ADM Final Project

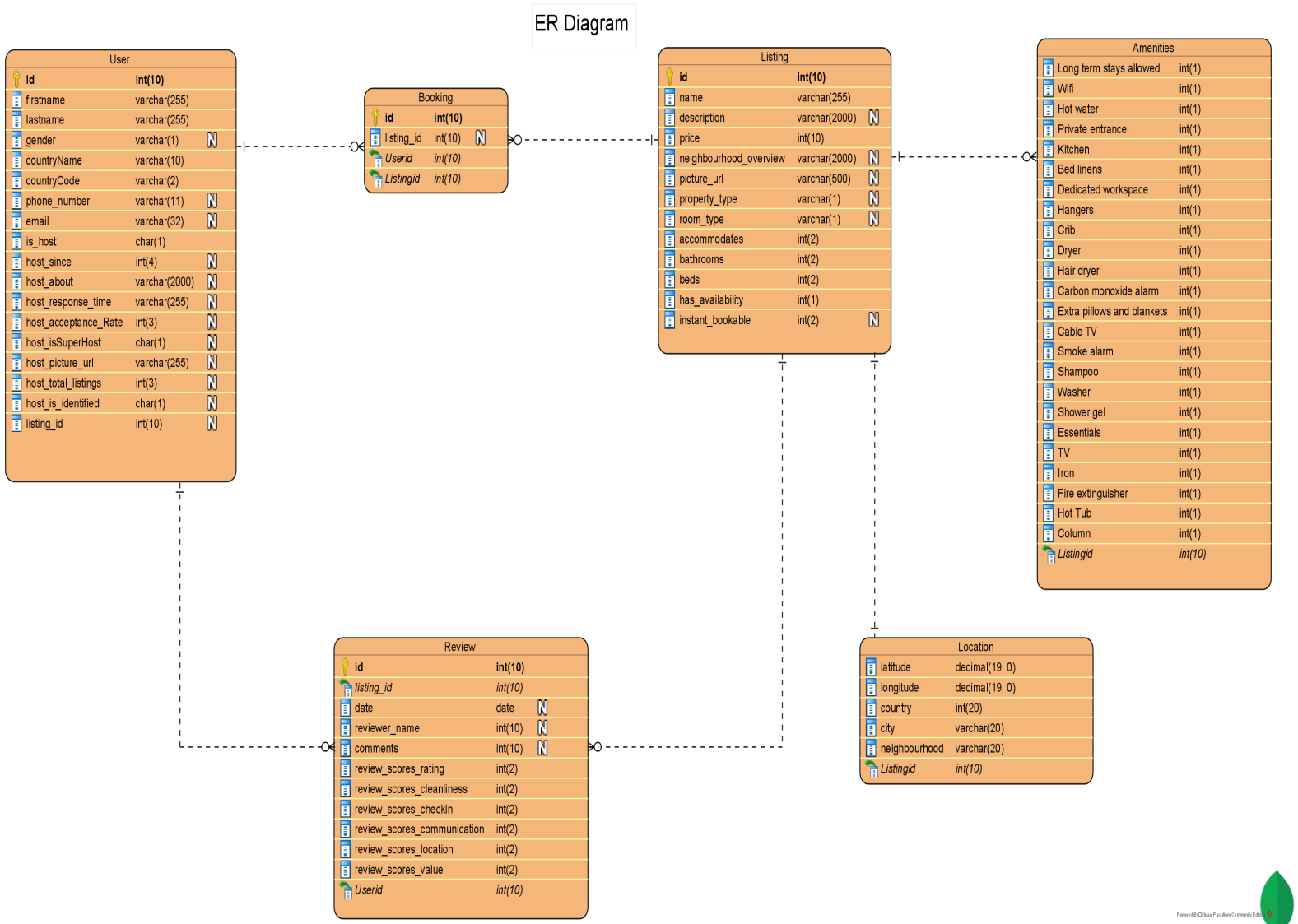
Sahitya Reddy Bollavaram 4849759
Sanchayan Bhunia 4849650

About the Dataset

- Airbnb dataset of Amsterdam area.
- Contents include Properties' information, Location, Amenities Available in the property, Reviews.
- User data is important, and since it's not available in the dataset, it is generated synthetically using a java library https://faker.readthedocs.io/ based on the host_id and reviewer_id in the dataset.
- The relations between the data is built using Pandas Dataframe in Python.
- Data size : 300 MB

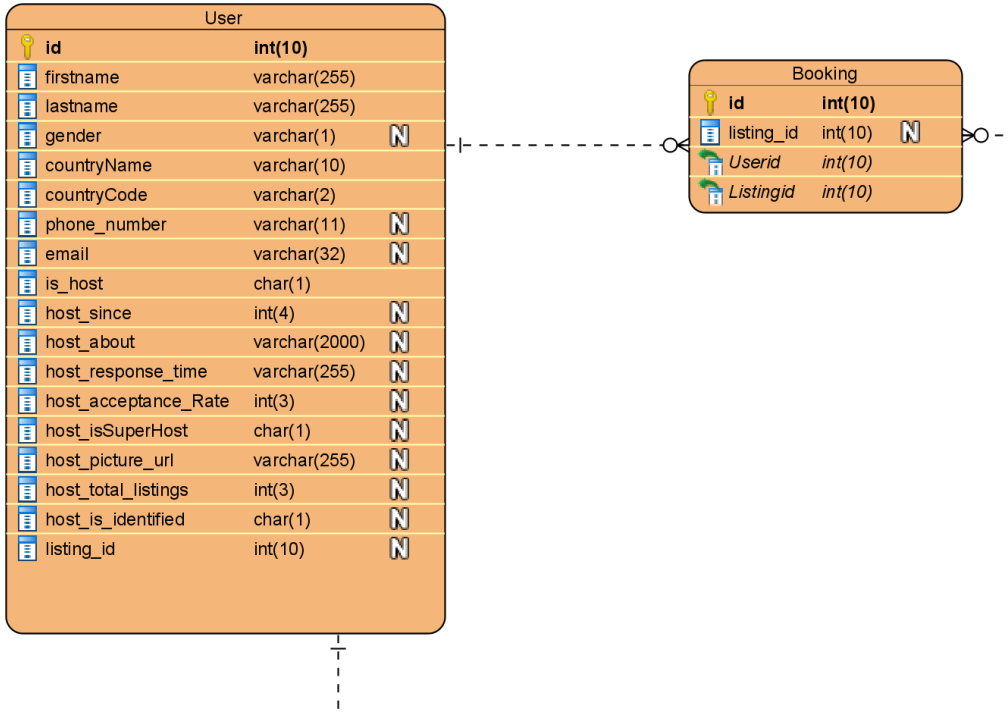
Conceptual Schema

Entities	Relation
User -> Bookings	One to Many
User -> Listings	One to Many
Listings -> Location	One to One
Listings -> Amenities	One to Many
Listings -> Reviews	One to Many



