SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

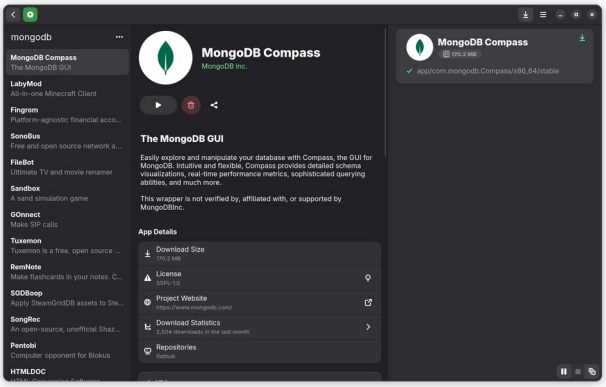
Name: Thanawin Pattanaphol **Date: 01324096**

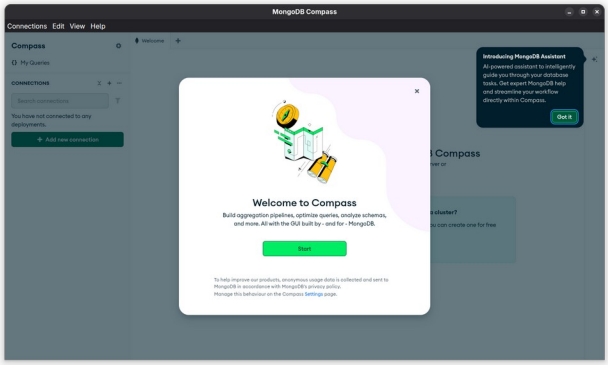
Part A: MongoDB Setup Instructions

1. Installation of MongoDB Community Edition

```
winsdominoes@bluefin:~$ sudo apt install -y mongodb-org
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  mongodb-database-tools mongodb-mongosh mongodb-org-database mongodb-org-database-tools-extra mongodb-org-mongos
  mongodb-org-server mongodb-org-shell mongodb-org-tools
The following NEW packages will be installed:
  mongodb-database-tools mongodb-mongosh mongodb-org mongodb-org-database mongodb-org-database-tools-extra
  mongodb-org-mongos mongodb-org-server mongodb-org-shell mongodb-org-tools
0 upgraded, 9 newly installed, 0 to remove and 0 not upgraded.
Need to get 191 MB of archives.
After this operation, 677 MB of additional disk space will be used.
```

2. MongoDB Compass Installation (Installed via Flathub for Linux)





SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

Name: Thanawin Pattanaphol **Date: 01324096**

3. Download the Sample Airbnb Dataset

I. Download the JSON dataset from MongoDB

```
❯ wget "https://atlas-education.s3.amazonaws.com/sampledata.archive"
Saving 'sampledata.archive'
HTTP response 200 OK [https://atlas-education.s3.amazonaws.com/sampledata.archive]
sampledata.archive   100% [===================================================================>]  353.09M    4.83MB/s
                         [Files: 1  Bytes: 353.09M [4.49MB/s] Redirects: 0  Todo: 0  Errors: 0]

~/Downloads via C v15.2.1-gcc via ☕ v25 via 🐍 v3.13.7 took 1m18s
❯
```

