UNIVERSITY OF MALAYA

MASTER OF DATA SCIENCE (SEMESTER 2 – 2022/2023)

FACULTY OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY

WQD7007 BIG DATA MANAGEMENT

GROUP PROJECT

Performance Comparison of Databases for Optimized Search in the Hospitality Industry

GROUP MEMBERS	
S2118013	GUI MIOW WAN
S2145827	KHAR SHIN YIN
17123313	LEOW JUN SHOU
17082915	LIM POH SZE
S2102170	SIM LIN ZHENG
S2148268	WONG ANN LI
S2136594	VIRGIL SIU TUNG CHUNG
S2121801	YU YUEN HERN

Table of Contents

List of Figures

List of Tables

1.0 Introduction

In this Big Data era, most industries, such as fintech, hospitality, telecommunications, and e-commerce, are struggling with the challenges of 5V's: volume, velocity, variety, veracity, and value. The surge in data in the hospitality industry is vital due to the emergence of customer interactions, online travel platforms, and digital booking processes (Gupta *et al*., 2017). One of the common issues identified is the slow performance of website loading time and searching time which leads to a service gap that might affect customers' experience when surfing the website (Stringam & Gerdes, 2019). The service gap is always filled by presenting system feedback or ads to keep users' attention and prevent them from clicking 'reload'.

Although a proper data pipeline can be set up properly – enabling the users to key in the information they are looking for in the search bar with mechanisms in place to extract the required data from the database and return the output to the user, the query execution time might not meet the customer expectation, and there is a room of improvement for the performance. There are several reasons causing slow website load or search, for example, poor server performance, unsuitable server location, heavy traffic, or inefficiently designed database. It was claimed that an inefficiently designed database might affect the query and search performance, and a good database strategy can result in better retrieval effectiveness on the website (Powell *et al*., 2000). To provide a better experience for customers when browsing the website for searching, booking, or payments, it is important to evaluate the root of the data pipeline – the database. It is crucial to evaluate the appropriateness of the database to be implemented in this specific use case or if there are other suitable databases by studying their features, performance, and capabilities.

Therefore, the aim goal of this study is to understand the performance of existing big data storage technologies on hospitality data, which translates to the query execution duration by comparing different suggested databases. This can be conducted by measuring the execution duration of different running queries, analysing and comparing the results. The next section will address the problem statement and literature review in a thorough manner, alongside the dataset description and justification of the selected database technologies. The database technologies will be compared based on several evaluation metrics, such as database storage size, data ingestion time, and query response time. The conclusion will be drawn based on the summarized findings.

1.1 Literature Review

There are several studies performed about how different databases perform when it comes to queries. Based on a study led by Hinai & Hamed (2016), the majority of NoSQL systems, such as MongoDB, Redis, and Cassandra, can execute fast queries at the expense of not being capable of retrieving the latest changes in data. Thus, these NoSQL systems compromise consistency for scalability. Relational systems are pitted against non-relational databases to evaluate their performance based on the time required to load. Despite NoSQL performing better than relational databases, the best option still lies within the features of the data, query type, and specific system used.

Ceresnak & Kvet (2019) emphasized the importance of the time taken aspect and storage size demands aspect when evaluating the performance comparison. The study was

carried out in 2 different stages; the first evaluation model is based on examining the speed of loading operation using a small data portion, while the second evaluation model is about storing a huge number of records. Since the data size storage matters, using non-relational databases is more effective when it comes to large datasets since non-relational databases store data in flat collections form.

Jose & Abraham (2020) observed the endeavor of migration from conventional database systems to NoSQL databases in big data management within commercial organizations currently. Speaking realistically, (Damodaran B et al., 2016) points out that the ALTER TABLE command to add a single new field to Craigslist's MySQL database required months to execute. Consequently, the Craigslist team has migrated their system to MongoDB to accommodate changes to the data model with cost efficiency in mind.