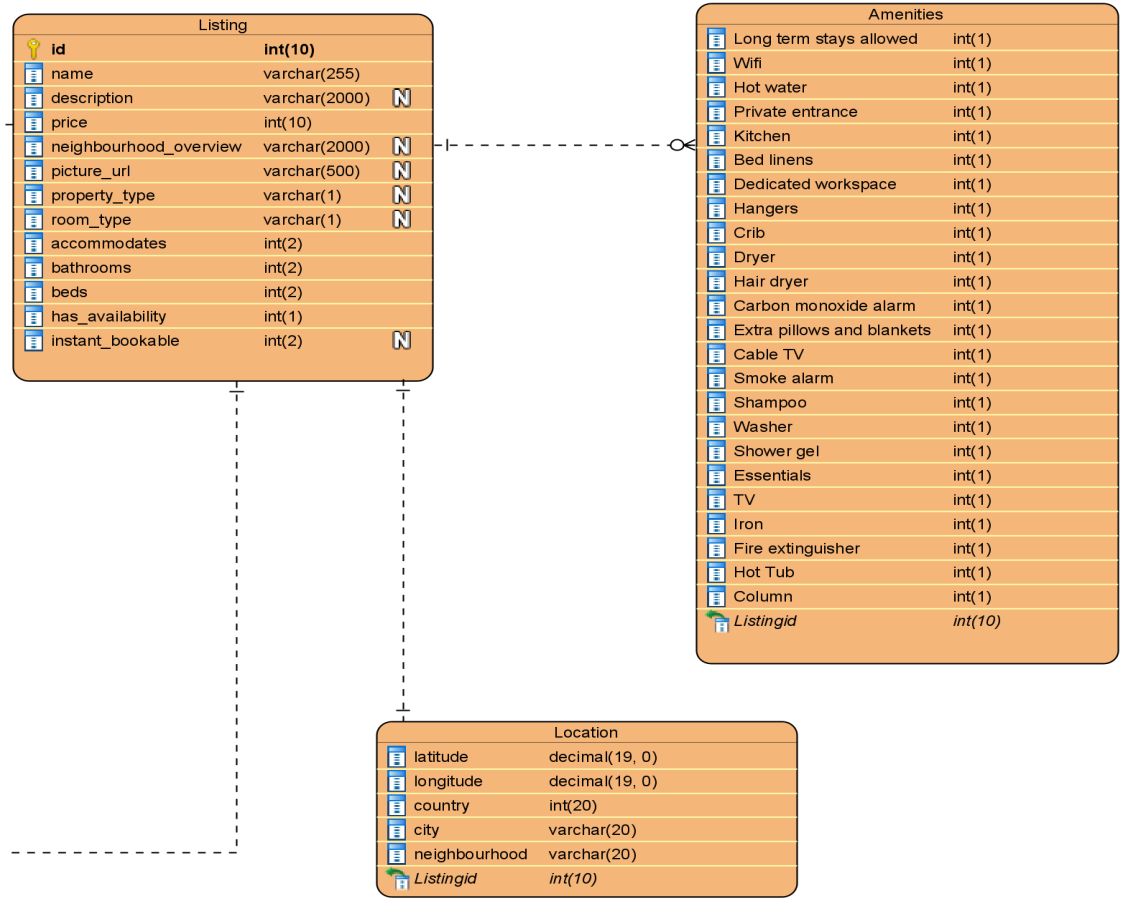
Analyzing workload – Aggregate modelling

1. Given a User ID, get the user's information. (user's first name, last name, email address, phone number, gender, country, if he is a host(is_host), host since, about him, his response time, picture url, identity verified status)
2. Given a User ID,get the booking history.(Booking ID, Listing ID(property ID), Total Price, No of Nights, Booking Date)
3. User makes a booking.
4. Get the number of bookings of each user.



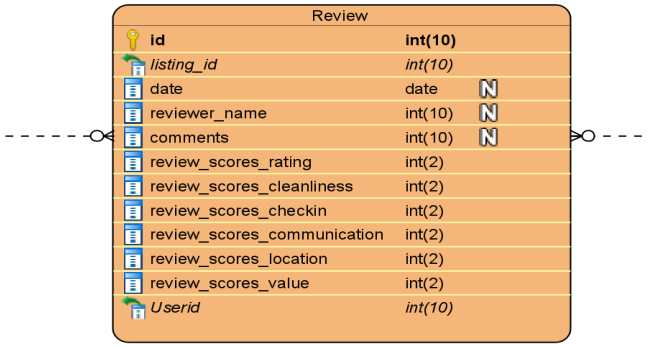
Analysing workload – Aggregate modelling

1. Given a User ID, get all of listings.
   Storing User's ID in Listing's aggregate would solve this problem.
2. Get all the Listings' names, prices, picture_urls, based on neighbourhood area and the number of people the property accomodates.
3. To the above query, price filter is applied.
4. Given User's Coordinates, Get all the listings' name, price, picture_url , coordinates in 1km radius.
5. Get the number of properties for each property type.
6. Add a new listing and change the user's is_host property to true.



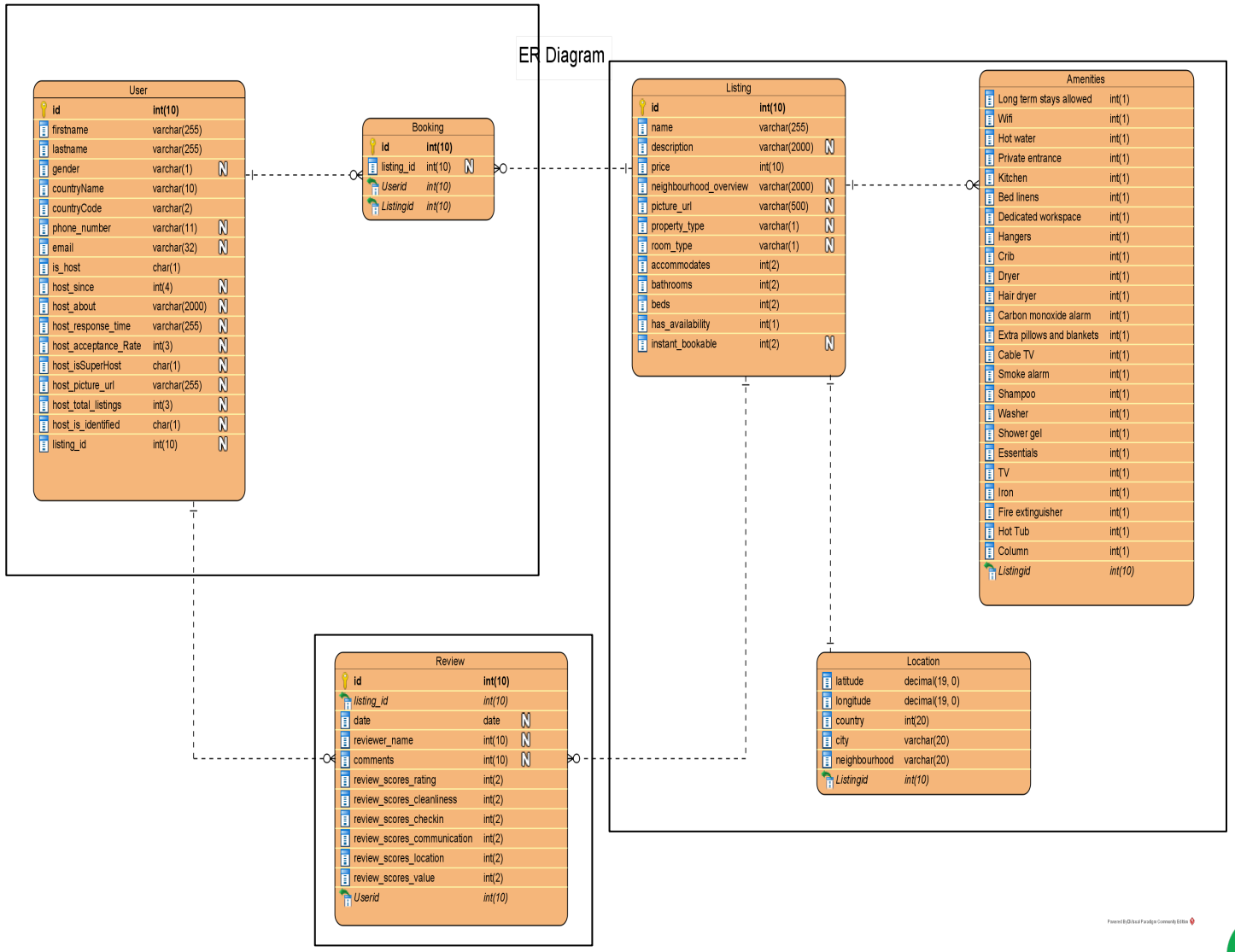
Analysing workload – Aggregate modelling