II. Install MongoDB Command Line Database Tools

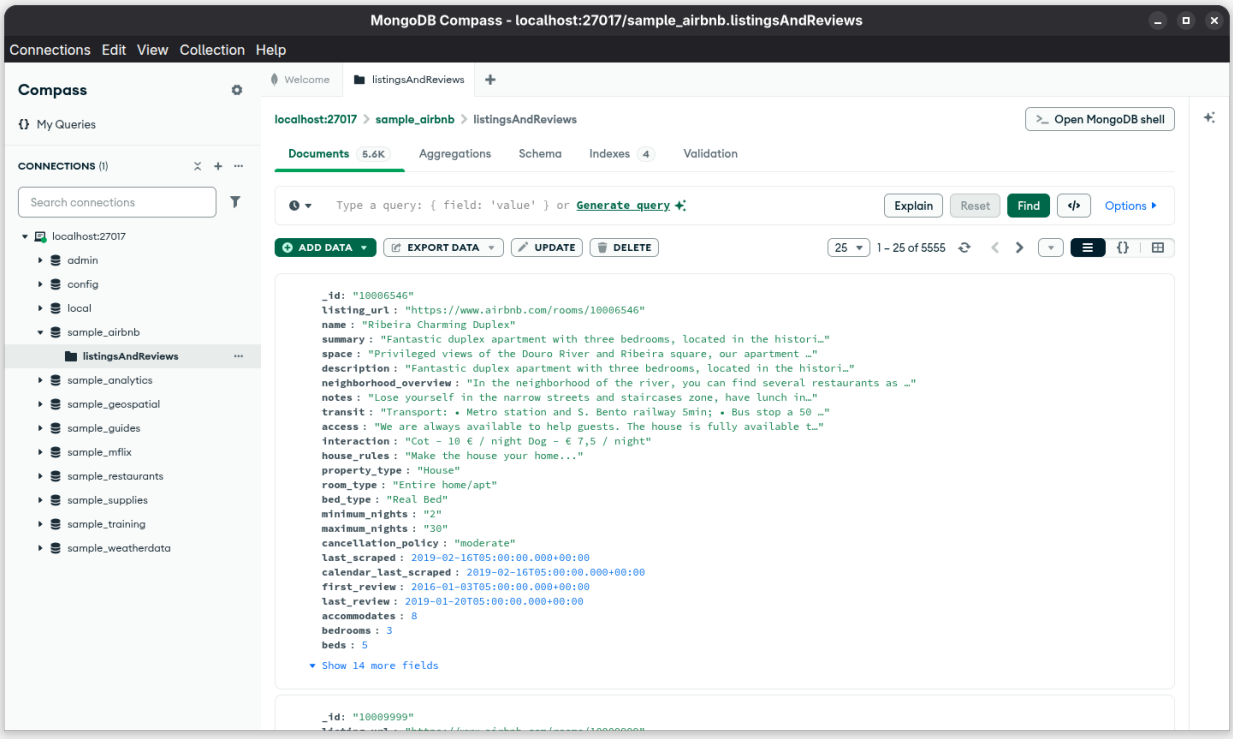
```
winsdominoes@bluefin:~/Downloads$ wget "https://fastdl.mongodb.org/tools/db/mongodb-database-tools-ubuntu2404-x86_64-1
00.13.0.tgz"
--2025-10-16 09:34:51--  https://fastdl.mongodb.org/tools/db/mongodb-database-tools-ubuntu2404-x86_64-100.13.0.tgz
Resolving fastdl.mongodb.org (fastdl.mongodb.org)... 18.172.202.99, 18.172.202.115, 18.172.202.11, ...
Connecting to fastdl.mongodb.org (fastdl.mongodb.org)|18.172.202.99|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 69684146 (66M) [binary/octet-stream]
Saving to: ‘mongodb-database-tools-ubuntu2404-x86_64-100.13.0.tgz’

mongodb-database-tools-ubuntu 100%[=============================================>]  66.46M  17.0MB/s    in 5.2s

2025-10-16 09:34:57 (12.8 MB/s) - ‘mongodb-database-tools-ubuntu2404-x86_64-100.13.0.tgz’ saved [69684146/69684146]

winsdominoes@bluefin:~/Downloads$ tar -xvf mongodb-database-tools-ubuntu2404-x86_64-100.13.0.tgz
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/LICENSE.md
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/README.md
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/THIRD-PARTY-NOTICES
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/mongodb-database-tools-ubuntu2404-x86_64-100.13.0.tgz.cdx.json
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/mongodb-database-tools-ubuntu2404-x86_64-100.13.0.tgz.sarif.json
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/bin/bsondump
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/bin/mongodump
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/bin/mongoexport
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/bin/mongofiles
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/bin/mongoimport
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/bin/mongorestore
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/bin/mongostat
mongodb-database-tools-ubuntu2404-x86_64-100.13.0/bin/mongotop
winsdominoes@bluefin:~/Downloads$ micro .bashrc
```

SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

Name: Thanawin Pattanaphol **Date: 01324096**

III. Mongorestore the DB

```
winsdominoes@bluefin:~/Downloads$ mongorestore --archive=sampledata.archive --port=27017
2025-10-16T10:03:13.081+0000    preparing collections to restore from
2025-10-16T10:03:13.088+0000    reading metadata for sample_mflix.users from archive 'sampledata.archive'
2025-10-16T10:03:13.088+0000    reading metadata for sample_mflix.comments from archive 'sampledata.archive'
2025-10-16T10:03:13.088+0000    reading metadata for sample_restaurants.restaurants from archive 'sampledata.archive'
2025-10-16T10:03:13.088+0000    reading metadata for sample_weatherdata.data from archive 'sampledata.archive'
2025-10-16T10:03:13.088+0000    reading metadata for sample_supplies.sales from archive 'sampledata.archive'
2025-10-16T10:03:13.088+0000    reading metadata for sample_geospatial.shipwrecks from archive 'sampledata.archive'
2025-10-16T10:03:13.089+0000    reading metadata for sample_training.posts from archive 'sampledata.archive'
2025-10-16T10:03:13.089+0000    reading metadata for sample_training.routes from archive 'sampledata.archive'
2025-10-16T10:03:13.089+0000    reading metadata for sample_mflix.movies from archive 'sampledata.archive'
2025-10-16T10:03:13.089+0000    reading metadata for sample_mflix.embedded_movies from archive 'sampledata.archive'
2025-10-16T10:03:13.089+0000    reading metadata for sample_restaurants.neighborhoods from archive 'sampledata.archive
'
```