1.2 Problem Statement

In the hospitality industry, the volume of data has witnessed significant growth due to the digitalization of hotel booking processes and the rise of online travel platforms. As a result, hotel booking engines face a critical challenge in efficiently handling the increased data load, particularly when it comes to search functions. The read-write speed of databases directly impacts the performance of search queries, and a slowdown can negatively impact user experience, leading to decreased customer satisfaction and potential revenue loss.

Although there were no studies on the direct impact of database efficiency on revenues, it is widely known that database efficiency contributes to the overall user experience of the platform. In fact, Nah (2004) found that the tolerable waiting time for information retrieval is approximately two seconds. Therefore, database efficiency is imperative for the retention of potential customers on online travel platforms.

1.3 Objective

The objective of the study is to evaluate the performance of various Hadoop technologies that can be used to store massive amounts of data generated by Airbnb and assess the performance of simple queries conducted.

1.4 Dataset Description

The Airbnb short-term rental listings and reviews structured datasets were used in this project which was obtained from Kaggle. These datasets in CSV format were initially separated by the 19 United States (US) states Alpha Codes (e.g. CA, FL, etc.). These datasets are then merged to form the two datasets for the study, namely **merged_listings** and **merged_reviews**. The datasets can be accessed from https://www.kaggle.com/datasets/konradb/inside-airbnb-usa.

The **merged_listings** dataset contains 75 attributes and 261,538 rows of data. The attributes comprise numeric and textual data which can generally be summarised to include information such as metadata of the Airbnb website scrapping, details of the hosts and their respective listings, geographical coordinates of the properties, number of rooms, amenities, prices, maximum and minimum days of stay, days of availability, reviews scores, and instant booking availability.

The **merged_reviews** dataset contains 6 attributes and 865,503 rows of data. The attributes comprise numeric and textual data, which include the listing ID, host ID, date, reviewer's ID, reviewer's name, and detailed comments on the listed properties.

Table 1:Dataset Description

File Name	File Size (Mb)	No. of Attributes	No. of Records
merged_listings.csv	681	75	261,538
merged_reviews.csv	2910	6	10,459,999

Please refer to Appendix A and B for the details of the dataset description.

2.0 Methodology

The study evaluates the performance of various Hadoop components, including Hive, HBase, MySQL and MongoDB.

First, on the Ubuntu instance, Hadoop and HDFS are installed and set up. This includes establishing the required configuration files and making sure the Hadoop cluster is operating successfully. Then, various database system components, including Hive, HBase, MySQL, and MongoDB, are installed and set up on the Ubuntu instance. To facilitate smooth operations, integration with the Hadoop environment is stipulated.

Data is imported from a CSV file into each database system after the database system has been configured. In this stage, the CSV file is converted into a database-friendly structure, and the data is loaded into the tables or collections. Through the execution of queries constructed based on real-world scenarios and workload patterns, the performance of the database system is measured. Database storage size, data ingestion time, and query response time are among the metrics that are monitored. The generated performance data is then carefully examined and contrasted. The performance of various databases (Hive, HBase, MySQL, and MongoDB) is compared using the selected evaluation metrics.

The conclusions drawn from the performance assessments and comparison evaluations will help decision-makers make informed decisions about performance optimization and enhancement approaches.

3.0 Technology Justification

3.1 Hive

Apache Hive is a data warehouse system that performs in a distributed and fault-tolerant approach, enabling the analysis of enormous volumes of data. It acts as a central repository of data, enabling the use of data in decision-making. Users may easily manipulate petabytes of data using SQL queries using Hive. (*What Is Apache Hive?*, 2023)

- **Scalability**: Hive and the Apache Hadoop ecosystem are deeply interconnected. It makes use of the underlying Hadoop architecture for data processing and storage, enabling clients to benefit from Hadoop's scalability and fault tolerance. Hive can be scaled and distributed effortlessly in accordance with requirements. (*What Is Apache Hive?*, 2023)

- **Fault tolerance**: Terabytes to petabytes of data can be managed with ease by this platform. Hive is designed to be fault-tolerant, giving great availability and data reliability. Without losing data or interfering with ongoing activities, it can gracefully tolerate faults and recover from them. (*What Is Apache Hive?*, 2023)
- **SQL-like interface**: Hive features a user-friendly interface that is identical to SQL, known as HiveQL, allowing those who already possess proficiency in SQL to work with Hive and make use of their prior knowledge. (*What Is Apache Hive?*, 2023)

3.2 HBase

HBase is a column-oriented non-relational database management system that builds on top of the Hadoop Distributed File System (HDFS). The data is stored in individual columns and indexed by a unique row key, following a 4-dimensional model: row key (a unique identifier for an entity), column family (group of columns), column (value identifier) and timestamp (version number for the updates to values in a column).

