**FIT5137 Group Assignment Report**

**Semester 2, 2019**

**Students Name: Dawei Gu, Jiexuan Du**

**Students ID: 29910226, 29218020**

**Tutorial Section: FIT5137 Laboratory 05,**

**12:00m-14:00pm Tue**

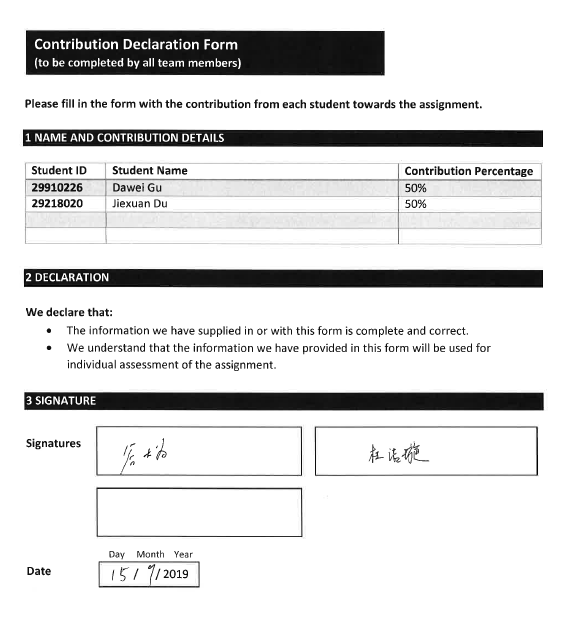
**Tutor: Agnes Haryanto**

**Contents**

1. **Signed Contribution Declaration Form………………………… 2**
2. **Contribution Declaration Detail…………………...……………. 3**
3. **Report**

* **C.1 Database Design………………………………………….. 4**
* **C.2 Database Modifications………………………………… 11**
* **C.3 Queries…………………………...……………………… 14**
* **C.4 Database Comparison…………………………………... 32**

**Signed Contribution Declaration Form**



**Contribution Declaration Detail**

Percentage of contribution:

* Name: Dawei Gu, ID: 29910226, Contribution: 50%
* Name: Jiexuan Du, ID: 29218020, Contribution: 50%

List of parts that each student did:

I. Dawei Gu: list the parts that Dawei did:

- Task C.1 (MongoDB Host data create collections and insert data,

Cassandra create table and insert data)

- Task C.2 (1, 3, 5, 7)

- Task C.3 (1, 3, 5, 7, 9, 11, 13, 15, 17, 19, Q1, Q2, Q4)

- Task C.4

□ Task C.3

II. Jiexuan Du: list the parts that Jiexuan did:

- Task C.1 (MongoDB listing data create collections and insert data

Cassandra create table)

- Task C.2 (2, 4, 6)

- Task C.3 (2, 4, 6, 8, 10, 12, 14, 16, 18, 20, Q3, Q5)

- Task C.4

**Report**

**C.1 Database Design**

**MongoDB:**

# create database

use FIT5137\_Assign

#create collection host

db.createCollection('Host')

#insert data to host

db.Host.insertMany([

{'host\_id': 'MONHOS01', 'host\_name':'Manju',

'host\_vertifications':['email', 'phone', 'reviews'],

'host\_since': new Date('2009-08-21'),

'host\_location': {'city': 'Clayton', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within a day', 'host\_is\_superhost':false},

{'host\_id': 'MONHOS02', 'host\_name':'Lindsay',

'host\_vertifications':['email', 'phone', 'reviews', 'jumio', 'government id'],

'host\_since': new Date('2009-09-16'),

'host\_location': {'city': 'Clifton Hill', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within an hour', 'host\_is\_superhost':true},

{'host\_id': 'MONHOS03', 'host\_name':'Adam',

'host\_vertifications':['email', 'phone', 'google', 'reviews', 'jumio', 'government id', 'work email'],

'host\_since': new Date('2009-10-31'),

'host\_location': {'city': 'Port Melbourne', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within an hour', 'host\_is\_superhost':false},

{'host\_id': 'MONHOS04', 'host\_name':'Eleni',

'host\_vertifications':['email', 'phone', 'facebook', 'reviews', 'jumio', 'offline government id', 'government id', 'work email'],

'host\_since': new Date('2009-12-03'),

'host\_location': {'city': 'Fitzroy', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within a day', 'host\_is\_superhost':false},

{'host\_id': 'MONHOS05', 'host\_name':'Colin',

'host\_vertifications':['email', 'phone', 'facebook', 'reviews', 'jumio', 'offline government id', 'government id'],

'host\_since': new Date('2009-12-22'),

'host\_location': {'city': 'Saint Kilda East', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within an hour', 'host\_is\_superhost':false},

{'host\_id': 'MONHOS06', 'host\_name':'Daryl',

'host\_vertifications':['email', 'phone', 'manual online', 'reviews', 'manual offline', 'work email'],

'host\_since': new Date('2010-07-12'),

'host\_location': {'city': 'Berwick', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within an hour', 'host\_is\_superhost':true},

{'host\_id': 'MONHOS07', 'host\_name':'Diana',

'host\_vertifications':['email', 'phone', 'facebook', 'reviews', 'jumio', 'offline government id', 'government id', 'work email'],

'host\_since': new Date('2010-07-27'),

'host\_location': {'city': 'Parkdale', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within a day', 'host\_is\_superhost':false},

{'host\_id': 'MONHOS08', 'host\_name':'Belinda',

'host\_vertifications':['email', 'phone', 'facebook', 'reviews', 'jumio', 'offline government id', 'selfie', 'government id', 'identity manual', 'work email'],

'host\_since': new Date('2010-08-03'),

'host\_location': {'city': 'Prahran', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within a few hours', 'host\_is\_superhost':false},

{'host\_id': 'MONHOS09', 'host\_name':'Allan',

'host\_vertifications':['email', 'phone', 'facebook', 'reviews', 'jumio', 'offline government id', 'selfie', 'government id', 'identity manual'],

'host\_since': new Date('2010-08-03'),

'host\_location': {'city': 'South Melbourne', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within an hour', 'host\_is\_superhost':true},

{'host\_id': 'MONHOS10', 'host\_name':'Vicki',

'host\_vertifications':['email', 'phone', 'facebook', 'reviews', 'jumio', 'government id'],

'host\_since': new Date('2010-08-06'),

'host\_location': {'city': 'Frankston', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within an hour', 'host\_is\_superhost':true}

])

#create collection listing

db.createCollection('Listing')

#insert data into listing

db.Listing.insertMany([

{'listing\_id':'MONLST01', 'name':'Monash Beautiful House',

'host\_id':'MONHOS14', 'neighbourhood':'Manningham',

'address':{'city':'Clayton', 'state':'VIC', 'post\_code':3800},

'latitude': -37.773, 'longitude':145.09213, 'room\_type':'Entire home',

'amenities':['TV', 'Wifi', 'Pets Allowed', 'Family friendly', '24-hour check-in', 'Self check-in'],

'price':61, 'extra\_people':22, 'minimum\_nights':1, 'availability\_365':365},

{'listing\_id':'MONLST02', 'name':'Monash Brunswick Deco',

'host\_id':'MONHOS08', 'neighbourhood':'Moreland',

'address':{'city':'Brunswick East', 'state':'VIC', 'post\_code':3057},

'latitude': -37.767, 'longitude':144.98074, 'room\_type':'Private room',

'amenities':['Kitchen', 'Toiletries', 'Hair dryer', 'Iron', 'Microwave', 'Coffee Maker', 'Refrigerator', 'Cooking basics', 'Stove', 'Garden'],

'price':35, 'extra\_people':15, 'minimum\_nights':3, 'availability\_365':194},

{'listing\_id':'MONLST03', 'name':'Monash Beachside Retreat',

'host\_id':'MONHOS01', 'neighbourhood':'Port Phillip',

'address':{'city':'St Kilda', 'state':'VIC', 'post\_code':3182},

'latitude': -37.86, 'longitude':144.97737, 'room\_type':'Entire home',

'amenities':['Cooking basics', 'Oven', 'Stove', 'Dishwasher', 'Balcony', 'Wifi', 'Kitchen', 'Washer', 'Toiletries', 'Hair dryer', 'Iron', 'Microwave', 'Coffee Maker', 'Refrigerator', 'Long term stay allowed', 'Luggage dropoff', '24-hour check-in', 'Self check-in'],

'price':159, 'extra\_people':0, 'minimum\_nights':2, 'availability\_365':82},

{'listing\_id':'MONLST04', 'name':'Monash Close2City',

'host\_id':'MONHOS04', 'neighbourhood':'Darebin',

'address':{'city':'Thornbury', 'state':'VIC', 'post\_code':3071},

'latitude': -37.759, 'longitude':144.98923, 'room\_type':'Private room',

'amenities':['TV', 'Internet', 'Wifi', 'Kitchen', 'Free parking on premises', 'Breakfast', 'Heating', 'Washer', 'Toiletries', 'Hair dryer'],

'price':50, 'extra\_people':20, 'minimum\_nights':2, 'availability\_365':0},

{'listing\_id':'MONLST05', 'name':'Monash City and Sports',

'host\_id':'MONHOS05', 'neighbourhood':'Port Phillip',

'address':{'city':'St Kilda East', 'state':'VIC', 'post\_code':3183},

'latitude': -37.865, 'longitude':144.99224, 'room\_type':'Private room',

'amenities':['Internet', 'Wifi', 'Heating', 'Washer', 'Toiletries', 'Bed linens', 'Gym Access', '24-hour check-in', 'Self check-in'],

'price':69, 'extra\_people':20, 'minimum\_nights':1, 'availability\_365':274},

{'listing\_id':'MONLST06', 'name':'Monash Trafford Apartment',

'host\_id':'MONHOS06', 'neighbourhood':'Casey',

'address':{'city':'Berwick', 'state':'VIC', 'post\_code':3806},

'latitude': -38.057, 'longitude':145.33936, 'room\_type':'Entire home',

'amenities':['Microwave', 'Coffee Maker', 'Wifi', 'Kitchen', 'Toiletries', 'Hair dryer'],

'price':99, 'extra\_people':30, 'minimum\_nights':1, 'availability\_365':353},

{'listing\_id':'MONLST07', 'name':'Monash Close2Airport',

'host\_id':'MONHOS07', 'neighbourhood':'Darebin',

'address':{'city':'Reservoir', 'state':'VIC', 'post\_code':3073},

'latitude': -37.697, 'longitude':145.00082, 'room\_type':'Private room',

'amenities':['Long term stay allowed', 'Luggage dropoff', '24-hour check-in', 'Self check-in', 'Airport shuttle service', 'Host greets you', 'Wifi', 'Kitchen', 'Heating', 'Toiletries', 'Microwave', 'Coffee Maker', 'Refrigerator', 'Cooking basics', 'Oven', 'Stove', 'Dishwasher', 'Garden'],

'price':50, 'extra\_people':20, 'minimum\_nights':7, 'availability\_365':0},

{'listing\_id':'MONLST08', 'name':'Monash Home In The City',

'host\_id':'MONHOS02', 'neighbourhood':'Melbourne',

'address':{'city':'East Melbourne', 'state':'VIC', 'post\_code':3002},

'latitude': -37.81, 'longitude':144.98592, 'room\_type':'Private room',

'amenities':['Cooking basics', 'TV', 'Wifi', 'Pets Allowed', 'Family friendly', '24-hour check-in', 'Self check-in'],

'price':99, 'extra\_people':25, 'minimum\_nights':15, 'availability\_365':62},

{'listing\_id':'MONLST09', 'name':'Monash Japanese-Style',

'host\_id':'MONHOS11', 'neighbourhood':'Monash',

'address':{'city':'Oakleigh East', 'state':'VIC', 'post\_code':3166},

'latitude': -37.9, 'longitude':145.11447, 'room\_type':'Entire home',

'amenities':['Kitchen', 'Toiletries', 'Hair dryer', 'Iron', 'Microwave', 'Garden', 'Coffee Maker', 'Refrigerator', 'Cooking basics', 'Stove', 'Dishwasher', 'Balcony', 'Paid Parking', 'Long term stay allowed'],