1. User writes a review about one listing (with all review_scores_rating,review_scores_cleanliness, review_scores_checkin, review_scores_communication, review_score_value, review_scores_location).
2. Get all reviews of a particular listings ID.
   Storing the associated listing ID would help this workload
3. Get the average rating of all kinds of ratings of each property based (cleanliness, check-in, communication, value, accuracy)
4. Order the average rating of neighbourhood area based on user score of the location.
   Storing the neighbourhood area's name would help with this workload.



Conceptual Schema

Entities	Relation	Decision
User -> Bookings	One to Few	To embed bookings is user data as an array of objects. **Embedded Document Pattern**
User -> Listings	One to Many	To store Listings as a separate collection by embedding the user entity's id attribute as user_id.
Listings -> Location	One to One	To embed location in listing's document.
Listings -> Amenities	One to Few	To embed the Amenities as an array of string values in listing's document.
Listings -> Reviews	One to Squillion	To store reviews separately in a new collection by embedding listing's id in each review. **Subset Pattern**



https://docs.mongodb.com/manual/tutorial/model-embedded-one-to-many-relationships-between-documents/#subset-pattern

Non-functional Requirements

1. The users should be able to always access the application (Availability)

- MongoDB automatically maintains replica sets.
- Multiple Copies of data.
- Primary Nodes and Secondary Nodes at any given time.
- If for any reason, hardware failure occurs, Primary Node is selected by voting procedure.
- Sharding – Partial Availability

```
sh.status(); (information under shards
property brings up the information about
replicas)

{  "_id" : "rs0",  "host" :
"rs0/it.unige.dibris.lsccluster.node3:27017,i
t.unige.dibris.lsccluster.node4:27017,it.unig
e.dibris.lsccluster.node5:27017",  "state" :
1 }
        {  "_id" : "rs1",  "host" :
"rs1/it.unige.dibris.lsccluster.node6:27017,i
t.unige.dibris.lsccluster.node7:27017",
"state" : 1 }
        {  "_id" : "rs2",  "host" :
"rs2/it.unige.dibris.lsccluster.node8:27017,i
t.unige.dibris.lsccluster.node9:27017",
"state" : 1 }

rs.conf() – for more detailed information.
```

Non-functional Requirements

1. The application should facilitate increased load.(Scalability)

- ❑ MongoDB supports horizontal scaling through Sharding, distributing data across several machines and facilitating high throughput operations with large sets of data.
- ❑ Load Distribution
- ❑ Storage Capacity

```
mongos> sh.enableSharding("projectUser2");
{
        "ok" : 1,
        "operationTime" : Timestamp(1631040629, 3),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1631040629,
3),
                "signature" : {
                        "hash" :
BinData(0,"Z6Iao6SwiQNVVg1AfiXIWA4tdV4="),
                        "keyId" :
NumberLong("6975140761171918849")
                }
        }
```

Non-functional Requirements

Data Consistency.

- ❑ Default readConcern for Mongo 4.4 is “available”
- ❑ Default writeConcern for Mongo 4.4 is w: 1

Read Concern	Write Concern	Read own writes	Monotonic reads	Monotonic writes	Writes follow reads
`"majority"`	`"majority"`	✅	✅	✅	✅
`"majority"`	`{ w: 1 }`		✅		✅
`"local"`	`{ w: 1 }`				
`"local"`	`"majority"`			✅	

https://docs.mongodb.com/manual/core/causal-consistency-read-write-concerns/

Analyzing workload – Partitioning

1. Given a User ID, get the user's information. (user's first name, last name, email address, phone number, gender, country, if he is a host(is_host), host since, about him, his response time, picture url, identity verified status)
2. User makes a booking.
3. Given a User ID, get the booking history.(Booking ID, Listing ID(property ID), Total Price, No of Nights, Booking Date)
4. Get the number of bookings of each user.

- The aggregate is partitioned by User ID, as it is looked up for multiple queries.

```
mongos> db.createCollection('users_bookings');

mongos> sh.shardCollection("projectUser2.users_bookings", {"_id" : "hashed"});
```

Analysing workload – Partitioning

1. Given a listing Id, get the name, description, location, amenities, number of baths, number of beds, property type, has_availability, instantly bookable.
2. Given a User ID, get all of listings.
3. Get all the Listings' names, prices, picture_urls, based on neighbourhood area and the number of people the property accomodates.
4. To the above query, price filter is applied.
5. Given User's Coordinates, Get all the listings's name, price, picture of the property(URL), coordinates in 1km radius.
6. Get the number of properties for each property type.
7. Add a new listing and change the user's is_host property to true.

- In order to optimize the execution of these workload queries, 2 instances of the data is required. R1, R2.
- R1 is partitioned by listingID and an index is applied on user ID. This satisfies Workload 1 and 2.
- R2 is partitioned by neighbourhood's name.
- Compound index on accommodates and price is applied for 3,4 workloads.
- For workload 5, a geospatial index is applied on coordinates object. This index can be applied in either R1 or R2.
- Workload 7 is addressed in the following slides.

```
mongos> db.createCollection("listings_1");
mongos> sh.shardCollection("projectUser2.listings_1", {"_id" : "hashed"});

db.listings_1.createIndex( { user_id: 1 } );

mongos> db.createCollection('listings');
mongos> sh.shardCollection("projectUser2.listings", { "location.neighbourhood_area": "hashed"});

db.listings.createIndex( { 'accommodates': 1 ,'price' : 1} );
db.listings.createIndex( { "location.coords": "2dsphere" } );
```

Analysing workload – Partitioning

1. User writes a review about one listing (with all review_scores_rating,review_scores_cleanliness, review_scores_checkin, review_scores_communication, review_score_value, review_scores_location).
2. Get all reviews of a Listing ID.
3. Get the average rating of all kinds of ratings of each property based (cleanliness, checkin, communication, value, accuracy)
4. Order the average rating of neighbourhood area based on user score of the location.