- **Scalable:** HBase can easily **scale horizontally** by adding more slave nodes to the cluster. As more nodes are added, data is automatically distributed across these nodes, allowing for more efficient resource utilization and improved storage capacity. HBase's distributed architecture enables it to utilise the collective power of the entire cluster to manage **enormous volumes** of **data**, ranging from terabytes to petabytes (Skokov, 2019).
- **Low latency:** HBase distributes requests from applications over a cluster of machines to enable low-latency random read and write access to petabytes of data. HBase stores tables using a multidimensional sparse map with rows and columns, allowing for **random real-time read** and **write access**. With the use of Java client APIs, tables in HBase can be used directly as targets for MapReduce jobs (Kailas, 2019).
- **Fault-tolerance:** HBase splits data **stored** in **tables** across **multiple hosts** in the cluster and is built to withstand individual host failures. ZooKeeper, an open-source Apache project, provides fault-tolerance capabilities for Hbase, which tracks partial failures in the database. Zookeepers are able to track failures and network partitions with the presence of the RegionServer. When RegionServer fails, the Master node recovers and redistributes the affected regions to other available RegionServer in the cluster (Kailas, 2019).

3.3 MongoDB

MongoDB is one of the famous NoSQL databases used in Big Data technology. MongoDB is chosen as one of the databases because it is proven that MongoDB **performs better in select query operations, and the total time taken to retrieve data is less** compared to non-relational databases based on the previously conducted findings (Patil *et al.*, 2017; Sirish & Akshay, 2019). Besides that, MongoDB is **better in terms of latency and database size** (Eyada *et al.*, 2020). On top of that, MongoDB is chosen because of the characteristics below:

- MongoDB is an open-source database that provides **high performance**. It supports embedded data models and reduces I/O activity on database systems, indexes support

faster queries and can include keys from embedded documents and arrays that reduce expensive joins (Chauhan, 2019).

- MongoDB is **flexible and scalable**. It allows to store and manage structured data, semi-structured data, and unstructured data. These data can be distributed horizontally with scalability across multiple serves. Performance will not be sacrificed if the volume of data increases.
- MongoDB is a **document database** that uses a document-oriented approach where one collection can hold different documents – different numbers of fields and content from one document to another (Chauhan, 2019).
- MongoDB is **high availability** as it has built-in replication that provides automatic failover and data redundancy. This feature allows the database to be created in multiple copies and keeps them synchronized (Chauhan, 2019).

3.4 MySQL

MySQL has options for relational and non-relational database systems. MySQL relational database is structured in the form of tables that consists of columns; as for document-based, MySQL saves data as documents. This provides the flexibility for developers to build applications that leverage traditional SQL relational databases as well as no-SQL schema-free document-based databases (Gyorodi, 2021).

- MySQL offers **Performance Optimization** methods to optimize query performance, especially in big data analytics. The optimization methods such as caching to improve the speed of query execution and adexed-based optimization help to reduce the query time (Jose & Abraham, 2020).
- As MySQL is an open-source database system, therefore it is **cost-effective** for organization to leverage MySQL for big data projects as no-cost to use MySQL. Open-source MySQL is also supported by a large community in the industries that would speed up product enhancements (Gyorodi, 2021).
- From a **Data Security** perspective, MySQL comes with security features such as authentication, access control, secure connection, and encryption to protect data integrity and sensitivity. MySQL also comes with auditing and logging capabilities that allow the organization to track and trace data activities (Maayan, 2022).
- **High scalability** is one of the advantages of using MySQL. MySQL offers high availability solutions to achieve redundancy, automatic failover and data synchronization. Other capabilities include replication, partitioning, and sharding, which enable the system to handle high loads with more requests and provide high availability (Jose & Abraham, 2020).