'price':98, 'extra\_people':0, 'minimum\_nights':2, 'availability\_365':219},

{'listing\_id':'MONLST10', 'name':'Beautiful Monash House',

'host\_id':'MONHOS10', 'neighbourhood':'Frankston',

'address':{'city':'Frankston', 'state':'VIC', 'post\_code':3199},

'latitude': -38.149, 'longitude':145.14157, 'room\_type':'Entire home',

'amenities':['TV', 'Wifi', 'AC', 'Kitchen', 'Heating', 'Washer', 'Toiletries', 'Hair dryer', 'Iron', 'Microwave', 'Coffee Maker', 'Refrigerator', 'Cooking basics', 'Oven', 'Stove', 'Dishwasher', 'Balcony', 'Paid Parking', 'Long term stay allowed', 'Luggage dropoff', '24-hour check-in', 'Self check-in'],

'price':59, 'extra\_people':10, 'minimum\_nights':2, 'availability\_365':318},

{'listing\_id':'MONLST11', 'name':'Fabulous Monash Richmond',

'host\_id':'MONHOS09', 'neighbourhood':'Yarra',

'address':{'city':'Richmond,', 'state':'VIC', 'post\_code':3121},

'latitude': -37.818, 'longitude':145.00442, 'room\_type':'Entire home',

'amenities':['Bed linen', 'Wifi', 'AC', 'Kitchen', 'Heating', 'Washer', 'Toiletries', 'Hair dryer', 'Iron', 'Microwave', 'Coffee Maker', 'Free Parking', 'Long term stay allowed', 'Luggage dropoff', '24-hour check-in', 'Self check-in', 'Host greets you', 'Refrigerator', 'Cooking basics', 'Oven', 'Stove', 'Dishwasher', 'Balcony', 'Garden'],

'price':98, 'extra\_people':30, 'minimum\_nights':14, 'availability\_365':16},

{'listing\_id':'MONLST12', 'name':'Monash Central Lux',

'host\_id':'MONHOS12', 'neighbourhood':'Port Phillip',

'address':{'city':'St Kilda', 'state':'VIC', 'post\_code':3182},

'latitude': -37.861, 'longitude':144.98038, 'room\_type':'Entire home',

'amenities':['Wifi', 'AC', 'Kitchen', 'Heating', 'Toiletries', 'Hair dryer', 'Iron', 'Coffee Maker', 'Refrigerator', 'Stove', 'Dishwasher', 'Beachside view', 'Free Parking 24-hour check-in', 'Self check-in'],

'price':189, 'extra\_people':29, 'minimum\_nights':2, 'availability\_365':6},

{'listing\_id':'MONLST13', 'name':'Central Monash Warehouse Apartment',

'host\_id':'MONHOS13', 'neighbourhood':'Melbourne',

'address':{'city':'Melbourne', 'state':'VIC', 'post\_code':3000},

'latitude': -37.815, 'longitude':144.96267, 'room\_type':'Entire home',

'amenities':['Dishwasher', 'Garden', 'Paid Parking', 'Long term stay allowed', 'Coffee Maker', 'Refrigerator', 'Cooking basics', 'Oven', 'Stove', 'Wifi', 'AC', 'Kitchen', 'Heating', 'Washer', 'Toiletries', 'Hair dryer', 'Iron', 'Microwave', 'Luggage dropoff', '24-hour check-in', 'Self check-in'],

'price':249, 'extra\_people':40, 'minimum\_nights':2, 'availability\_365':353},

{'listing\_id':'MONLST14', 'name':'Monash Near the Park',

'host\_id':'MONHOS03', 'neighbourhood':'Bayside',

'address':{'city':'Melbourne', 'state':'VIC', 'post\_code':3187},

'latitude': -37.928, 'longitude':145.02518, 'room\_type':'Private room',

'amenities':['Pets Allowed', 'Breakfast', 'Garden', 'Family friendly', '24-hour check-in', 'Self check-in'],

'price':40, 'extra\_people':11, 'minimum\_nights':2, 'availability\_365':365}

])

**Cassandra:**

# create keyspace

CREATE KEYSPACE FIT5137\_Assign with

replication = {'class':'SimpleStrategy', 'replication\_factor':1};

#swich to keyspace

USE FIT5137\_Assign;

#create table review

CREATE TABLE review (listing\_id text,

review\_id text,

review\_date date,

sequence time,

reviewer\_id text,

reviewer\_name text,

rating int,

reason set<text>,

comments text,

PRIMARY KEY (review\_id));

#create second index

CREATE INDEX ON review (reviewer\_id );

#insert data into reiview

BEGIN BATCH

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST02', 'REV01', '2017-03-22', '10:37:50', '500001', 'Miriam', 90, {'location', 'amenities'}, 'Beautiful View');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST02', 'REV02', '2017-03-22', '11:37:50', '500002', 'Johannes', 90, {'host'}, 'Good Host');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST02', 'REV03', '2017-03-22', '11:37:50', '500003', 'Camille', 100, {'location', 'view'}, 'Nice View and Location');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST02', 'REV04', '2017-03-22', '12:37:50', '500004', 'Paige', 95, {'price'}, 'Excellent Price');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST01', 'REV05', '2017-03-22', '15:37:50', '500005', 'Adele', 93, {'location', 'price'}, 'Good Location');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST03', 'REV06', '2017-03-22', '17:37:50', '500006', 'Greg', 87, {'host', 'view'}, 'Very Clean House');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST04', 'REV07', '2017-03-22', '19:37:50', '500007', 'Wolfgang', 91, {'location', 'price'}, 'Nice Location');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST05', 'REV08', '2017-03-22', '20:37:50', '500008', 'Klaus', 96, {'location', 'view'}, 'Nice Building');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST06', 'REV09', '2017-03-23', '11:37:50', '500009', 'Rox', 100, {'host', 'price'}, 'Friendly Host');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST05', 'REV10', '2017-03-23', '12:37:50', '500010', 'Elisabeth', 98, {'host', 'price'}, 'Friendly Host');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST09', 'REV11', '2017-03-23', '19:37:50', '500011', 'Derek', 100, {'space', 'clean'}, 'Very Clean and comfortable');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST09', 'REV12', '2017-03-25', '10:07:40', '500012', 'Joy', 92, {'host', 'clean'}, 'Friendly and Nice Host');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST10', 'REV13', '2017-03-26', '10:02:10', '500013', 'Anouck', 93, {'host', 'view'}, 'Very Comfortable');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST12', 'REV14', '2017-03-26', '10:49:40', '500014', 'Jerome', 85, {'location', 'clean'}, 'Friendly Host');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST13', 'REV15', '2017-03-26', '10:48:40', '500015', 'Jehan', 98, {'location', 'amenities'}, 'Beautiful View');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST14', 'REV16', '2017-03-26', '10:48:10', '500012', 'Joy', 97, {'amenities', 'view'}, 'Good Location');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST14', 'REV17', '2017-03-26', '10:47:40', '500014', 'Jerome', 30, {'price', 'view'}, 'Bad Location');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST10', 'REV18', '2017-03-26', '10:47:10', '500002', 'Johannes', 20, {'amenities', 'view'}, 'Bad Service');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST12', 'REV19', '2017-03-27', '09:37:50', '500013', 'Anouck', 87, {'space', 'clean'}, 'Good Location');

INSERT INTO review(listing\_id, review\_id, review\_date, sequence, reviewer\_id, reviewer\_name, rating, reason, comments)

VALUES('MONLST05', 'REV20', '2017-03-27', '10:45:10', '500011', 'Derek', 96, {'host', 'view'}, 'Nice Building');

APPLY BATCH;

**C.2 Database Modifications**

**MonogoDB:**

1.

Code:

db.Host.update({'host\_name':'Adam'}, {$push:{'host\_vertifications': 'facebook'}})



2.

Code:

db.Host.insertMany([

{'host\_id': 'MONHOS11', 'host\_name':'Alison',

'host\_vertifications':['email', 'phone','facebook', 'reviews'],

'host\_since': new Date('2019-01-9'),

'host\_location': {'city': 'Caulfield', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within an hour', 'host\_is\_superhost':false},

{'host\_id': 'MONHOS12', 'host\_name':'Mike',

'host\_vertifications':['email', 'phone'],

'host\_since': new Date('2019-01-9'),

'host\_location': {'city': 'Clayton', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within a day', 'host\_is\_superhost':true},

{'host\_id': 'MONHOS13', 'host\_name':'Robyn',

'host\_vertifications':['facebook', 'reviews'],

'host\_since': new Date('2019-01-9'),

'host\_location': {'city': 'Berwick', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within an hour', 'host\_is\_superhost':false},

{'host\_id': 'MONHOS14', 'host\_name':'Daniel',

'host\_vertifications':['email', 'manual offline','work email'],

'host\_since': new Date('2019-01-9'),

'host\_location': {'city': 'Frankston', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within a day', 'host\_is\_superhost':true},

{'host\_id': 'MONHOS15', 'host\_name':'Ron',

'host\_vertifications':['facebook'],

'host\_since': new Date('2019-01-9'),

'host\_location': {'city': 'Caulfield', 'state':'Victoria', 'country':'Australia'},

'host\_response\_rate':'within a day', 'host\_is\_superhost':false}

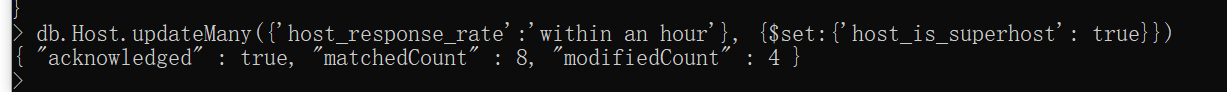
])



3.

Code:

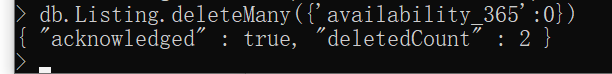
db.Host.updateMany({'host\_response\_rate':'within an hour'}, {$set:{'host\_is\_superhost': true}})



4.

Code:

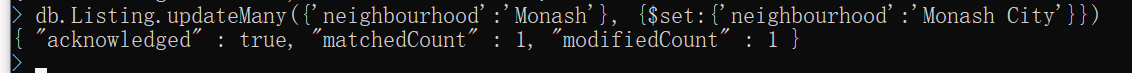
db.Listing.deleteMany({'availability\_365':0})



5.

Code:

db.Listing.updateMany({'neighbourhood':'Monash'}, {$set:{'neighbourhood':'Monash City'}})

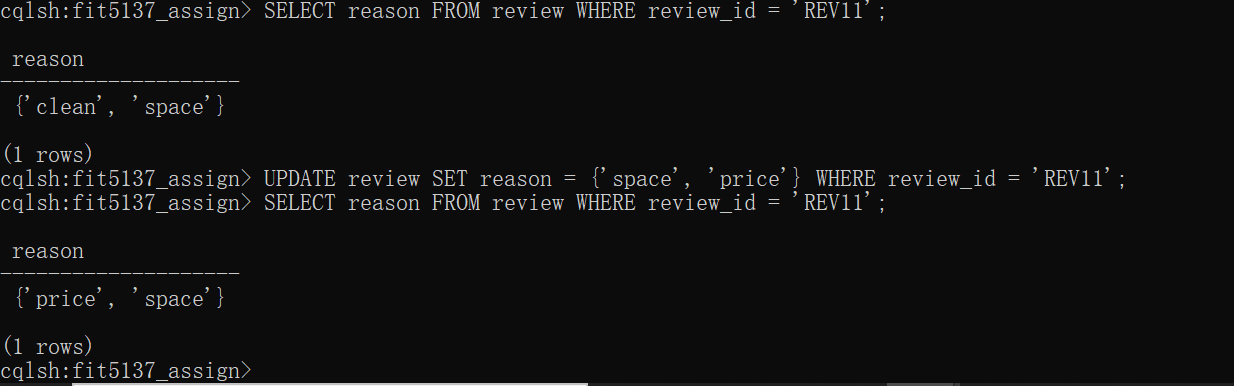


**Cassandra:**

6.