- In order to optimize the execution of these workload queries, 2 instances of the data is required. R1, R2.
- R1 is partitioned by ListingID to satisfy the 2nd workload.
- R2 is partitioned by neighbourhood area's name.

```
mongos> db.createCollection('reviews_1’);

mongos> sh.shardCollection("projectUser2.reviews_1", { "listing_id": "hashed"});

mongos> db.createCollection('reviews');

mongos> sh.shardCollection("projectUser2.reviews", { "neighbourhood_area": "hashed"});
```

Data Import

mongoimport --db=projectUser2 --collection=users_bookings --authenticationDatabase "projectUser2" -u user2 -p 5l7n5z --file=users_bookings.json

mongoimport --db=projectUser2 --collection=listings --authenticationDatabase "projectUser2" -u user2 -p 5l7n5z --file=listings.json

mongoimport --db=projectUser2 --collection=reviews --authenticationDatabase "projectUser2" -u user2 -p 5l7n5z --file=reviews.json

mongoimport --db=projectUser2 --collection=listings_1 --authenticationDatabase "projectUser2" -u user2 -p 5l7n5z --file=listings_1.json

mongoimport --db=projectUser2 --collection=reviews_1 --authenticationDatabase "projectUser2" -u user2 -p 5l7n5z --file=reviews_1.json

```
{
        "_id" : 8536,
        "first_name" : "Jessica",
        "last_name" : "Bosco",
        "gender" : "M",
        "country" : "Italy",
        "country_code" : "IT",
        "phone_number" : "+39 69879224776",
        "email" : "5ptjgkham0x7kep8pz74@yahoo.com",
        "is_host" : false,
        "bookings" : [
                {
                        "_id" : 134393,
                        "listing_id" : 4898741,
                        "total_price" : 3850,
                        "no_of_nights" : 22,
                        "booking_date" : "09/08/2018"
                }
        ]
}
```


Users_bookings collection

Listings collection


```
{
        "_id" : 4898741,
        "name" : "Luxurious houseboat on Amstel River",
        "description" : "Luxurious houseboat located at a very central but quiet location on
the Amstel river directly at the famous skinny bridge and the Hermitage museum and on walking
distance to most touristic sites. Staying on this ship will be an indigenous experience!<br
/><br /><b>The space</b><br />The ship is very comfortable with central heating, floor
heating in living room, bathroom / wellness area with own hammam (!) and luxurious fully
equipped kitchen. Sitting on the sofa in the living room you overlook the river and canals
and can expect swans passing by and knocking on the windows. Both during day and night the
view is breathtaking.<br /><br /><b>Guest access</b><br />The complete ship<br /><br
/><b>Other things to note</b>",
        "neighborhood_overview" : "MS \"Morgenster\" is build in 1907 as a freight ship and
recently renovated into a very luxurious apartment. Located at a very central but quiet
location on the Amstel river directly at the famous skinny bridge and the Hermitage museum
and on walking distance to most touristic sites. Staying on this ship will be an indigenous
experience!",
        "picture_url" : "https://a0.muscache.com/pictures/71412632/f74b0a67_original.jpg",
        "user_id" : 3356377,
        "price" : 175,
        "has_availability" : true,
        "instantly_bookable" : false,
        "property_type" : "Houseboat",
        "room_type" : "Entire home/apt",
        "accommodates" : 4,
        "bathrooms_text" : "1 bath",
        "bedrooms" : 2,
        "beds" : 2,
        "location" : {
                "_id" : 4898741,
                "country" : "NL",
                "city" : "Amsterdam",
                "neighbourhood_area" : "Centrum-Oost",
                "coords" : [
                        52.36326,
                        4.90345
                ]
        },
        "amenities" : [
                "Dishwasher",
                "Long term stays allowed",
                "Microwave",
                "Wifi“
              ]
}
```



```
{
    "_id" : 26008180,
    "reviewer_id" : 789143,
    "comments" : "The houseboat was exactly as pictured and described. “
    "date" : ISODate("2015-01-01T23:00:00Z"),
    "review_scores_rating" : 97,
    "review_scores_accuracy" : 10,
    "review_scores_cleanliness" : 9,
    "review_scores_checkin" : 10,
    "review_scores_communication" : 10,
    "review_scores_location" : 10,
    "review_scores_value" : 9,
    "neighbourhood_area" : "Centrum-Oost",
    "price" : 175,
    "listing_id" : 4898741
}
```


Reviews collection

Implementing workload using MongoDB

1. Given a User ID, get the user's information. (user's first name, last name, email address, phone number, gender, country, if he is a host(is_host), host since, about him, his response time, picture url, identity verified status)

```
    db.users_bookings.find({"_id": 30390},{first_name:
1,last_name:1,gender:1,country:1,country_code:1,phone_
number:1,
        phone_number:1,email:1,is_host:1, host_about:1
, host_response_time:1, host_response_rate:1, host_is_
superhost:1, host_picture_url:1,host_identity_verified
:1 }).pretty();
```


```
{
    "_id" : 37404,
    "first_name" : "Thomas R.",
    "last_name" : "Fahey",
    "gender" : "F",
    "country" : "United States",
    "country_code" : "US",
    "phone_number" : "+1 45875127419",
    "email" : "ma0fjo7gtz851jkxyoui@yahoo.com",
    "is_host" : false
}

{
    "_id" : 30390,
    "first_name" : "Federica",
    "last_name" : "Herman",
    "gender" : "F",
    "country" : "Netherlands",
    "country_code" : "NL",
    "phone_number" : "+31 87429999835",
    "email" : "klf3d7k7y7wfc85i7zga@gmail.com",
    "is_host" : true,
    "host_about" : "Hello!\r\nI'm an italian architect living in The Netherlands!'
    "host_response_time" : "a few days or more",
    "host_response_rate" : "0%",
    "host_is_superhost" : 0,
    "host_picture_url" : "https://a0.muscache.com/im/pictures/user/eabcd253
a46d-4b21-b735-76b67d94afdb.jpg?aki_policy=profile_x_medium",
    "host_identity_verified" : 1
}
```

Implementing workload using MongoDB

2. User makes a booking. ($push)

```
db.users_bookings.updateOne(
    {"_id": 37404},
    {'$push' : {bookings: {"_id": 340983 , listing_id:
 340992, total_price:140, number_of_nights:2, booking_
date: "08/09/2021"} }
    },{writeConcern: {w:'majority'}});
```

3 . Given a User ID, get the booking history.(each booking has Booking ID, Listing ID(property ID), Total Price, No of Nights, Booking Date)

```
db.users_bookings.findOne({"_id": 37404}, {bookings: 1
});
```


```
{
    "_id" : 37404,
    "bookings" : [
        {
            "_id" : 389565,
            "listing_id" : 28976111,
            "total_price" : 150,
            "no_of_nights" : 2,
            "booking_date" : "28/04/2019"
        },
        {
            "_id" : 340983,
            "listing_id" : 340992,
            "total_price" : 140,
            "number_of_nights" : 2,
            "booking_date" : "08/09/2021"
        }
    ]
}
```


Implementing workload using MongoDB

4. Get the number of bookings of each listing in descending order. ($unwind)

```
db.users_bookings.aggregate( [
    {"$unwind": "$bookings"},
    { "$group": { "_id": "$bookings.listi
ng_id", "noOfBookings": { '$sum': 1 }, }
        },
    { "$sort" : {noOfBookings: -1 } }
    ] );
```

```
{ "_id" : 82482, "noOfBookings" : 860 }
{ "_id" : 802052, "noOfBookings" : 798 }
{ "_id" : 785432, "noOfBookings" : 783 }
{ "_id" : 1814121, "noOfBookings" : 772 }
{ "_id" : 35632344, "noOfBookings" : 707 }
{ "_id" : 694850, "noOfBookings" : 702 }
{ "_id" : 35927687, "noOfBookings" : 695 }
{ "_id" : 851044, "noOfBookings" : 691 }
{ "_id" : 1247334, "noOfBookings" : 686 }
{ "_id" : 654931, "noOfBookings" : 682 }
{ "_id" : 68290, "noOfBookings" : 659 }
{ "_id" : 4210531, "noOfBookings" : 620 }
{ "_id" : 608432, "noOfBookings" : 619 }
{ "_id" : 4449764, "noOfBookings" : 617 }
{ "_id" : 6441316, "noOfBookings" : 608 }
{ "_id" : 193038, "noOfBookings" : 598 }
{ "_id" : 7276869, "noOfBookings" : 595 }
{ "_id" : 68873, "noOfBookings" : 579 }
{ "_id" : 655216, "noOfBookings" : 578 }
{ "_id" : 2497346, "noOfBookings" : 577 }
```