4.0 Results

The four different technologies, including HIVE, HBase, MySQL, and MongoDB, is compared using different evaluation metrics: performance, ease of use, and storage utilisation.

4.1 Performance

Table 2: Performance of each Storage-based Technology

Task	Hive	HBase	MongoDB	MySQL
	Time Taken (seconds)			
Data Import				
Import merged_listings	12.577	52.313	86.883	151.880
Import merged_reviews	66.376	140.440	1139.104	509.660
Total time taken (seconds)	78.953	192.753	1225.987	661.540
Query: Reviews				
Top 10 listings with the most reviews	119	-	11.417	86.100
Top 10 listings with the most reviews and recently commented	133.808	-	17.333	71.090
Top 10 reviewers with the most reviews	118.524	-	415.977	90.420
Top 10 reviewers with the most reviews and recently commented	146.545	-	413.139	96.450
10 reviewer named 'Jessie' after the company have listed more than 30m Airbnb	121.254	-	0.461	12.620
Total time taken (seconds)	639.131	-	858.327	356.680
Query: Listings				
Top 10 cheapest superhost hotels with the best rating and fastest response time and rate	57.647	-	3.153	3.170
Top 10 cheapest superhost hotel with the best rating and accommodates 10 person, and are instantly bookable	30.921	-	1.676	0.630
Total count of superhost	42.897	21 mins (scan)	3.199	0.650
Top 10 neighbourhood group locations with count	56.246	-	0.420	0.720
Top 10 host locations with count	60.340		2.478	1.140
Top 5 property types and accommodates with count	56.794	-	0.980	1.220
10 hosts who are located in Tijuana, Baja California, Mexico and have a 'host_total_listings_count' of 3	2.519	0.350	0.641	0.180
Total time taken (seconds)	307.364	-	10.069	7.710

Table 2 shows the performance of data import and queries for Hive, HBase, MongoDB and MySQL. Please refer to Appendix C, D, E and F for further details about the implementation result in detail. Figures 1 – 3 demonstrate the performance results in visualization.