Code:

UPDATE review SET reason = {'space', 'price'} WHERE review\_id = 'REV11';

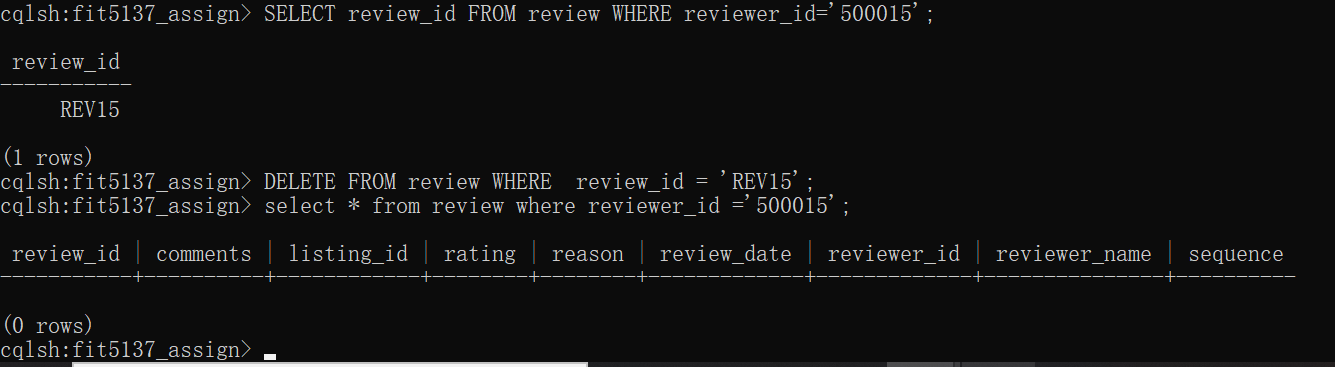


7.

Code:

SELECT review\_id FROM review WHERE reviewer\_id='500015';

DELETE FROM review WHERE review\_id = 'REV15';



**C.3 Queries**

**MongoDB**

#preparing

#create database

use FIT5137\_Assign\_C3

#quit the mongoDB

quit()

#insert data into Host

mongoimport --db FIT5137\_Assign\_C3 --collection Host --file D:\host.json

#insert data into Listing

mongoimport --db FIT5137\_Assign\_C3 --collection Listing --file D:\listing.json

#run shall

mongod

#enter shall

mongo

#enter database

use FIT5137\_Assign\_C3

#create embedding model collection called Listing\_host

db.Listing.aggregate([

{$lookup: {from:'Host',

localField:'host\_id',

foreignField:'host\_id',

as: 'host'}},

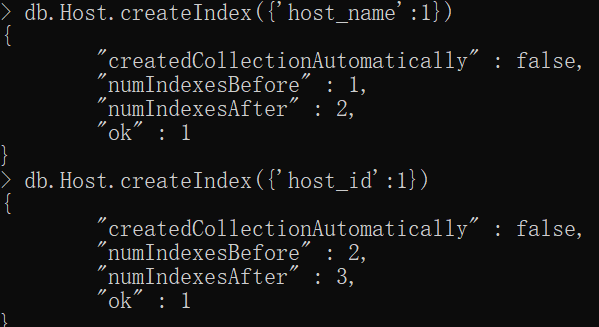
{$out: "Listing\_host"}

])

#create index for Host collection

db.Host.createIndex({'host\_name':1})

db.Host.createIndex({'host\_id':1})

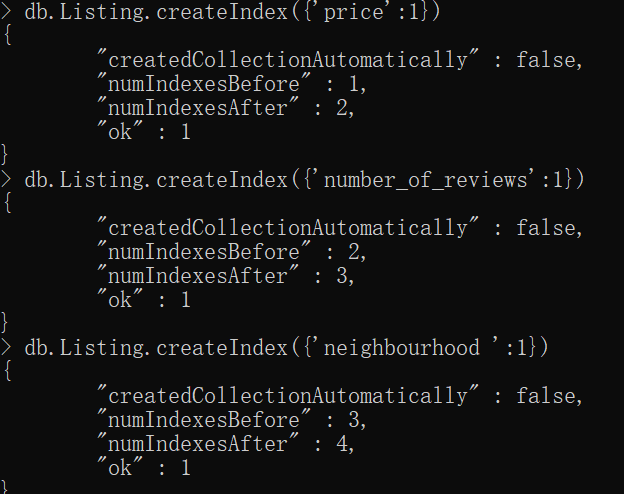


#create index for Listing collection

db.Listing.createIndex({'price':1})

db.Listing.createIndex({'number\_of\_reviews':1})

db.Listing.createIndex({'neighbourhood ':1})

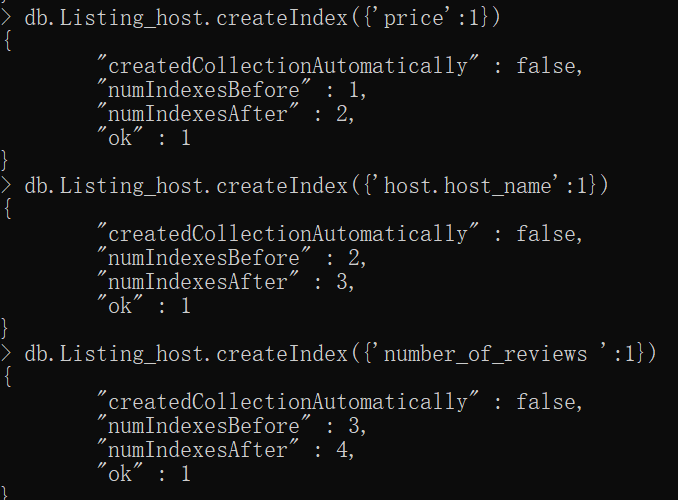


#create index for Listing\_host collection

db.Listing\_host.createIndex({'price':1})

db.Listing\_host.createIndex({'host.host\_name':1})

db.Listing\_host.createIndex({'number\_of\_reviews ':1})



**1.**

**Referencing code:**

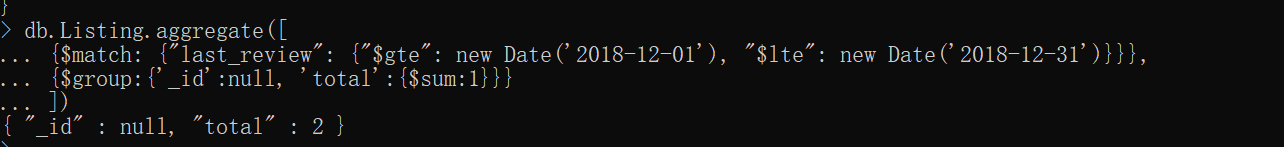
db.Listing.aggregate([

{$match: {"last\_review": {"$gte": new Date('2018-12-01'), "$lte":

new Date('2018-12-31')}}},

{$group:{'\_id':null, 'total':{$sum:1}}}

])



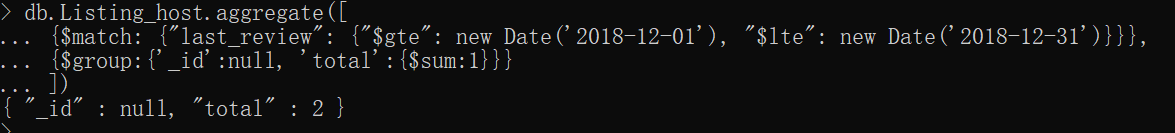
**Embedding code:**

db.Listing\_host.aggregate([

{$match: {"last\_review": {"$gte": new Date('2018-12-01'), "$lte": new Date('2018-12-31')}}},

{$group:{'\_id':null, 'total':{$sum:1}}}

])



**2.**

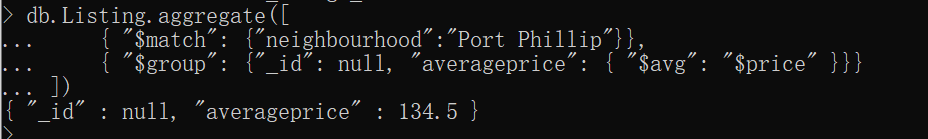
**Referencing code:**

db.Listing.aggregate([

{ "$match": {"neighbourhood":"Port Phillip"}},

{ "$group": {"\_id": null, "averageprice": { "$avg": "$price" }}}

])



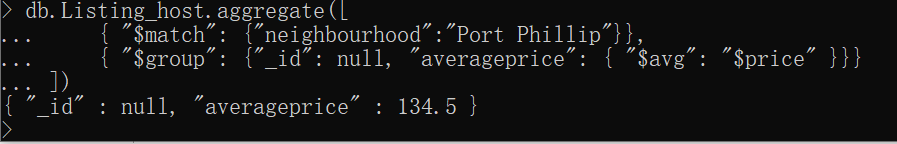
**Embedding code:**

db.Listing\_host.aggregate([

{ "$match": {"neighbourhood":"Port Phillip"}},

{ "$group": {"\_id": null, "averageprice": { "$avg": "$price" }}}

])



**3.**

**Referencing code:**

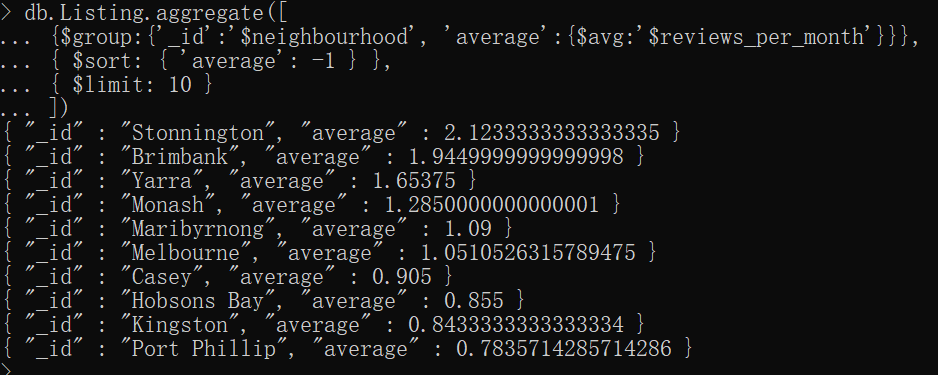
db.Listing.aggregate([

{$group:{'\_id':'$neighbourhood', 'average':{$avg:'$reviews\_per\_month'}}},

{ $sort: { 'average': -1 } },

{ $limit: 10 }

])



**Embedding code:**

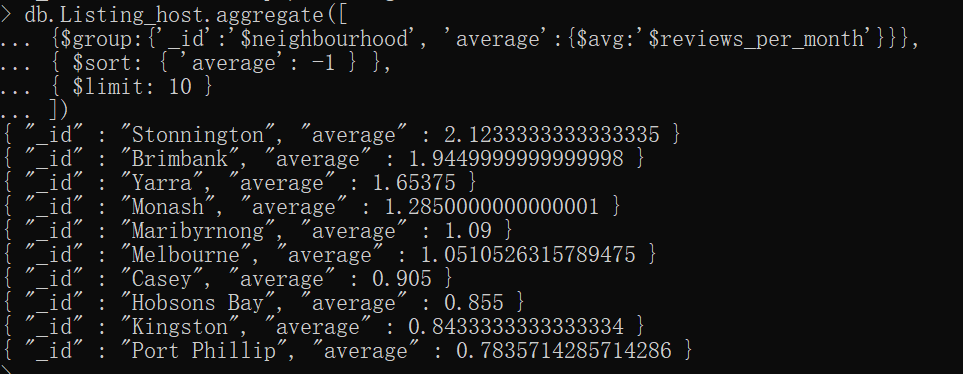
db.Listing\_host.aggregate([

{$group:{'\_id':'$neighbourhood', 'average':{$avg:'$reviews\_per\_month'}}},

{ $sort: { 'average': -1 } },

{ $limit: 10 }

])



**4.**

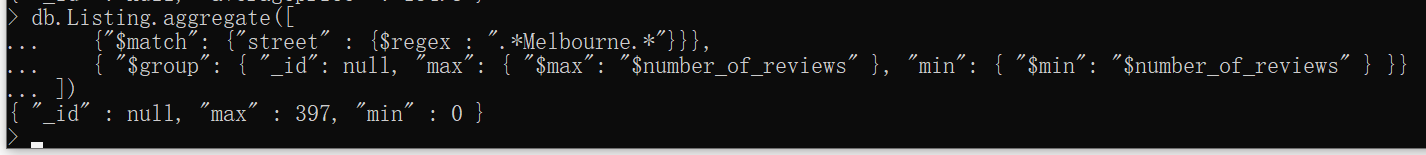
**Referencing code:**

db.Listing.aggregate([

{"$match": {"street" : {$regex : ".\*Melbourne.\*"}}},

{ "$group": { "\_id": null, "max": { "$max": "$number\_of\_reviews" }, "min": { "$min": "$number\_of\_reviews" } }}