Implementing workload using MongoDB

5. Given a Listing id, get the name, description, location, amenities, number of baths, number of beds, property type, has_availability, instantly bookable.

```
db.listings_1.findOne({
    _id : 340992
},
{name :1 , decription : 1, location : 1, amenities : 1
, bathrooms_text : 1, beds : 1, bedrooms : 1, property
_type : 1, has_availability : 1, instantly_bookable: 1
});
```


```
{
    "_id" : 340992,
    "location" : {
        "country" : "NL",
        "city" : "Amsterdam",
        "neighbourhood_area" : "Centrum-Oost",
        "coords" : [
            52.37,
            3.89393
        ]
    },
    "name" : "Stylish  light apartment in de pijp",
    "has_availability" : true,
    "instantly_bookable" : false,
    "property_type" : "Entire apartment",
    "bathrooms_text" : "1 bath",
    "bedrooms" : 1,
    "beds" : 1,
    "amenities" : [
        "wifi",
        "kitchen"
    ]
}
```


Implementing workload using MongoDB

6. Get all listings of a user id.

```
db.listings_1.find({"user_id":  11812316});
```

```
db.listings_1.find({"user_id":  11812316})
.explain("executionStats");
```

```
db.listings_1.count();
17827
```

Execution stats without Index


```
"executionStats" : {
"nReturned" : 2,
"executionTimeMillis" : 8,
"totalKeysExamined" : 0,
"totalDocsExamined" : 17827,
"shards" : [
{
"shardName" : "rs0",
"executionSuccess" : true,
"nReturned" : 1,
"executionTimeMillis" : 6,
"totalKeysExamined" : 0,
"totalDocsExamined" : 5917,
}
{
"shardName" : "rs1",
"executionSuccess" : true,
"nReturned" : 1,
"executionTimeMillis" : 6,
"totalKeysExamined" : 0,
"totalDocsExamined" : 5804
}
{
"shardName" : "rs2",
"executionSuccess" : true,
"nReturned" : 1,
"executionTimeMillis" : 7,
"totalKeysExamined" : 0,
"totalDocsExamined" : 6106
}]
```


Execution stats with Index


```
"executionStats" : {
"nReturned" : 2,
"executionTimeMillis" : 2,
"totalKeysExamined" : 2,
"totalDocsExamined" : 2,
"shards" : [
{
"shardName" : "rs0",
"executionSuccess" : true,
"nReturned" : 1,
"executionTimeMillis" : 0,
"totalKeysExamined" : 1,
"totalDocsExamined" : 1,
}
{
"shardName" : "rs1",
"executionSuccess" : true,
"nReturned" : 1,
"executionTimeMillis" : 0,
"totalKeysExamined" : 1,
"totalDocsExamined" : 1,
}
{
"shardName" : "rs2",
"executionSuccess" : true,
"nReturned" : 0,
"executionTimeMillis" : 0,
"totalKeysExamined" : 0,
"totalDocsExamined" : 0,
}]
```


Implementing workload using MongoDB

7. Get all the Listings' names, prices, picture_urls, based on neighbourhood area and the number of people the property accommodates. (if the requested page number 1, return 10 first records, else return next 10 for page 2…) (using $limit and $skip)

```
db.listings.find(
    { 'location.neighbourhood_area' : "Zuid", accommod
ates: 2},
    {name : 1, picture_url: 1, price : 1}

db. listings.find({ 'location.neighbourhood_area' : "Z
uid", accommodates: 2},{name : 1, picture_url: 1, pric
e : 1}).skip(10).limit(10);
```


{ "_id" : 5299925, "name" : "Apartment a stone's throw from the Rijksmuseam!", "picture_url" "https://a0.muscache.com/pictures/ae90e473-6674-42fd-ba11-a437259faf08.jpg", "price" : 99 }
{ "_id" : 8963247, "name" : "Home 200 mtr. from Museumsquare", "picture_url" : "https://a0.muscache.com/pictures/ce33a349-20b2-4123-8dfb-cce0593f5abf.jpg", "price" : 75 }
{ "_id" : 1018703, "name" : "Cosy private studio on houseboat in the centre!*", "picture_url" : "https://a0.muscache.com/pictures/73004ad9-e8be-4e5c-a8f1-d947e4637dc0.jpg", "price" : 86 }
{ "_id" : 1102657, "name" : "Studio with balcony near Vondelpark", "picture_url" : "https://a0.muscache.com/pictures/3408018e-70fc-4ed9-b327-7a5ca1297abe.jpg", "price" : 60 }
{ "_id" : 9816257, "name" : "Houseboat studio with canal view and bikes", "picture_url" : "https://a0.muscache.com/pictures/b24d9b8a-6173-47ae-9269-ad4e8faf3afe.jpg", "price" : 77 }
{ "_id" : 16378597, "name" : "Bright and cosy apartment", "picture_url" : "https://a0.muscache.com/pictures/13425234-bdf4-4ddb-92b5-17fd9b148dc5.jpg", "price" : 75 }
{ "_id" : 20127193, "name" : "Stylish boho apartment in Amsterdam South", "picture_url" : "https://a0.muscache.com/pictures/1aa952aa-0107-47e1-a275-100735561e32.jpg", "price" : 95 }
{ "_id" : 85008, "name" : "Museum district/Vondelpark studio-apartment", "picture_url" : "https://a0.muscache.com/pictures/miso/Hosting-85008/original/41e6e043-f6dc-41cb-b3cf-134c33527aa5.jpeg", "price" : 66 }
{ "_id" : 43980, "name" : "View into park / museum district (long/short stay)", "picture_url" : "https://a0.muscache.com/pictures/923eae51-40df-4783-b2b1-7032966a23a3.jpg", "price" : 65 }
{ "_id" : 97476, "name" : "Urban Oasis~beautiful street in Old South", "picture_url" : "https://a0.muscache.com/pictures/823bec74-2169-40f3-a1cc-485c92d3c847.jpg", "price" : 99 }
{ "_id" : 213721, "name" : "Amsterdam Apartment (waterview)", "picture_url" : "https://a0.muscache.com/pictures/85547760/4de4bff9_original.jpg", "price" : 151 }
{ "_id" : 272597, "name" : "Luxury appartm, good neighbourhood", "picture_url" : "https://a0.muscache.com/pictures/2745454/242ed6d2_original.jpg", "price" : 145 }
{ "_id" : 1630603, "name" : "Charming quite room with fast Wifi", "picture_url" : "https://a0.muscache.com/pictures/33a2e342-018e-42c9-be4a-6105f742d648.jpg", "price" : 41 }
{ "_id" : 1057494, "name" : "Lovely bright room with balcony and beatiful view!", "picture_url" : "https://a0.muscache.com/pictures/16397337/5539c9c0_original.jpg", "price" : 79 }
{ "_id" : 447151, "name" : "creative ground floor house close to Vondelpark", "picture_url" : "https://a0.muscache.com/pictures/054dc658-909e-48c3-8cd1-74b358cdf7d8.jpg", "price" : 105 }
{ "_id" : 32325478, "name" : "Spacious room in a bright and big apartment", "picture_url" : "https://a0.muscache.com/pictures/7e0e8e6f-69ea-4e50-aa3e-bad455d572cd.jpg", "price" : 60 }
{ "_id" : 11962960, "name" : "â˜†stylish and spacious Apt in great area w balconyâ˜†", "picture_url" : "https://a0.muscache.com/pictures/miso/Hosting-11962960/original/efa831f1-dabd-4a12-b780-810bff53eb05.jpeg", "price" : 103 }
{ "_id" : 12868928, "name" : "cosy apartment Amsterdam south", "picture_url" : "https://a0.muscache.com/pictures/bd01b991-384f-4b00-89a2-71ccd728fff0.jpg", "price" : 100 }
{ "_id" : 28375507, "name" : "Amsterdam Penthouse Apartment", "picture_url" : "https://a0.muscache.com/pictures/3a25ff00-2df0-4082-97fa-5bb1a0db628d.jpg", "price" : 150 }
{ "_id" : 14752749, "name" : "Spacious an bright apartment next to Vondelpark", "picture_url" : "https://a0.muscache.com/pictures/3711676c-98b0-47a8-80b9-6f3b10606945.jpg", "price" : 120