Figure 1: Performance of Data Import to each Storage-based Technology

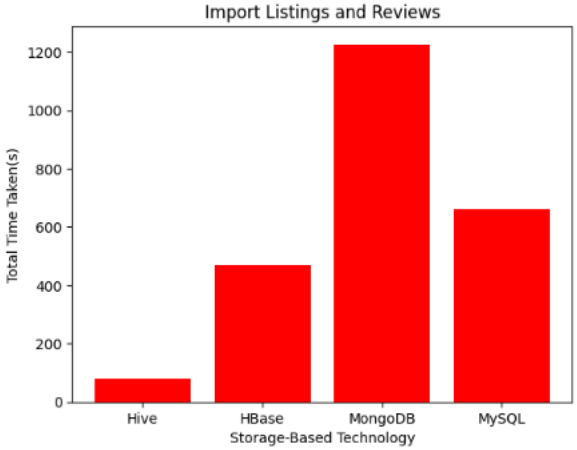


Figure 2: Performance of Reviews Queries for each Storage-based Technology

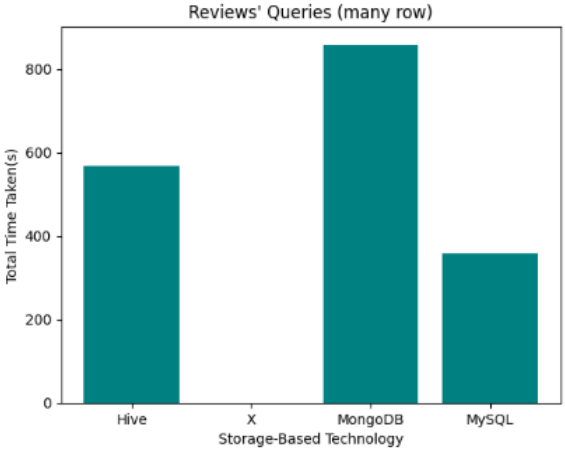
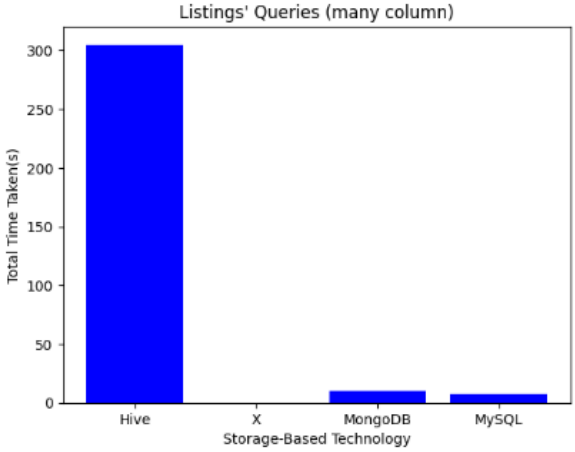


Figure 3: Performance of Listings Queries for each Storage-based Technology



According to the comparison, Hive exhibits the shortest time during data import while MongoDB takes the longest time among the four compared technology. In the context of query performance, HBase is not considered due to its limitation, which will be discussed in the Hbase Justification section. Among the three compared technology, MySQL, in overall, demonstrates the best performance in different queries.

HBase Justifications

HBase has certain limitations when it comes to executing complex queries. In this project, we found 2 limitations when accessing the HBASE performance. First, HBase does not have a SQL-like query language. Instead, we need to interact with HBase by using APIs such as Java API or the HBase Shell. This means that complex query capabilities, such as joins or aggregations, are not available out-of-the-box in Hbase (Apache HBase Team, 2023).

However, when we are using HBase Shell, we found a second limitation, which is that HBase primarily relies on scanning the data based on row keys or predefined filters. The scan does not have the capability of SQL-alike functions such as ORDER, CAST, GROUP, and aggregation functions which makes most of the query inaccessible (Cloudera, 2023). We use the scan to query the Total count of superhost in the Listings Table and obtain 21 minutes, proving that scan cannot compete with the capabilities of the count. In conclude, only scan in HBase Shell can filter and display the records, but unable to perform complex queries.

4.2 Ease of Use

Table 3 describes the ease of use for each storage-based technology from four factors, including the query language, data model, scalability, and configuration.

Table 3: Comparison between the ease of use for each Storage-based Technology

	Query Language	**Data Model**	**Scalability**	**Configuration**
Hive	Offers an **SQL-like interface**, making it understandable to users with prior SQL familiarity.	Compatible with **structured** data and supports a tabular data model with schema-on-read.	**Scalable** for large data analytics since it leverages Hadoop's distributed processing capabilities.	The **Hadoop** ecosystem's components must be set up and configured, which may require additional administrative effort.
HBase	Uses **HBase Shell.** Simple command-line interface to communicate with HBase. CRUD (Create, Read, Update, Delete) operations perform.	Compatible with **structured** data, it stores data in a table with rows and columns. Each table consists of **primary key and column families** (which group together all the columns in the database).	Highly **scalable** by partitioning tables into regions and distributing across several region servers.	Requires more effort in setting up the **Hadoop** ecosystem components and HBase configuration.
MongoDB	Very different from SQL and HQL; thus, it could be a learning curve if transitioning	Not compatible with structured data and therefore does not support tabular data	Has built-in **scalability**.	Minimal effort. The setup only requires the installation of **GnuPG** and a configuration file

	from MySQL and Hive. The query language seems like **object-oriented and JSON-based**.	model. The data are kept in a **document format**.		before MongoDB can be installed. MongoDB can be started right after installation.
MySQL	MySQL uses using a standardized query language that called **Structured Query Language SQL** to manage relational databases. Easy to pick up with users who have familiarity with SQL syntax.	MySQL supports relational data models. MySQL also supports **Object-relational, Spatial, JSON and XML** data models.	MySQL comes with built-in features to optimize read/write **scalability**. MySQL supports clustered deployment, sharding techniques, caching, and partitioning.	Minimal effort, easy to set up. The setup can use Command-Line Interface (CLI) to download the package and install **MySQL** default configuration and database engine.