])



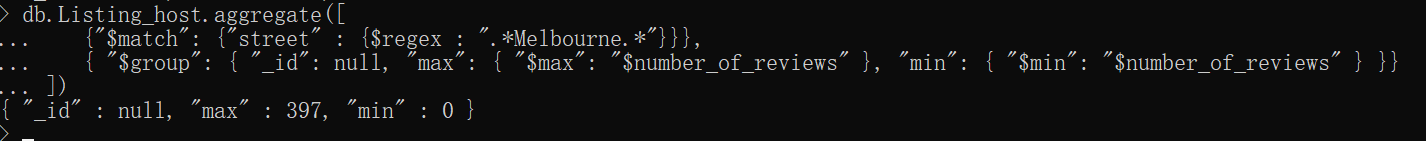
**Embedding code:**

db.Listing\_host.aggregate([

{"$match": {"street" : {$regex : ".\*Melbourne.\*"}}},

{ "$group": { "\_id": null, "max": { "$max": "$number\_of\_reviews" }, "min": { "$min": "$number\_of\_reviews" } }}

])



**5.**

**Referencing code:**

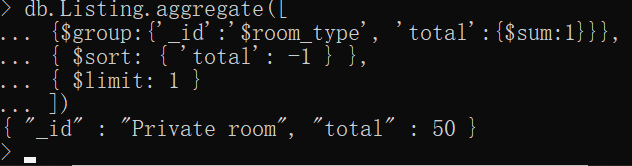
db.Listing.aggregate([

{$group:{'\_id':'$room\_type', 'total':{$sum:1}}},

{ $sort: { 'total': -1 } },

{ $limit: 1 }

])



**Embedding code:**

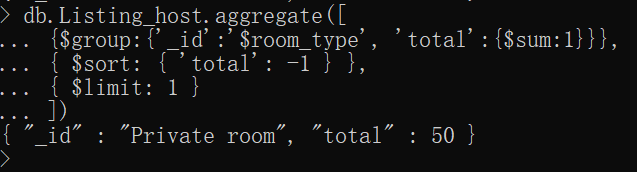
db.Listing\_host.aggregate([

{$group:{'\_id':'$room\_type', 'total':{$sum:1}}},

{ $sort: { 'total': -1 } },

{ $limit: 1 }

])



**6.**

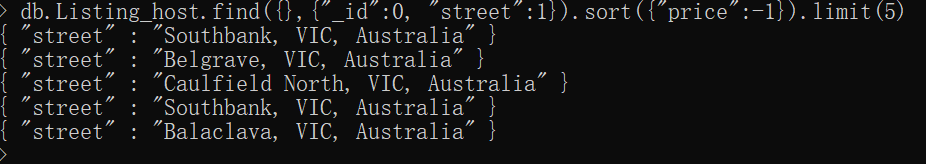
**Referencing code:**

db.Listing.find({},{"\_id":0,"street":1}).sort({"price":-1}).limit(5)



**Embedding code:**

db.Listing\_host.find({},{"\_id":0, "street":1}).sort({"price":-1}).limit(5)



**7.**

**Referencing code:**

db.Listing.aggregate([

{$lookup:

{from:'Host',

localField:'host\_id',

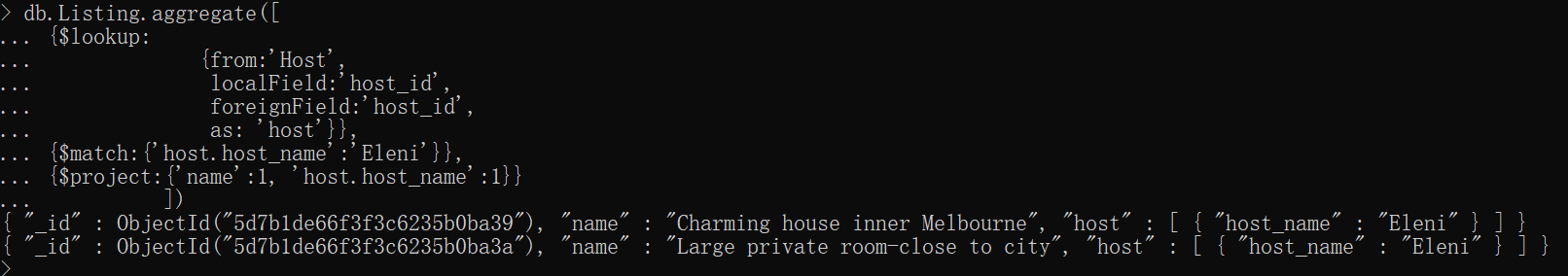
foreignField:'host\_id',

as: 'host'}},

{$match:{'host.host\_name':'Eleni'}},

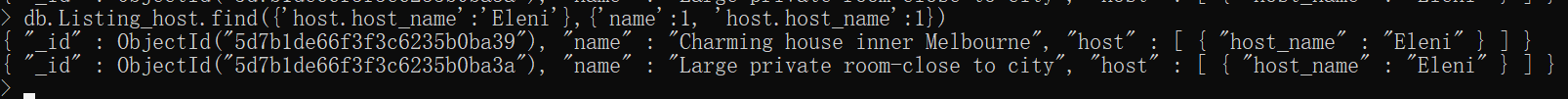
{$project:{'name':1, 'host.host\_name':1}}

])



**Embedding code:**

db.Listing\_host.find({'host.host\_name':'Eleni'},{'name':1, 'host.host\_name':1})



**8.**

**Referencing code:**

db.Listing.aggregate([

{"$match":{"room\_type":"Entire home/apt"}},

{$lookup:{from:'Host',localField:'host\_id',foreignField:'host\_id',as:'result'}},

{"$match":{"result.host\_response\_time":"within a few hours"}},

{$project:{\_id:0,name:1,'room\_type':1,'result.host\_response\_time':1}}

])

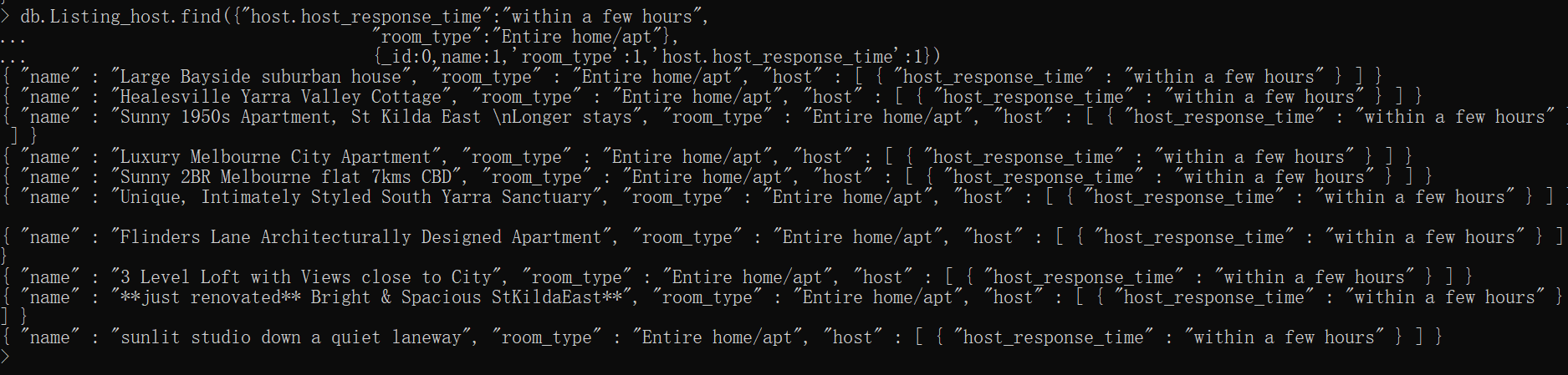


**Embedding code:**

db.Listing\_host.find({"host.host\_response\_time":"within a few hours",

"room\_type":"Entire home/apt"},

{\_id:0,name:1,'room\_type':1,'host.host\_response\_time':1})



**9.**

**Referencing code:**

db.Listing.aggregate([

{$lookup:

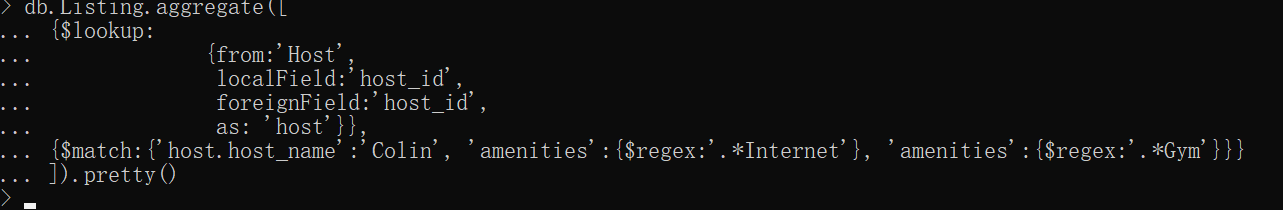
{from:'Host',

localField:'host\_id',

foreignField:'host\_id',

as: 'host'}},

{$match:{'host.host\_name':'Colin', 'amenities':{$regex:'.\*Internet'}, 'amenities':{$regex:'.\*Gym'}}}]).pretty()



**Embedding code:**

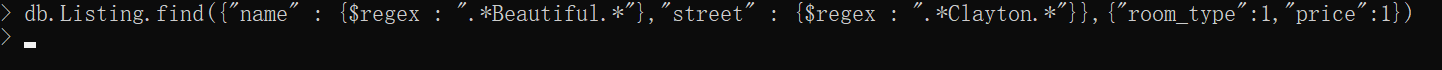
db.Listing\_host.find({'host.host\_name':'Colin', 'amenities':{$regex:'.\*Internet'}, 'amenities':{$regex:'.\*Gym'}}).pretty()



**10.**

**Referencing code:**

db.Listing.find({"name" : {$regex : ".\*Beautiful.\*"},"street" : {$regex : ".\*Clayton.\*"}},{"room\_type":1,"price":1})



**Embedding code:**

db.Listing\_host.find({"name" : {$regex : ".\*Beautiful.\*"},"street" : {$regex : ".\*Clayton.\*"}},{"room\_type":1,"price":1})



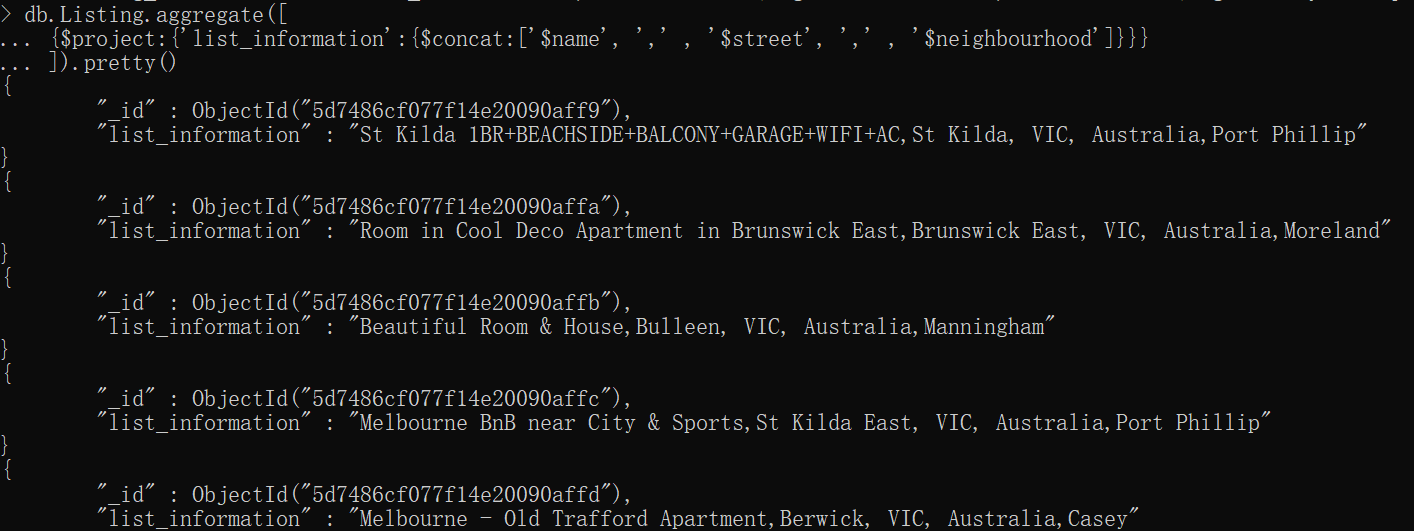
**11.**

**Referencing code:**

db.Listing.aggregate([

{$project:{'list\_information':{$concat:['$name', ',' , '$street', ',' , '$neighbourhood']}}}