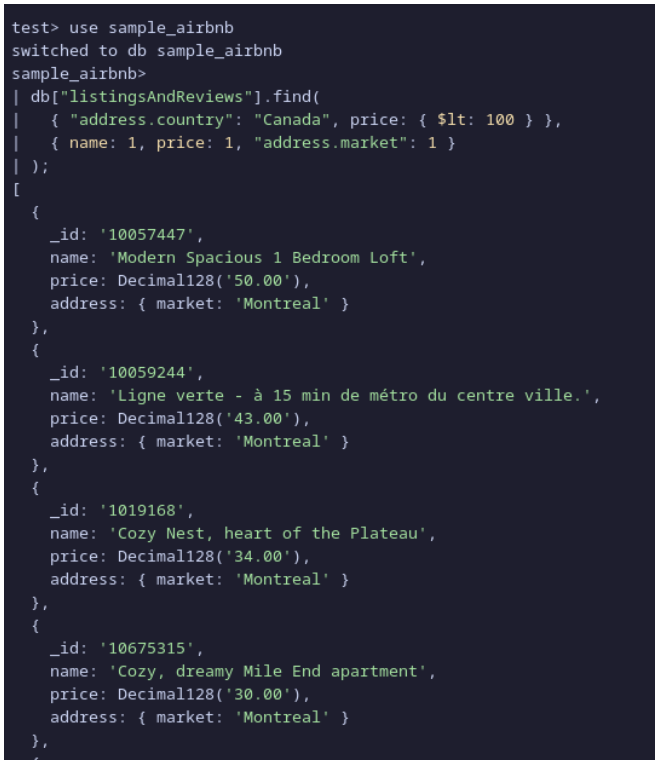


SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

Name: Thanawin Pattanaphol **Date: 01324096**

Part B: Basic CRUD Operations

Query	Results

`db["listingsAndReviews"].find(`
`  { "address.country": "Canada", price: { $lt: 100 } },`
`  { name: 1, price: 1, "address.market": 1 }`
`);`
	

```
test> use sample_airbnb
switched to db sample_airbnb
sample_airbnb>
| db["listingsAndReviews"].find(
|   { "address.country": "Canada", price: { $lt: 100 } },
|   { name: 1, price: 1, "address.market": 1 }
| );
[
  {
    _id: '10057447',
    name: 'Modern Spacious 1 Bedroom Loft',
    price: Decimal128('50.00'),
    address: { market: 'Montreal' }
  },
  {
    _id: '10059244',
    name: 'Ligne verte - à 15 min de métro du centre ville.',
    price: Decimal128('43.00'),
    address: { market: 'Montreal' }
  },
  {
    _id: '1019168',
    name: 'Cozy Nest, heart of the Plateau',
    price: Decimal128('34.00'),
    address: { market: 'Montreal' }
  },
  {
    _id: '10675315',
    name: 'Cozy, dreamy Mile End apartment',
    price: Decimal128('30.00'),
    address: { market: 'Montreal' }
  },
```

SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

Name: Thanawin Pattanaphol **Date: 01324096**

```
db["listingsAndReviews"].find(
  { amenities: { $in: ["Pets allowed"] } },
  { name: 1, amenities: 1 }
);
```

```
sample_airbnb> db["listingsAndReviews"].find(
|   { amenities: { $in: ["Pets allowed"] } },
|   { name: 1, amenities: 1 }
| );
[
  {
    _id: '10006546',
    name: 'Ribeira Charming Duplex',
    amenities: [
      'TV',
      'Cable TV',
      'Wifi',
      'Kitchen',
      'Paid parking off premises',
      'Smoking allowed',
      'Pets allowed',
      'Buzzer/wireless intercom',
      'Heating',
      'Family/kid friendly',
      'Washer',
      'First aid kit',
      'Fire extinguisher',
      'Essentials',
      'Hangers',
      'Hair dryer',
      'Iron',
      'Pack 'n Play/travel crib',
      'Room-darkening shades',
      'Hot water',
      'Bed linens',
      'Extra pillows and blankets',
      'Microwave',
      'Coffee maker',
      'Refrigerator',
      'Dishwasher',
      'Dishes and silverware',
      'Cooking basics',
      'Oven',
      'Stove',
      'Cleaning before checkout',
      'Waterfront'
    ]
  },
```