Implementing workload using MongoDB

8. To workload 7 , price filter is applied.

```
db.listings.find({
        'location.neighbourhood_area': 'Bijlmer-
Centrum', 'accommodates' :2,
         'price': {
        $lt: 100,
        $gt : 0
        }
        },
        {
        name : 1, picture_url: 1, price : 1
        }
        ).limit(10);
```


```
{ "_id" : 954629, "name" : "Stylish, clean studio, high service", "picture_url" :
"https://a0.muscache.com/pictures/bad11e40-3e1b-4410-af9a-1623fe6cfb2e.jpg", "price" : 72 }
{ "_id" : 954633, "name" : "Private studio 25m2 for two near A'dam Arena", "picture_url" :
"https://a0.muscache.com/pictures/6f98344b-5c41-4cac-8f9e-2462c346fe07.jpg", "price" : 79 }
{ "_id" : 5875932, "name" : "Nice place to saty, to enjoy! :)", "picture_url" :
"https://a0.muscache.com/pictures/270187b1-6ba6-4cc1-9f76-53995c51d2b6.jpg", "price" : 65 }
{ "_id" : 4336334, "name" : "Cozy room for two", "picture_url" :
"https://a0.muscache.com/pictures/9a014ee9-6741-4039-8d12-876eef1b31f1.jpg", "price" : 55 }
{ "_id" : 16822162, "name" : "Very big private room > center/airport  nearby!", "picture_url"
: "https://a0.muscache.com/pictures/f838c467-f437-4ad4-905a-b085ced472a2.jpg", "price" : 50 }
{ "_id" : 27971206, "name" : "Modern and cozy room 20 mins by metro from centre",
"picture_url" : "https://a0.muscache.com/pictures/87bb4034-b85d-4945-8a48-6ed540f8c5fd.jpg",
"price" : 49 }
{ "_id" : 19495563, "name" : "GREAT APARTMENT CLOSE TO THE CENTRE IN AMSTERDAM", "picture_url"
: "https://a0.muscache.com/pictures/75f49081-3d21-4d94-95e1-78437e201582.jpg", "price" : 45 }
{ "_id" : 43976021, "name" : "Bright and spacious bedroom close to metro", "picture_url" :
"https://a0.muscache.com/pictures/miso/Hosting-43976021/original/404c87b8-ea99-47b9-ba17-
3971042e3a97.jpeg", "price" : 35 }
{ "_id" : 15105977, "name" : "Lovely private 35m2 studio in Amsterdam", "picture_url" :
"https://a0.muscache.com/pictures/7e99d902-b2cf-43c3-9f55-9d1b4666992f.jpg", "price" : 87 }
{ "_id" : 24528823, "name" : "Cosy apartment in Amsterdam near metro", "picture_url" :
"https://a0.muscache.com/pictures/10e5b96a-4816-45a2-9d2c-97d96290da43.jpg", "price" : 88 }
```


Implementing workload using MongoDB

9. Given User's Coordinates, Get all the listings's name, price, picture_url , coordinates in 1km radius. (geospatial index)

```
db.listings.find( { 'location.coords': { $geoWithin: { $centerSphere: [ [ 52.3663823,4.9042306] , 1 / 6378.1 ] } } },
        { "name": 1 , "location.coords" : 1, price : 1});
```


```
{ "_id" : 48638609, "name" : "Ideal Double Standard At Amsterdam", "price" : 154, "location" : { "coords" : [ 52.3703, 4.89622 ] } }
{ "_id" : 10496067, "name" : "city cultural apartment Amsterdam", "price" : 98, "location" : { "coords" : [ 52.36984, 4.89629 ] } }
{ "_id" : 42417593, "name" : "Spacious townhouse in centre Amsterdam", "price" : 90, "location" : { "coords" : [ 52.36888, 4.89635 ] } }
{ "_id" : 16825279, "name" : "Best view in the hart of the center", "price" : 200, "location" : { "coords" : [ 52.36891, 4.89586 ] } }
{ "_id" : 809026, "name" : "Boss", "price" : 119, "location" : { "coords" : [ 52.36925, 4.89579 ] } }
{ "_id" : 46348037, "name" : "Rembrandt Studio One", "price" : 45, "location" : { "coords" : [ 52.36722, 4.89531 ] } }
{ "_id" : 8303992, "name" : "The Secret Chapel", "price" : 400, "location" : { "coords" : [ 52.36597, 4.89574 ] } }
{ "_id" : 19030625, "name" : "Apartment next to Dam Square | With a cat", "price" : 225, "location" : { "coords" : [ 52.37175, 4.89931 ] } }
{ "_id" : 9947853, "name" : "Spacious appartment in city centre", "price" : 185, "location" : { "coords" : [ 52.37151, 4.89919 ] } }
{ "_id" : 12971906, "name" : "Authentic Amsterdam Canal Apt WITH PRIVATE CHEF", "price" : 57, "location" : { "coords" : [ 52.3695, 4.89661 ] } }
{ "_id" : 46939988, "name" : "Oudezijds Voorburgwal - 1 bedroom - Sleeps 2", "price" : 90, "location" : { "coords" : [ 52.37045, 4.89679 ] } }
{ "_id" : 28798431, "name" : "Delft Blue Suite", "price" : 25, "location" : { "coords" : [ 52.37021, 4.89717 ] } }
{ "_id" : 8413979, "name" : "Comlete independent quiet studio", "price" : 36, "location" : { "coords" : [ 52.37059, 4.89758 ] } }
{ "_id" : 48660090, "name" : "Exclusive Double Premium At Amsterdam", "price" : 226, "location" : { "coords" : [ 52.37054, 4.89769 ] } }
{ "_id" : 20405959, "name" : "Renovated, Cozy Room in City Centre", "price" : 25, "location" : { "coords" : [ 52.37037, 4.89785 ] } }
{ "_id" : 23119390, "name" : "Perfect Canal apartment @RedLightDistrict", "price" : 90, "location" : { "coords" : [ 52.37113, 4.89846 ] } }
{ "_id" : 5675874, "name" : "Beautiful apartment with great view", "price" : 200, "location" : { "coords" : [ 52.37161, 4.89862 ] } }
{ "_id" : 34052508, "name" : "Double Bed, Single Room in the Heart of Amsterdam", "price" : 145, "location" : { "coords" : [ 52.37184, 4.89854 ] } }
{ "_id" : 47012402, "name" : "Longterm centre Amsterdam", "price" : 111, "location" : { "coords" : [ 52.37193, 4.89813 ] } }
{ "_id" : 251444, "name" : "Luxurious apartment at the Prinsengracht", "price" : 65, "location" : { "coords" : [ 52.37169, 4.89807 ] } }
```