]).pretty()

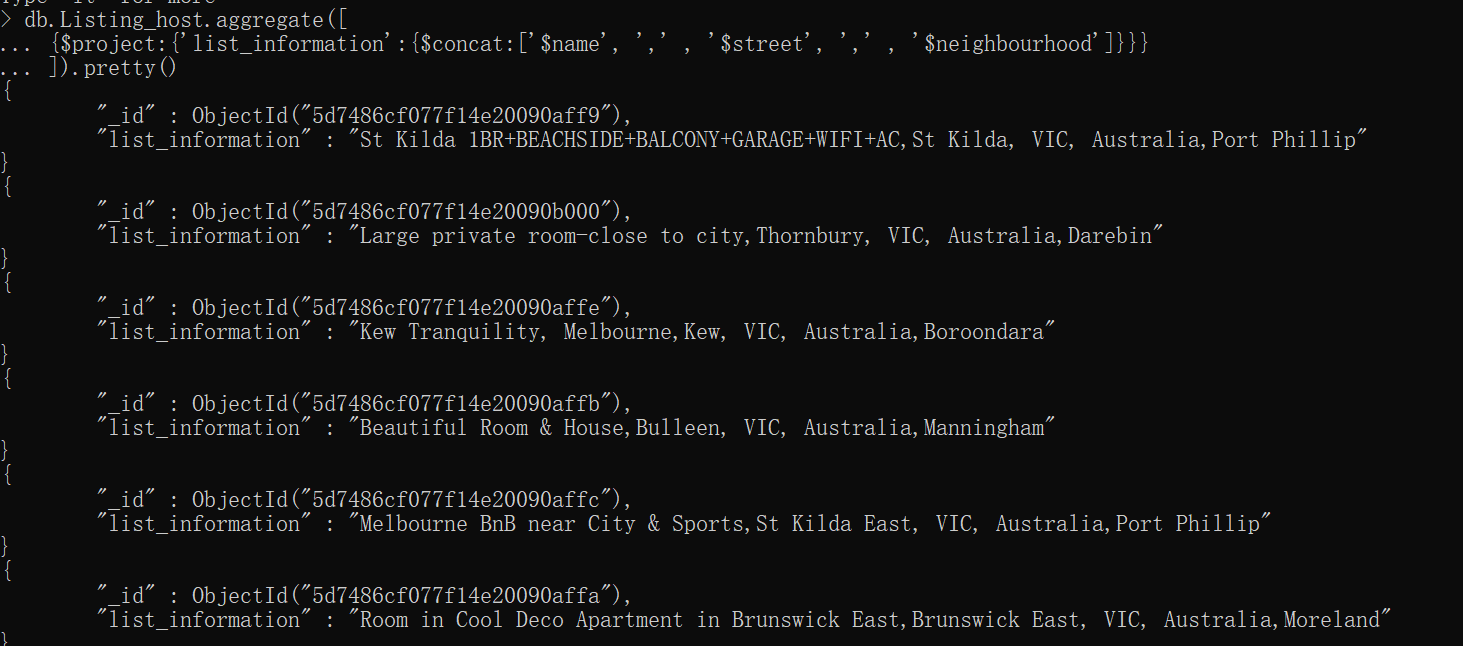


**Embedding code:**

db.Listing\_host.aggregate([

{$project:{'list\_information':{$concat:['$name', ',' , '$street', ',' , '$neighbourhood']}}}

]).pretty()



**12.**

**Referencing code:**

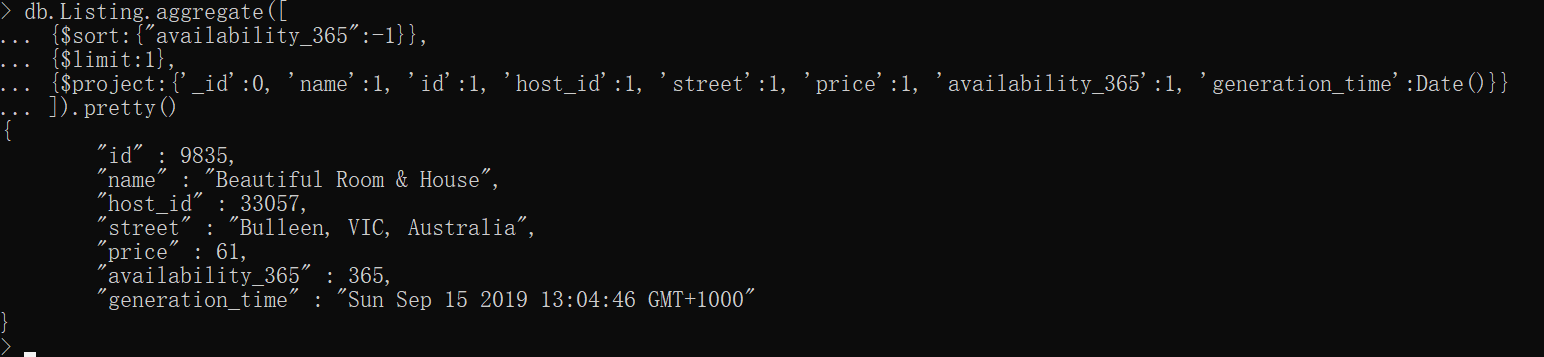
db.Listing.aggregate([

{$sort:{"availability\_365":-1}},

{$limit:1},

{$project:{'\_id':0, 'name':1, 'id':1, 'host\_id':1, 'street':1, 'price':1, 'availability\_365':1, 'generation\_time':Date()}}

]).pretty()



**Embedding code:**

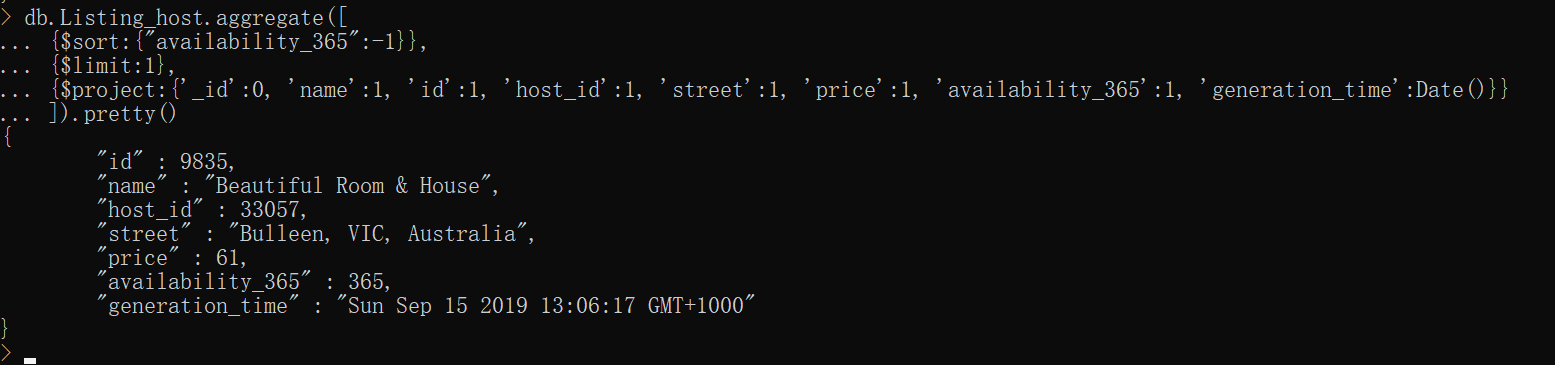
db.Listing\_host.aggregate([

{$sort:{"availability\_365":-1}},

{$limit:1},

{$project:{'\_id':0, 'name':1, 'id':1, 'host\_id':1, 'street':1, 'price':1, 'availability\_365':1, 'generation\_time':Date()}}

]).pretty()



**13.**

**Referencing code:**

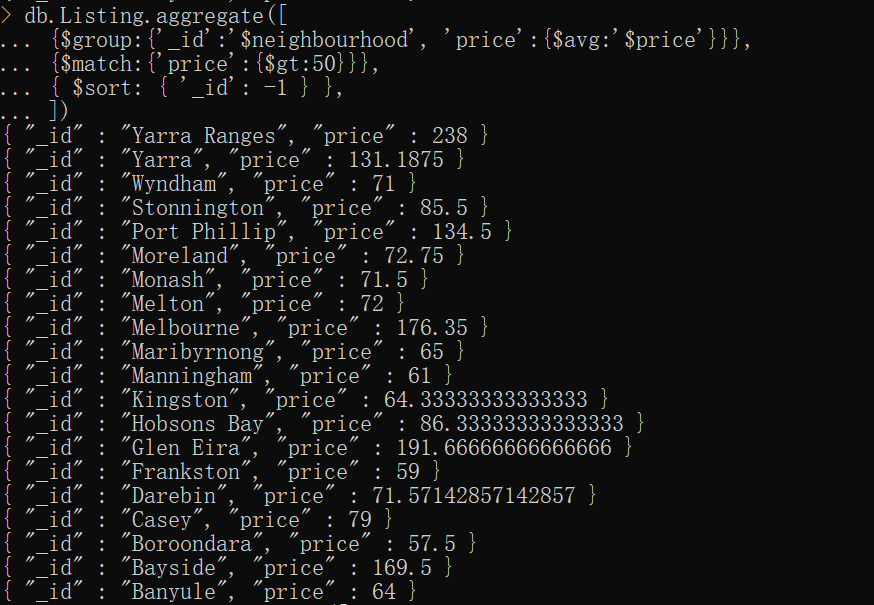
db.Listing.aggregate([

{$group:{'\_id':'$neighbourhood', 'price':{$avg:'$price'}}},

{$match:{'price':{$gt:50}}},

{ $sort: { '\_id': -1 } },

])



**Embedding code:**

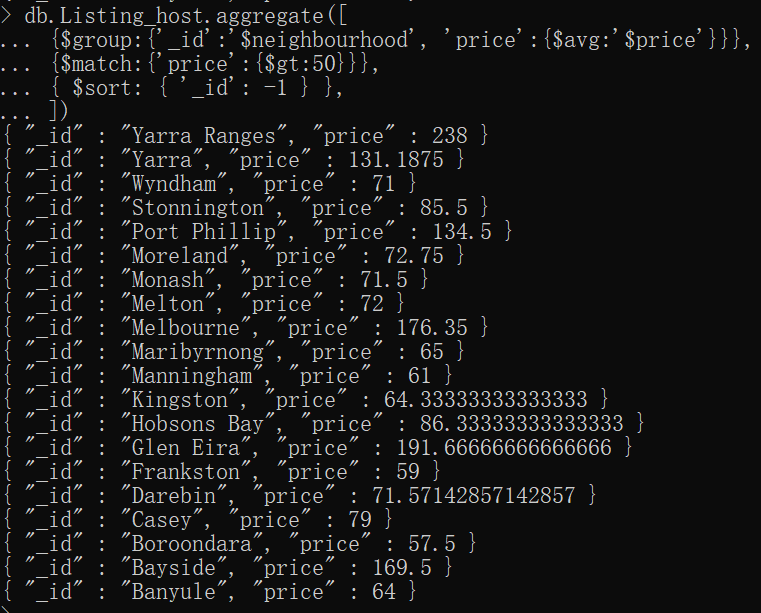
db.Listing\_host.aggregate([

{$group:{'\_id':'$neighbourhood', 'price':{$avg:'$price'}}},

{$match:{'price':{$gt:50}}},

{ $sort: { '\_id': -1 } },

])