Limited to only one result for visual purposes.

```
db["listingsAndReviews"].updateOne(
  { name: "Beautiful Beach Condo" },
  { $set: { price: 120 } }
);
```

```
sample_airbnb> db["listingsAndReviews"].updateOne(
|   { name: "Beautiful Beach Condo" },
|   { $set: { price: 120 } }
| );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
sample_airbnb>
```

SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

Name: Thanawin Pattanaphol **Date: 01324096**

```
db["listingsAndReviews"].insertOne({
  name: "Student Test Listing",
  room_type: "Entire home/apt",
  price: 99,
  address: { country: "Thailand",
market: "Bangkok" },
  amenities: ["WiFi", "Air
conditioning"],
  reviews: []
});
```

```
sample_airbnb> db["listingsAndReviews"].insertOne({
|   name: "Student Test Listing",
|   room_type: "Entire home/apt",
|   price: 99,
|   address: { country: "Thailand", market: "Bangkok" },
|   amenities: ["WiFi", "Air conditioning"],
|   reviews: []
| });
{
  acknowledged: true,
  insertedId: ObjectId('68f0c70924cc7e5711273ba3')
}
sample_airbnb>
```

```
db["listingsAndReviews"].deleteOne({ nam
e: "Student Test Listing" });
```

```
sample_airbnb>
| db["listingsAndReviews"].deleteOne({ name: "Student Test Listing" });
{ acknowledged: true, deletedCount: 1 }
sample_airbnb>
```

Part C: Aggregation Policies

Query	Results

```
db["listingsAndReviews"].aggregate([
  { $match: { "address.country":
"Canada" } },
  { $group: { _id: "$room_type",
avgPrice: { $avg: "$price" } } }
]);
```

```
sample_airbnb> db["listingsAndReviews"].aggregate([
|   { $match: { "address.country": "Canada" } },
|   { $group: { _id: "$room_type", avgPrice: { $avg: "$price" } } }
| ]);
[
  {
    _id: 'Private room',
    avgPrice: Decimal128('61.30158730158730158730158730158730')
  },
  { _id: 'Shared room', avgPrice: Decimal128('154.60') },
  {
    _id: 'Entire home/apt',
    avgPrice: Decimal128('115.64175824175824175824175824175824')
  }
]
sample_airbnb>
```

SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

Name: Thanawin Pattanaphol **Date: 01324096**

```
db["listingsAndReviews"].aggregate([
  { $project: { name: 1,
number_of_reviews: 1 } },
  { $sort: { number_of_reviews: -1 } },
  { $limit: 5 }
]);
```

```
sample_airbnb> db["listingsAndReviews"].aggregate([
|   { $project: { name: 1, number_of_reviews: 1 } },
|   { $sort: { number_of_reviews: -1 } },
|   { $limit: 5 }
| ]);
[
  {
    _id: '4069429',
    name: '#Private Studio - Waikiki Dream',
    number_of_reviews: 533
  },
  {
    _id: '12954762',
    name: 'Near Airport private room, 2 bedroom granny flat**',
    number_of_reviews: 469
  },
  {
    _id: '95560',
    name: 'La Sagrada Familia (and metro) 4 blocks!',
    number_of_reviews: 463
  },
  {
    _id: '476983',
    name: 'PRIVATE Room in Spacious, Quiet Apt',
    number_of_reviews: 420
  },
  {
    _id: '5283892',
    name: 'traditional and Charming room',
    number_of_reviews: 408
  }
]
sample_airbnb>
```

```
db["listingsAndReviews"].aggregate([
  { $group: { _id: "$address.country",
count: { $sum: 1 } } },
  { $sort: { count: -1 } }
]);
```

```
sample_airbnb>
| db["listingsAndReviews"].aggregate([
|   { $group: { _id: "$address.country", count: { $sum: 1 } } },
|   { $sort: { count: -1 } }
| ]);
[
  { _id: 'United States', count: 1222 },
  { _id: 'Turkey', count: 661 },
  { _id: 'Canada', count: 649 },
  { _id: 'Spain', count: 633 },
  { _id: 'Australia', count: 610 },
  { _id: 'Brazil', count: 606 },
  { _id: 'Hong Kong', count: 600 },
  { _id: 'Portugal', count: 555 },
  { _id: 'China', count: 19 }
]
sample_airbnb>
```

SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

Name: Thanawin Pattanaphol **Date: 01324096**

Part D: Design & Reflection Challenge

1. If you were to convert listingsAndReviews into relational tables, what tables would you create?

Converting listingsAndReviews into a relational tables would require creating many tables, such as, room_types, bed_types, amenities, addresses, images, hosts, reviews and so on.