Implementing workload using MongoDB

10. Get the number of properties grouped by property type.($sum)

```
db.listings_1.aggregate(
          [
            {
              "$group":
                {
                  "_id": "$property_type",
                  "count": { '$sum': 1 },
                }
            }
          ]
        );
```

```
{ "_id" : "Entire condominium", "count" : 251 }
{ "_id" : "Shared room in houseboat", "count" : 2 }
{ "_id" : "Private room in cabin", "count" : 3 }
{ "_id" : "Entire cabin", "count" : 5 }
{ "_id" : "Room in hotel", "count" : 101 }
{ "_id" : "Entire guesthouse", "count" : 16 }
{ "_id" : "Bus", "count" : 1 }
{ "_id" : "Private room in apartment", "count" : 2152 }
{ "_id" : "Tipi", "count" : 1 }
{ "_id" : "Entire cottage", "count" : 12 }
{ "_id" : "Private room in guest suite", "count" : 106 }
{ "_id" : "Entire place", "count" : 8 }
{ "_id" : "Shared room in boat", "count" : 5 }
{ "_id" : "Campsite", "count" : 2 }
{ "_id" : "Houseboat", "count" : 186 }
{ "_id" : "Private room in lighthouse", "count" : 1 }
{ "_id" : "Entire home/apt", "count" : 2 }
{ "_id" : "Private room in tiny house", "count" : 9 }
{ "_id" : "Private room in hostel", "count" : 26 }
{ "_id" : "Private room in townhouse", "count" : 167 }
```

Implementing workload using MongoDB

11. Add a new listing and change the user's is_host property to true. (using Transactions)

```
  mongos> var location = {country:"NL",city:"Amsterdam",neighb
ourhood_area:"Zuid",coords:[52.3396576,4.8711542]};
        mongos> var amenities = ["wifi", "kitchen"];
        mongos> var host_date = new Date(2021, 09, 10, 1, 12)
;
        mongos> db.listings_1.insertOne({_id: 340993, user_id
 : 33284, location:location, name : "Stylish  light apartment
in Zuid",description: "My stylish and comfortable one bedroom
apartment is located in the most beautiful place in Amsterdam,
 close to the subway.",neighborhood_overview:"The Zuid is one
of most enjoyable areas of Amsterdam.",picture_url:"https://a0
.muscache.com/pictures/ba15946c-c019-46de-85b6-
3d83bd896a7a.jpg",price:120,has_availability:true,instantly_bo
okable:true,property_type:"Entire apartment",room_type:"Entire
 home/apt",accommodates:4,bathrooms_text:"2 bath",bedrooms:3,b
eds:4, amenities: amenities});

        { "acknowledged" : true, "insertedId" : 340993 }
```

```
        var session1 = db.getMongo().startSession({readPreference: { mode: "primary" }});
        var sessionUser = session1.getDatabase('projectUser2').getCollection('users_bookin
gs');
        var sessionListing = session1.getDatabase('projectUser2').getCollection('listings_
1');
        var location = {country:"NL", city:"Amsterdam",neighbourhood_area:"Centrum-
Oost",coords:[52.37, 3.89393]};
        var amenities = ["wifi", "kitchen"];
        session1.startTransaction({readConcern:{level: 'snapshot'},writeConcern: {w: 'majo
rity'}});
        var host_date = new Date(2021, 09, 09, 13, 12);

        try {
        sessionUser.update(
        {_id: 33284},
        {'$set' : {is_host: true, host_since: host_date, host_is_superhost: 0, host_identi
ty_verified : 1}
        });
        sessionListing.insertOne(
        {_id: 340992, user_id : 33284, location:location, name : "Stylish  light apartment
 in de pijp",description: "My stylish and comfortable one bedroom apartment is located in t
he most beautiful place in Amsterdam.",neighborhood_overview:"The Pijp is one of most enjoy
able areas of Amsterdam. It is very peaceful and beautiful.",picture_url:"https://a0.muscac
he.com/pictures/ba15946c-c019-46de-85b6-
3d83bd896a7a.jpg",price:70,has_availability:true,instantly_bookable:false,property_type:"En
tire apartment",room_type:"Entire home/apt",accommodates:2,bathrooms_text:"1 bath",bedrooms
:1,beds:1, amenities: amenities
        });
        } catch (error) {
        session1.abortTransaction();
        throw error;
        }
        session1.commitTransaction();
        session1.endSession();

        { "acknowledged" : true, "insertedId" : 340992 }
```

Implementing workload using MongoDB

12. User writes a review about one listing (with all review_scores_rating,review_scores_cleanliness, review_scores_checkin, review_scores_communication, review_score_value, review_scores_location).

```
db.reviews_1.insertOne({
    _id:851025,
    reviewer_id:1008593,
    comments:"Excellent accommodation, close to ever
ything, lovely hosts, beautiful",date:
    new Date("2021-09-12T22:00:00.000+00:00"),
    review_scores_rating:99,
    review_scores_accuracy:10,
    review_scores_cleanliness:10,
    review_scores_checkin:10,
    review_scores_communication:10,
    review_scores_location:10,
    review_scores_value:10,
    listing_id:340992,
    neighbourhood_area: “Ziuid”,
    price : 40
     });

    { "acknowledged" : true, "insertedId" : 851025 }
```

Implementing workload using MongoDB

13. Get all reviews given a Listing ID.

```
db.reviews_1.find({listing_id : 12332987 }).pr
etty();
```


```
{
        "_id" : 74428652,
        "reviewer_id" : 64342256,
        "comments" : "We had a wonderful stay at Janine's apartment. She is a great
host. Fully equipped place, great view very clean. Highly recommended!",
        "date" : ISODate("1970-01-01T00:00:00Z"),
        "review_scores_rating" : 100,
        "review_scores_accuracy" : 10,
        "review_scores_cleanliness" : 10,
        "review_scores_checkin" : 10,
        "review_scores_communication" : 10,
        "review_scores_location" : 10,
        "review_scores_value" : 10,
        "neighbourhood_area" : "De Baarsjes - Oud-West",
        "price" : 110,
        "listing_id" : 12332987,
        "accommodates" : 2
}
{
        "_id" : 72786711,
        "reviewer_id" : 66453761,
        "comments" : "Really nice apartments for staying in the Amsterdam. It has a good
kitchen which a big plus to simplify your living. You can get to the center of the
Amsterdam in 15-20 minutes by walk. Janine and Maarten are great hosts, I felt myself
like I was at home.",
        "date" : ISODate("2016-04-04T22:00:00Z"),
        "review_scores_rating" : 100,
        "review_scores_accuracy" : 10,
        "review_scores_cleanliness" : 10,
        "review_scores_checkin" : 10,
        "review_scores_communication" : 10,
        "review_scores_location" : 10,
        "review_scores_value" : 10,
        "neighbourhood_area" : "De Baarsjes - Oud-West",
        "price" : 110,
        "listing_id" : 12332987
}
```