**14.**

**Referencing code:**

db.Host.aggregate([

{$project: {\_id:0,host\_id: 1,host\_name:1,

number\_of\_verification\_methods: { $cond: { if: { $isArray: "$host\_verifications" }, then:

{ $size: "$host\_verifications" }, else: "NA"} }}},

{$sort:{ number\_of\_verification\_methods:-1}}

] )



**Embedding code:**

db.Listing\_host.aggregate([

{$group:{'\_id':{'host\_id':'$host.host\_id','host\_name':'$host.host\_name',

'num':'$host.host\_verifications' }}},

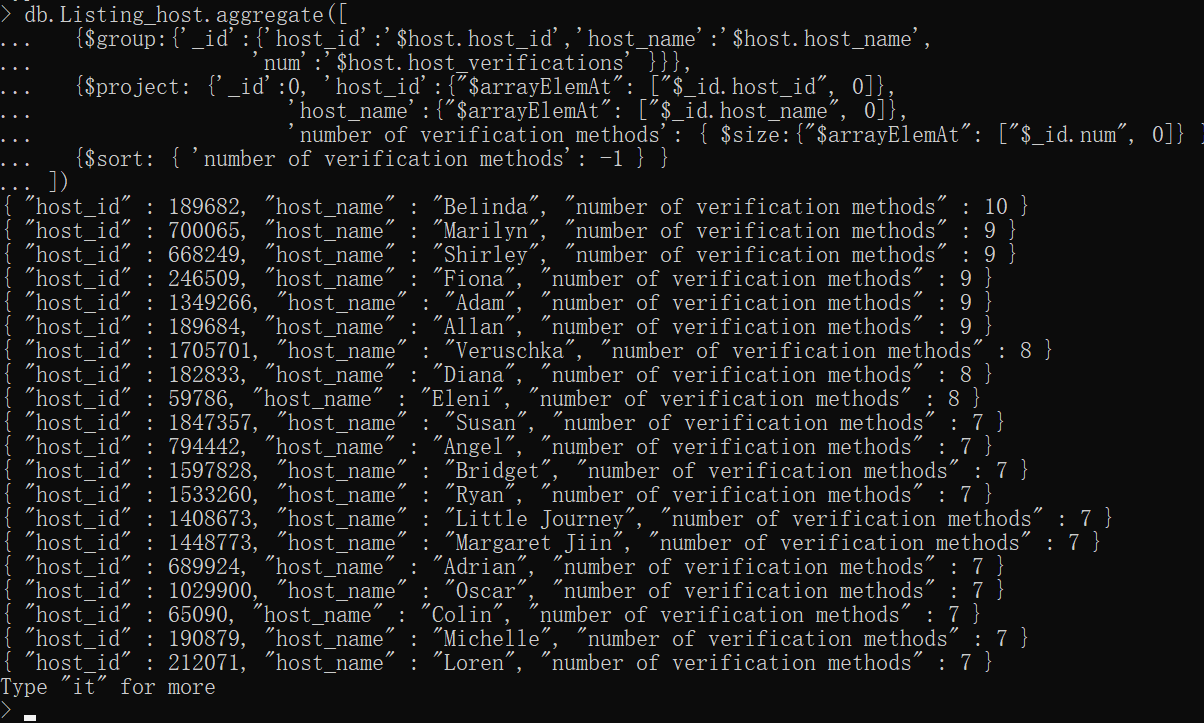
{$project: {'\_id':0, 'host\_id':{"$arrayElemAt": ["$\_id.host\_id", 0]},

'host\_name':{"$arrayElemAt": ["$\_id.host\_name", 0]},

'number of verification methods': { $size:{"$arrayElemAt": ["$\_id.num", 0]} }}},

{$sort: { 'number of verification methods': -1 } }

])



**Cassandra:**

#preparation

#create the keyspace

CREATE KEYSPACE FIT5137\_Assign\_C3 with

replication = {'class':'SimpleStrategy', 'replication\_factor':1};

#swich to keyspace

USE FIT5137\_Assign\_C3;

#create table in keyspace

CREATE TABLE review (listing\_id text,

review\_id text,

review\_date date,

reviewer\_id text,

reviewer\_name text,

review\_scores int,

comments text,

PRIMARY KEY (listing\_id, review\_date));

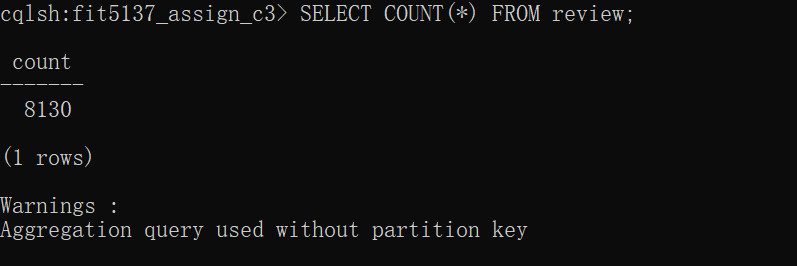
#insert data into table

COPY review (listing\_id, review\_id, review\_date, reviewer\_id, reviewer\_name, review\_scores, comments)

FROM 'D:\review.csv' WITH HEADER = true AND ESCAPE = '"' ;

#check the number of rows is equal to 8208 or not

SELECT COUNT(\*) FROM review;



#create second index

CREATE INDEX ON review (review\_date);

**15.**

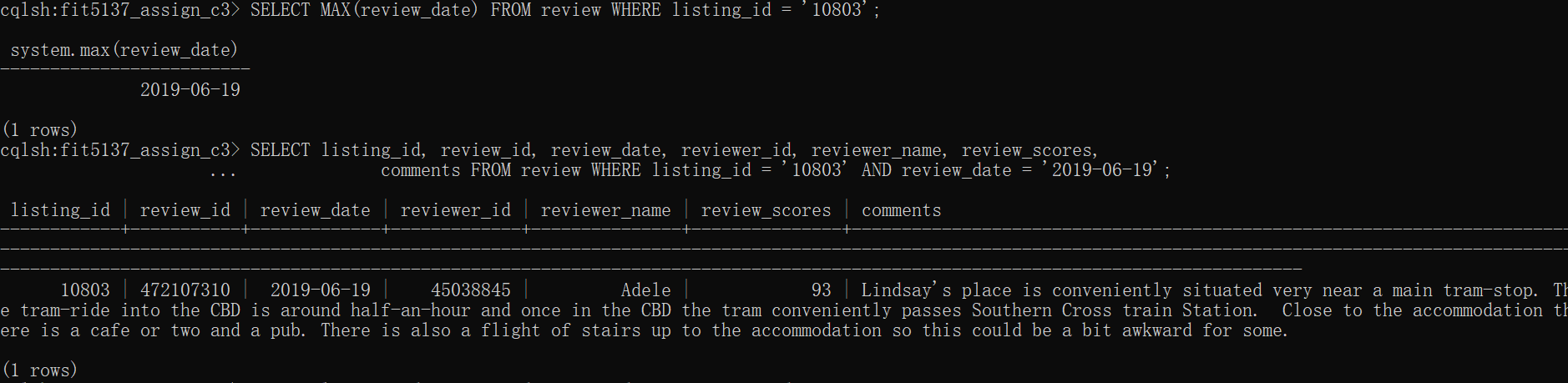
Since we need to displace the whole review of the most recent one, the max function only returning the most recent date, the information of other may not be same row. If we use the order by review\_date, it could work. The problem is that question 19 also need order by score which will be conflict with this question. So this question will be done by two queries.

#find the recent date first

SELECT MAX(review\_date) FROM review WHERE listing\_id = '10803';

#base on the listing id and recent date find the recent review

SELECT listing\_id, review\_id, review\_date, reviewer\_id, reviewer\_name, review\_scores,

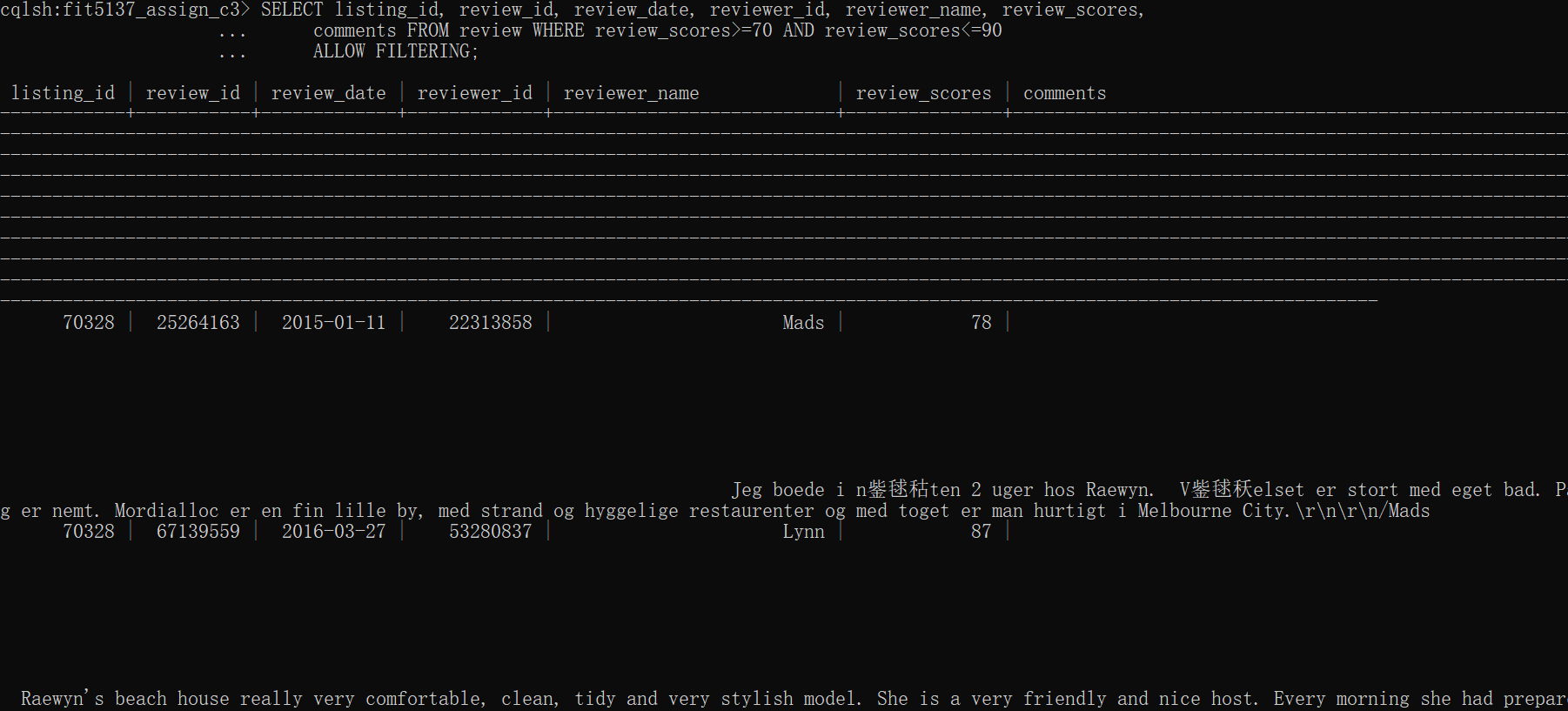
comments FROM review WHERE listing\_id = '10803' AND review\_date = '2019-06-19';

**16.**

SELECT listing\_id, review\_id, review\_date, reviewer\_id, reviewer\_name, review\_scores,

comments FROM review WHERE review\_scores>=70 AND review\_scores<=90

ALLOW FILTERING;



