirBnB provide the service all over world. They are facing large volume, high velocity, high variety data every day. Since the data in AirBnB is not just data itself, AirBnB also more focus on the relationship of data. In that case, the graph database is very suitable for store the data and its relationship. Since the Neo4j is the one of best graph database, so AirBnB use Neo4j for store the data.

In the blog of Bodley, J & Williams, C, AirBnB start up to store data into Hive data warehouse, trying to create the database similar as graph database in the Hive data warehouse. Then, the data from Hive data warehouse have been through a tool called Airflow to transform the data, Python did the further processing to help data fit into neo4j. The neo4j driver input the data into neo4j, the user can use a python web framework called 'Flask' to connect to a search engine called 'Elasticsearch' to search, or in other word querying. In this framework, the Hive data warehouse is used to store the raw data. Airflow and Python are used as data stream process and data wrangling. Neo4j is the main database used to store the data. The Flask and Elasticsearch are used for query or search.

For our system MonashBnB, the volume of data may not be very big, and the velocity of incoming data also may not be very high, but we need to prepare for big data situation. We can build similar framework as AirBnB. Since our data volume and velocity is not big or high as AirBnB, we may not need data warehouse to store the data first, we can use framework like apache spark to handle the income data. Since Python is very suitable for neo4j, we can use Python for data wrangling and transfer the data, and finally storing data into neo4j. That is my first suggestion.

For second suggestion, the data form in current data is inconsistent. For example, the location of some host contains city, state and country like host 38901 in 'Melbourne, Victoria, Australia', but some host like host 33057 only write 'AU', it will cause problem when we do some query. We may create the property like city, state and country to make it clear.

The third suggestion is about the standard of data in different table. In the listing data, this problem also exists. The location recorded in street. Although it stores location as in host, but the standard is different. The state is stored as abbreviation in listing, but full name in host., it causes the inconsistent to cause problem in future query. The name of city also is inconsistent, for example, there is another name 'Saint Kilda East' for 'St. Kilda East'. As solution, we need to perform the data wrangling of data before storing it in database to ensure the data have same standard.

For fourth suggestion, we may need to combine the reviewer and host table together. Currently, we have two separate tables (labels) for host and reviewer, but the host also can be reviewer. For example, the host named 'Timmy', and system have recorded the name 'Timmy' and unique id in host table (label), and recorded the name 'Tim' and another unique id in reviewer table (label). As result, it may cause inconsistency and duplication in the database, the solution is that create table (label) called user to combine the data from host and reviewer. That also avoid the host use another name to review their own listing.

Reference

Bodley, J & Williams, C. (2018). Democratizing Data Discovery at Airbnb. Retrieved from https://neo4j.com/blog/democratizing-data-discovery-airbnb/