Primary keys (and possible foreign keys) would be ones like room_id, image_id, review_id, and so on. They keep tables clean and tidy while still providing the same functionality.

Example CREATE TABLE queries:

```
CREATE TABLE host (
      host_id INT NOT NULL,
      host_url TEXT NOT NULL,
      host_name VARCHAR(100) NOT NULL,
      host_location VARCHAR(150) NOT
NULL,
      host_about VARCHAR(255),
      host_response_time VARCHAR(50) NOT
NULL,
      host_thumbnail_url TEXT,
      host_picture_url TEXT,
      host_neighbourhood VARCHAR(100),
      host_response_rate INT NOT NULL,
      host_is_superhost BOOLEAN NOT
NULL,
      host_has_profile_pic BOOLEAN,
      host_identity_verified BOOLEAN,
      host_listings_count INT,
      host_total_listings_count INT,
      host_verifications INT FOREIGN KEY
REFERENCES
host_verify_details(host_verifications),
      PRIMARY KEY (host_id),
);
```

```
CREATE TABLE reviews (
      review_id INT NOT NULL,
      date TIMESTAMP NOT NULL,
      listing_id VARCHAR(10) NOT NULL,
      reviewer_id VARCHAR(10) NOT NULL,
      reviewer_name VARCHAR(100) NOT
NULL,
      comments TEXT,
);
```

2.

a) What kind of data (like reviews, amenities, or address) is easier to store as embedded documents?

JSON-like data is the easiest to store as embedded documents or specifically, data that is related and requires fast reading speeds and ones that require flexible structures or schemas.

SEN-209: Designing And Implementing Databases
Lab 8: NoSQL

Name: Thanawin Pattanaphol **Date: 01324096**

b) What data would benefit from normalization in SQL?

Data such as, addresses, amenities, will benefit from SQL normalization as it reduces the amount of anomalies in the data and especially redundancy as it eliminates the need to store duplicates of the same data for each record.

Part E – Reflection

1. What advantages did you notice using MongoDB locally compared to SQL?

   There were several advantages that were clear when compared to SQL, especially the way MongoDB stores its data. MongoDB stores its data in a JSON-like format, in which, is more appealing towards new users, especially those who work with TypeScript / JavaScript or JSON-related tasks. This also applies to the query syntax in which it resembles in that of a function and a JSON input. Another significant benefit that is related to the data storage format is the fact that it is easier to explore, search or navigate through the data, thanks to the JSON format, it is relatively simple to read and understand, but along with how MongoDB is very fast when reading and writing queries; very beneficial when working with a huge amount of data, commonly referred to as “Big Data”.

2. What challenges or errors did you encounter during setup or data import?

   During the setup process, there were certainly many several challenges or errors that came up. The MongoDB website that provided the installation link for the MongoDB Community Edition contained a specific section that discusses MongoDB Atlas, which was a fully managed version of MongoDB, this caused some confusion during the downloading process, along with other issues, such as, installing MongoDB Community Edition using different package managers e.g. brew, apt-get, as the brew-based installation were not working when installed on an Ubuntu Linux system while the native apt-get one did.

   This also applied to the MongoDB Command Line Database Tools as the brew based package wasn’t working fully on Linux and therefore had to switch to the classic installation process of tarball extraction and setting the binary directory in the environment variables.

3. How does schema flexibility help or hurt when dealing with complex datasets like Airbnb’s?

   In complex datasets, such as, Airbnb’s dataset, schema flexibility offers several pros and cons. There were several advantages when it comes to MongoDB’s flexible schema format, it enabled Airbnb to store different sets of data for each write query which eliminates the need for there to be a preset schema and that the schema does

not have to be re-created in multiple iterations, this leads to a more diverse set of data that can be stored, more information that are unique to a specific record can be stored, and many other benefits.

However, there are also several disadvantages from a flexible schema, since each write query can contain different data in each time, this can lead to an increased number of anomalies, incorrect and inconsistent data. This can potentially negatively impact data consistency in complex datasets, such as Airbnb's. Another disadvantage is that, due to the data being stored documents, this will lead to multiple copies or duplicates of data being stored, leading to less efficiency and increased storage.