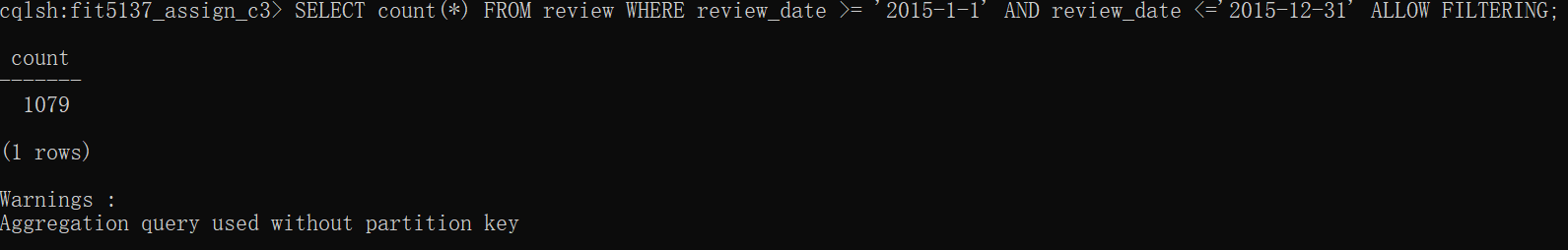
**17.**

SELECT listing\_id, reviewer\_name, comments FROM review WHERE review\_scores < 50 ALLOW FILTERING;



**18.**

SELECT count(\*) FROM review WHERE review\_date >= '2015-1-1' AND review\_date <='2015-12-31' ALLOW FILTERING;



**19.**

**The reason of using 2 query is same as question 15.**

#find the highest scores in that day

SELECT MAX(review\_scores) FROM review WHERE review\_date = '2017-03-26';

#base on the highest scores and date to dispace the review

SELECT listing\_id, review\_id, review\_date, reviewer\_id, reviewer\_name, review\_scores,

comments FROM review WHERE review\_scores = 100

AND review\_date = '2017-03-26' ALLOW FILTERING; 

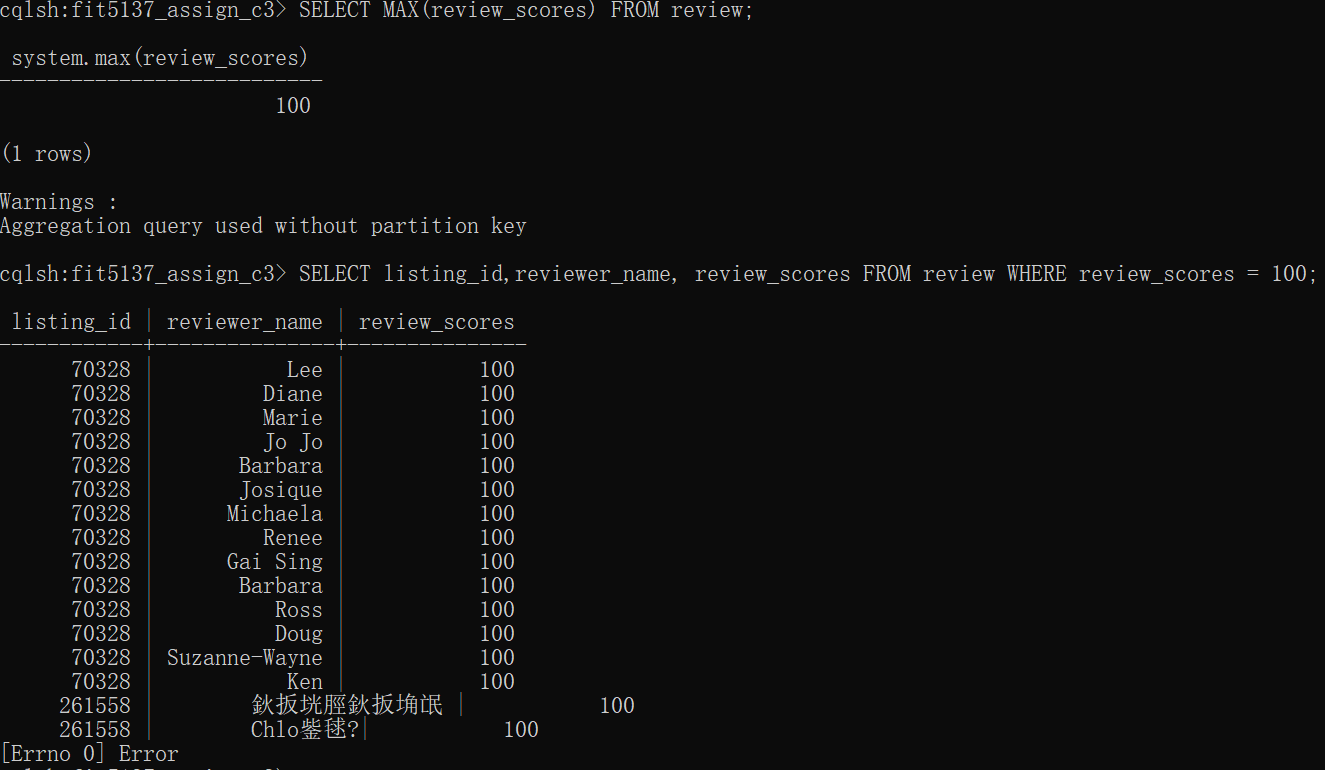
**20.**

#find the highest scores in table

SELECT MAX(review\_scores) FROM review;

#dispace the review with the highest review scores

SELECT listing\_id,reviewer\_name, review\_scores FROM review WHERE review\_scores = 100;



**Additional 5 questions:**

**MongoDB**

**Additional Q1.** Number of listing, average price, max price, min price in each city, and sort by the number of listing order.

**Referencing code:**

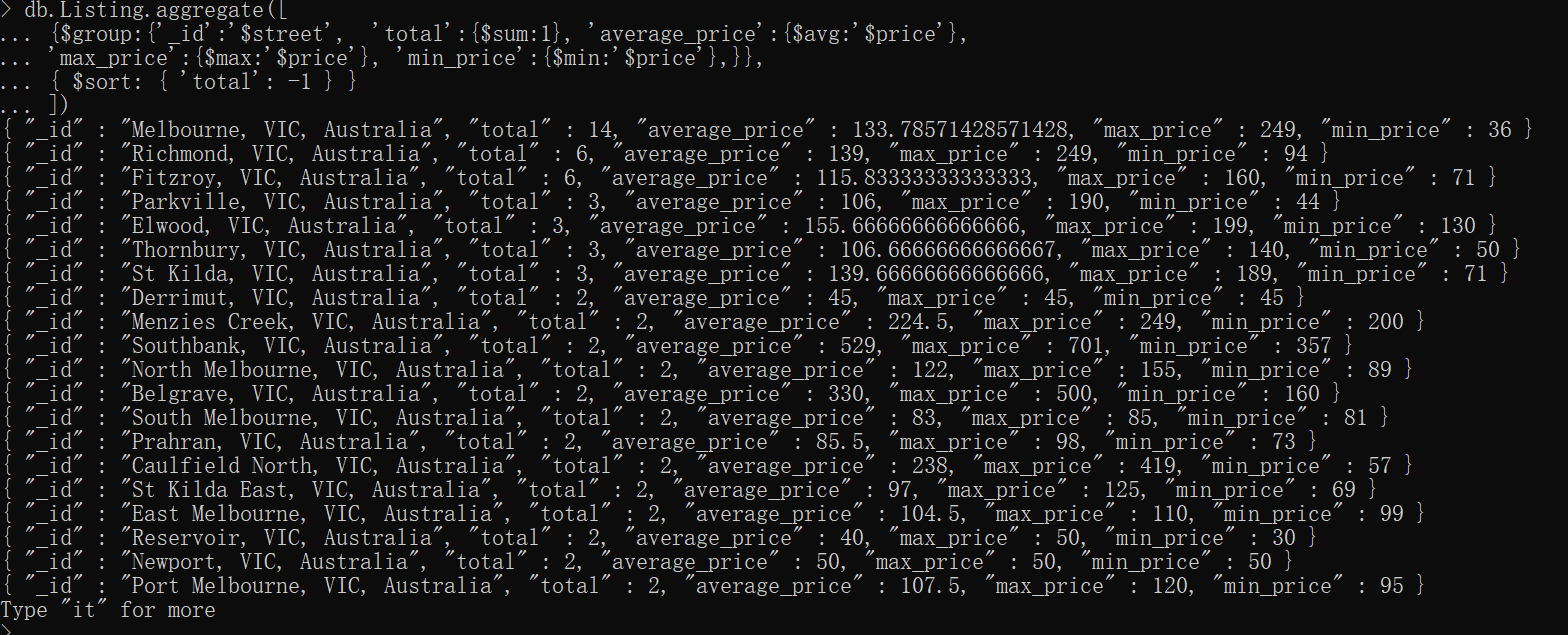
db.Listing.aggregate([

{$group:{'\_id':'$street', 'total':{$sum:1}, 'average\_price':{$avg:'$price'},

'max\_price':{$max:'$price'}, 'min\_price':{$min:'$price'},}},

{ $sort: { 'total': -1 } }

])



**Embedding code:**

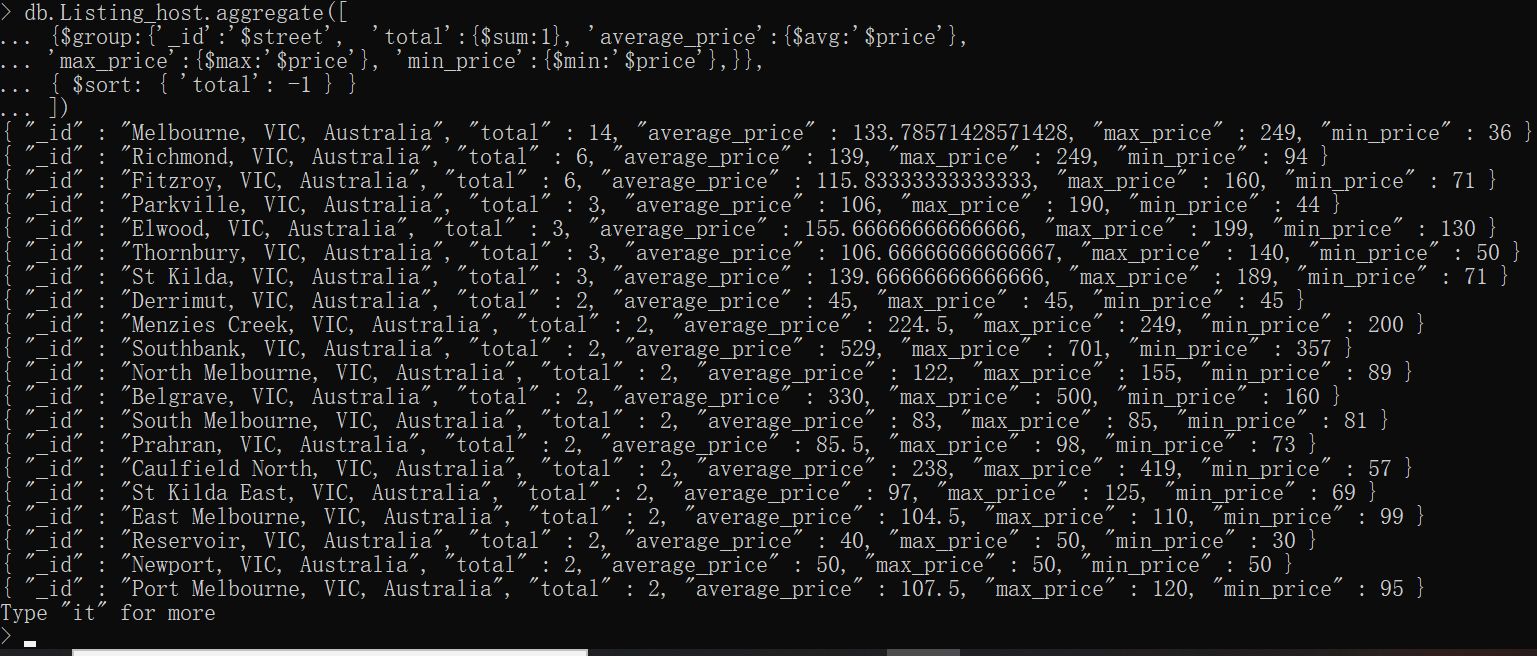
db.Listing\_host.aggregate([

{$group:{'\_id':'$street', 'total':{$sum:1}, 'average\_price':{$avg:'$price'},

'max\_price':{$max:'$price'}, 'min\_price':{$min:'$price'},}},

{ $sort: { 'total': -1 } }

])