Implementing workload using MongoDB

14. Get the average rating of all kinds of ratings of each property based on cleanliness, checkin, communication, value, accuracy

```
db.reviews_1.aggregate(
            [
              {
                "$group":
                 {
                   "_id": "$listing_id",
                   "avgRatingOverall": { "$avg": "$re
view_scores_rating" },
                   "avgRatingAccuracy": { "$avg": "$r
eview_scores_accuracy" },
                   "avgRatingCleanliness": { "$avg":
"$review_scores_cleanliness" },
                   "avgRatingCheckin": { "$avg": "$re
view_scores_checkin" },
                   "avgRatingCommu": { "$avg": "$revi
ew_scores_communication" },
                    "avgRatingValue": { "$avg": "$rev
iew_scores_value" }
                 }
              }
            ]
        );
```


```
{ "_id" : 11275833, "avgRatingOverall" : 100, "avgRatingAccuracy" : 10,
"avgRatingCleanliness" : 10, "avgRatingCheckin" : 10, "avgRatingCommu" : 10,
"avgRatingValue" : 10 }
{ "_id" : 34877330, "avgRatingOverall" : 91, "avgRatingAccuracy" : 9,
"avgRatingCleanliness" : 9, "avgRatingCheckin" : 9, "avgRatingCommu" : 10,
"avgRatingValue" : 9 }
{ "_id" : 25257623, "avgRatingOverall" : 100, "avgRatingAccuracy" : 10,
"avgRatingCleanliness" : 10, "avgRatingCheckin" : 10, "avgRatingCommu" : 10,
"avgRatingValue" : 10 }
{ "_id" : 41688975, "avgRatingOverall" : 93, "avgRatingAccuracy" : 9,
"avgRatingCleanliness" : 10, "avgRatingCheckin" : 10, "avgRatingCommu" : 9,
"avgRatingValue" : 9 }
{ "_id" : 11035643, "avgRatingOverall" : 97, "avgRatingAccuracy" : 10,
"avgRatingCleanliness" : 10, "avgRatingCheckin" : 10, "avgRatingCommu" : 10,
"avgRatingValue" : 10 }
{ "_id" : 44024370, "avgRatingOverall" : 90, "avgRatingAccuracy" : 9,
"avgRatingCleanliness" : 9, "avgRatingCheckin" : 9, "avgRatingCommu" : 10,
"avgRatingValue" : 8 }
{ "_id" : 23031953, "avgRatingOverall" : 80, "avgRatingAccuracy" : 8,
"avgRatingCleanliness" : 6, "avgRatingCheckin" : 10, "avgRatingCommu" : 10,
"avgRatingValue" : 8 }
{ "_id" : 14938104, "avgRatingOverall" : 100, "avgRatingAccuracy" : 10,
"avgRatingCleanliness" : 10, "avgRatingCheckin" : 10, "avgRatingCommu" : 10,
"avgRatingValue" : 10 }
{ "_id" : 1704905, "avgRatingOverall" : 93, "avgRatingAccuracy" : 9,
"avgRatingCleanliness" : 9, "avgRatingCheckin" : 10, "avgRatingCommu" : 10,
"avgRatingValue" : 9 }
{ "_id" : 17187058, "avgRatingOverall" : 93, "avgRatingAccuracy" : 10,
"avgRatingCleanliness" : 8, "avgRatingCheckin" : 9, "avgRatingCommu" : 10,
"avgRatingValue" : 9 } ….
```


Implementing workload using MongoDB

15. Order the average rating of neighbourhood area based on user score of the location.

```
db.reviews.aggregate(
            [
              {
                "$group":
                 {
                   "_id": "$neighbourhood_area",
                   "avgRatingLocation": { "$avg": "$review_
scores_location" },
                 }
              },
              {"$sort":{avgRatingLocation : -1}}
            ]
          );
```

16. Get the average rating of the neighbourhood area “Zuid”. ($match)

```
db.reviews.aggregate(
            [
               { "$match" : { neighbourhood_area : "Zuid" } },
              {
                "$group":
                 {
                   "_id": "$neighbourhood_area",
                   "avgRatingLocation": { "$avg": "$review_scores_locat
ion" },
                 }
              } ]);
```

```
{ "_id" : "Centrum-West", "avgRatingLocation" : 9.969346057611306 }
{ "_id" : "Centrum-Oost", "avgRatingLocation" : 9.903504626895058 }
{ "_id" : "De Pijp - Rivierenbuurt", "avgRatingLocation" : 9.825147144159828 }
{ "_id" : "Zuid", "avgRatingLocation" : 9.682558096022154 }
{ "_id" : "De Baarsjes - Oud-West", "avgRatingLocation" : 9.6628544697874 }
{ "_id" : "Westerpark", "avgRatingLocation" : 9.561889433693903 }
{ "_id" : "Oud-Oost", "avgRatingLocation" : 9.522287718614507 }
{ "_id" : "Oud-Noord", "avgRatingLocation" : 9.472269906245874 }
{ "_id" : "Oostelijk Havengebied - Indische Buurt", "avgRatingLocation" :
9.421457647911671 }
{ "_id" : "Noord-Oost", "avgRatingLocation" : 9.389780769894701 }
{ "_id" : "IJburg - Zeeburgereiland", "avgRatingLocation" : 9.336862911195821 }
{ "_id" : "Osdorp", "avgRatingLocation" : 9.31387132763109 }
{ "_id" : "Buitenveldert - Zuidas", "avgRatingLocation" : 9.295803183791607 }
{ "_id" : "Watergraafsmeer", "avgRatingLocation" : 9.251362950933766 }
{ "_id" : "Slotervaart", "avgRatingLocation" : 9.192068334350214 }
{ "_id" : "De Aker - Nieuw Sloten", "avgRatingLocation" : 9.168248090719741 }
{ "_id" : "Bos en Lommer", "avgRatingLocation" : 9.122705143996715 }
{ "_id" : "Bijlmer-Centrum", "avgRatingLocation" : 9.115959719255416 }
{ "_id" : "Noord-West", "avgRatingLocation" : 9.067109482363719 }
{ "_id" : "Geuzenveld - Slotermeer", "avgRatingLocation" : 9.062358276643991 }
```

Implementing workload using MongoDB

17. Average price of Zuid and Osdorp areas neighbourhood. (map reduce and $in)

```
var mapFunction1 = function() {
    emit(this.location.neighbourhood_area, this.price);
 };
 var reduceFunction1 = function(keyId, valuesPrices)
        {
    return Array.avg(valuesPrices);
 };
 db.listings.mapReduce(
    mapFunction1,
    reduceFunction1,
    { out: "map_reduce_location_price" }
 );
 db.map_reduce_location_price.find( { "_id": { "$in": [
"Osdorp", "Zuid" ] }});
```

```
{ "_id" : "Osdorp", "value" : 108.5 }
{ "_id" : "Zuid", "value" : 167.1990915972748 }
```

```
mongos> show collections;
listings
listings_1
map_reduce_location_price
reviews
reviews_1
users_bookings
```