**Additional Q2.** Count the number of amernities in each listings.

**Referencing code:**

db.Listing.aggregate([

{ $project : { id:'$id', name:'$name', amernites : { $split: ["$amenities", ","] }, qty : 1 } },

{ $unwind : "$amernites" },

{ $group : { \_id: { "listing\_id" : "$id", 'name':'$name' }, amernities\_num: { "$sum" : 1 } } },

{ $sort : { listing\_id : -1 } }

])



**Embedding code:**

db.Listing\_host.aggregate([

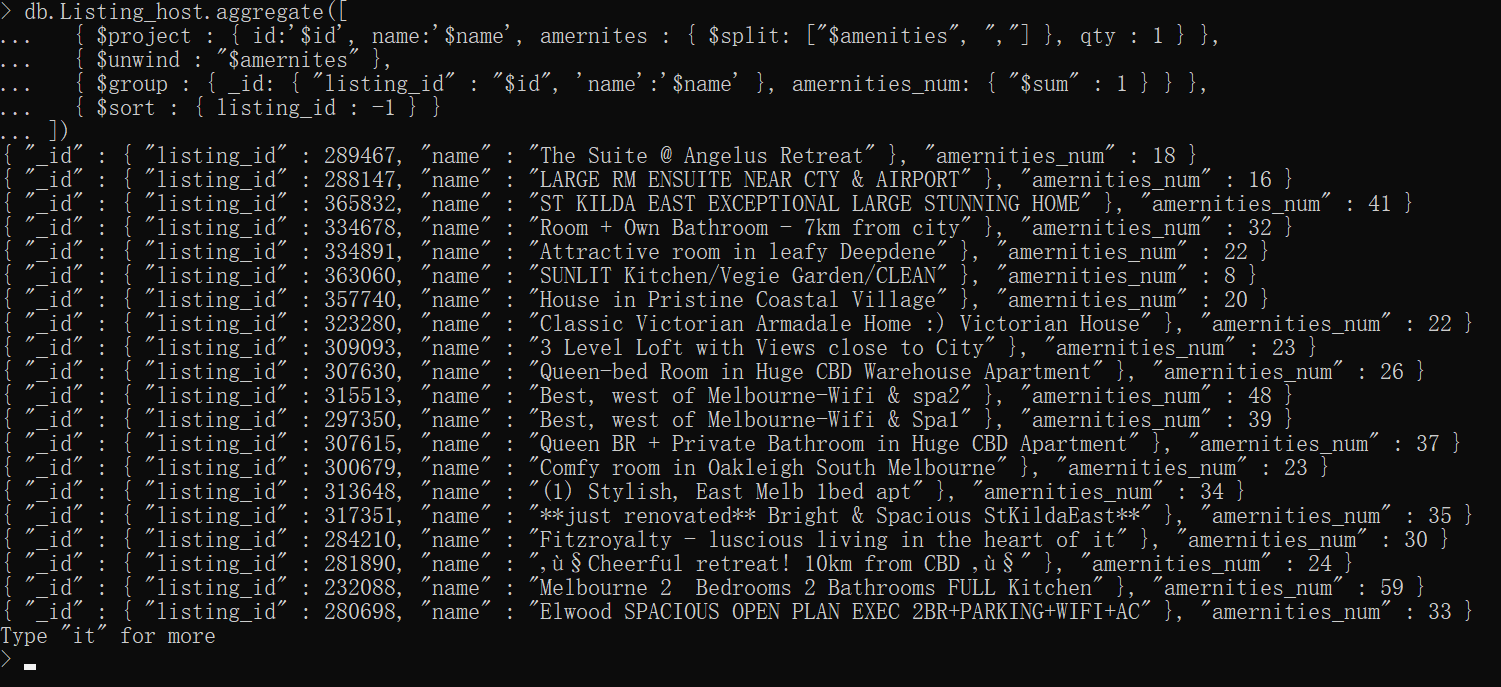
{ $project : { id:'$id', name:'$name', amernites : { $split: ["$amenities", ","] }, qty : 1 } },

{ $unwind : "$amernites" },

{ $group : { \_id: { "listing\_id" : "$id", 'name':'$name' }, amernities\_num: { "$sum" : 1 } } },

{ $sort : { listing\_id : -1 } }

])



**Additional Q3.** Count how many host which room\_type is "Private room" and the host\_verification allows email.

**Referencing code:**

db.Listing.aggregate([

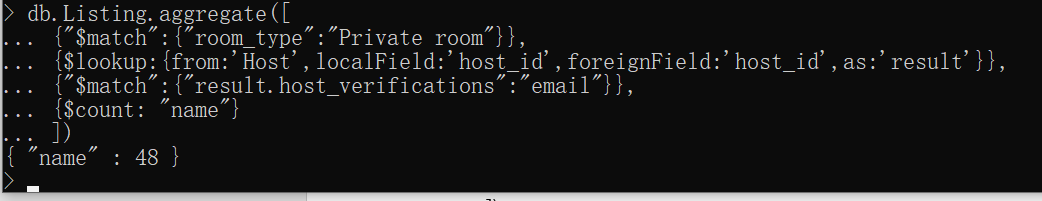
{"$match":{"room\_type":"Private room"}},

{$lookup:{from:'Host',localField:'host\_id',foreignField:'host\_id',as:'result'}},

{"$match":{"result.host\_verifications":"email"}},

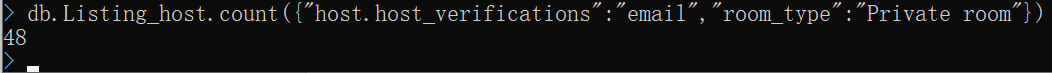
{$count: "name"}

])



**Embedding code:**

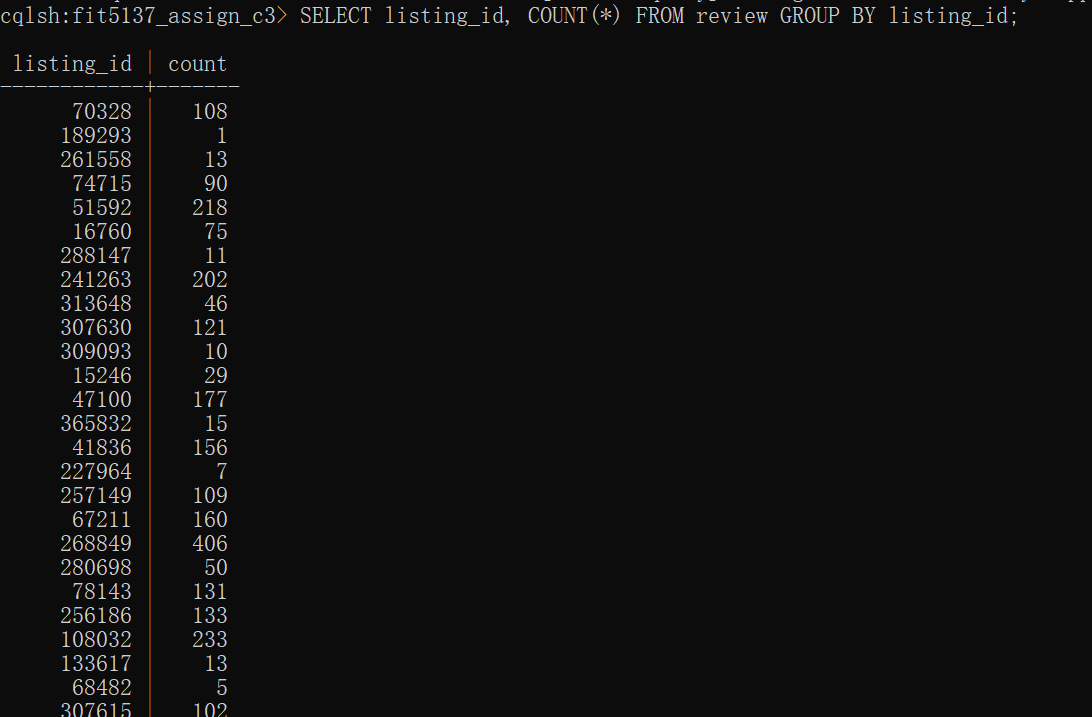
db.Listing\_host.count({"host.host\_verifications":"email","room\_type":"Private room"})



**Cassandra**

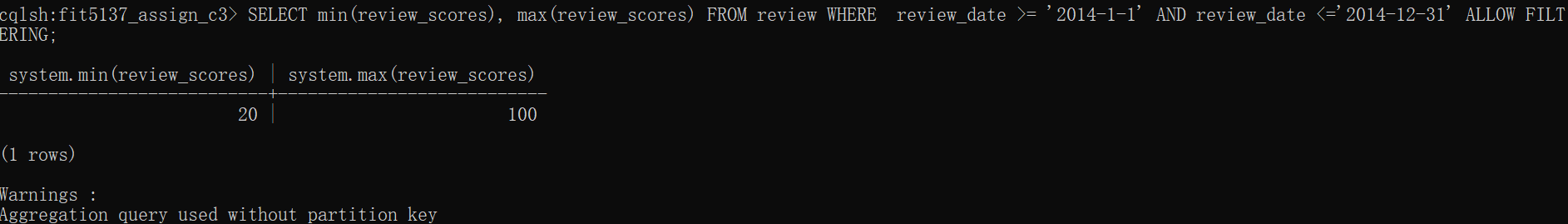
**Additional Q4.** What is the number of review of each listing?

SELECT listing\_id, COUNT(\*) FROM review GROUP BY listing\_id;



**Additional Q5**. Display the highest rating and lowest rating left in 2014.

SELECT min(review\_scores), max(review\_scores) FROM review WHERE review\_date >= '2014-1-1' AND review\_date <='2014-12-31' ALLOW FILTERING;



**C.4 Database Comparison**

Through the practice the demo of two database, we have basic idea about the features of two types of Non-SQL database. Following two tables are the advantages and disadvantages of MongoDB and Cassandra:

**MongoDB**

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| **1.** **Very friendly for querying.**  MongoDB can handle different kinds  of query without changing collection  a lot. | **1. Don’t have proper join function.**  MongoDB only have left outer join. |
| **2.** **Data is easy to handle.** because the  smallest unit is document, did not  have a lot of limit in data type. |  |
| **3. MongoDB can scale out.** |  |

**Cassandra**

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| **1. The Cassandra is good at Scaling.**  data can be partitioning into different  data centers, racks and nodes. | **1. Data need to be predefined.** |
| **2. Cassandra have fault tolerance.**  The duplication feature ensures the  data will not lost, even one of node,  rack or data center is down. | **2. The system somehow is not very**  **stable.** |
| **3. Cassandra have good availability.**  The duplication ensures that there is  always the node available to query,  even the oriental data storage node is  down. | **3. A lot of restrictions on querying.**  - The delate, update and group by can  only using first partitioning key,  even the second index will not work.  - For order by, we must specify the  first partitioning key in restriction  condition.  - Cassandra did not support sub query  or nested query.  - If there is null value in certain  column, that column cannot be  second index or partitioning key. For  query, we have to add ‘ALLOW  FILTERING’, it causes the query  inefficient. |

Comparing the strength and weakness of MongoDB and Cassandra, and the situation of MonashBnB also have been considering, the MongoDB is more suitable for MonashBnB. There are two reasons. The first one is about query, since the MonashBnB is a start-up website, the query function is not very stable, we need to try all different queries to see which queries are more useful for staff and consumer. The Cassandra is not very suitable for this condition, its query is mainly depending on table structure, the different queries may need to changing data table a lot. In the other hand, the MongoDB is much more flexible, the query did not have a lot of restrictions. The second reason is about the size of website, the MonashBnB is start-up website. The data volume of website is not very big, we don’t need to estimate the other datacenter, so the MongoDB should be enough to handling current data size. In the future, Cassandra may can be considered as database for this website when the data volume is very big.

Following is the merging step for merging review table from Cassandra into MongoDB:

* Export the data from Cassandra into csv file by using copy + out function.
* Using pandas in Python to load csv file to wrangling data.
* Define a function to convert the data into library type data which looks like json format, then using json package in Python completely convert the data into json format, export the json format data into json file.
* Insert json file into new mongodb collection, then the merging step is done.