4.3 Storage Utilisation

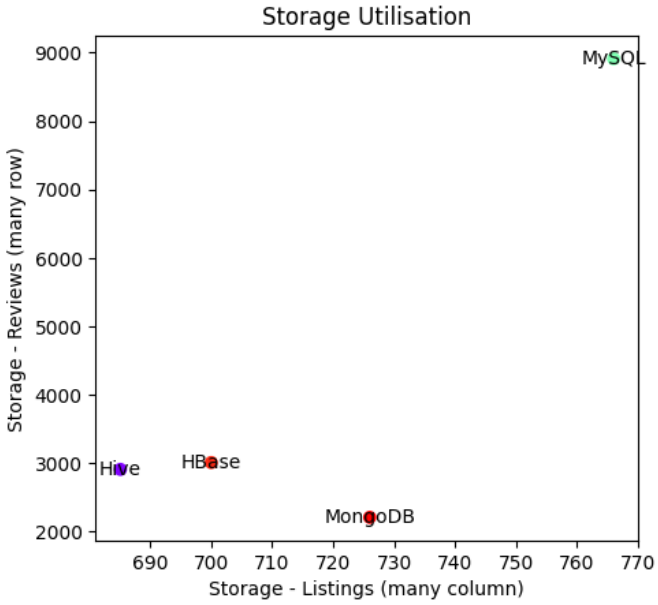
Table 4 compares the original file size of the 2 datasets, *merged_listings.csv* and *merged_reviews.csv*, with the storage utilisation for Hive, HBase, MongoDB and MySQL.

Table 4: Storage Utilisation of Each Storage-based Technology

File Name	Original File (MB)	Hive Table (MB)	HBase Table (MB)	MongoDB Table (MB)	MySQL Table (MB)
merged_listings.csv	681.0	685.1	700.0	726.0	766.0
merged_reviews.csv	2910.0	2910.0	3010.0	2209.0	8911.0

Figure 4 visualizes the comparison in a scatter chart. The results show that MySQL has the largest storage size among the four storage-based technologies.

Figure 4: Scatter chart of Storage Utilisation Comparison



5.0 Conclusion

In conclusion, **Hive has the greatest performance in importing** both CSV files as it has the shortest import times. Based on the performance of the queries for both files, only HBase has the limitation in computing the complex query (other than CRUD operations). If complex queries are executed frequently, **HBase is not a good choice for computing such queries**. Among the 12 queries for both CSV files, **MongoDB and MySQL have excellent performance**, which each of the technology handling five queries in the fastest execution time (two in reviews, three in listings). However, MongoDB has a performance issue in executing certain complex queries (which are the third and fourth queries in reviews); this might be due to query optimization; creating a compound index can speed up the query performance when multiple conditions are needed to search (MongoDB, 2023). Based on the overall query performance, **MySQL has the best performance in computing the queries**.

Hive and MySQL offer SQL-like interface, which is familiar with computation. HBase uses a simple command-line interface for CRUD operations, limiting complex query execution. MongoDB sends data in JSON-like format, which does not support structured data (Salva, 2021).

Based on the storage utilization, the size of the table created in each technology is shown. **MySQL requires the most significant storage space** to store both the CSV files data. Besides that, Hive takes approximate storage space to store the data as the original imported file.

Lastly, **MySQL** is suggested to be used due to the fast performance of query computation and ease to use. However, MySQL requires large storage space to store the table created from the original data. Since users are more concerned with the query speed, thus the large storage space limitation in MySQL can be overcome as big companies able to pay for the storage.

6.0 References

Apache HBase Team. (2023). Apache HBase TM reference guide. https://hbase.apache.org/book.html#arch.overview.nosql

Čerešňák, R., & Kvet, M. (2019). Comparison of query performance in relational a non-relation databases. *Transportation Research Procedia, 40,* 170–177. https://doi.org/10.1016/j.trpro.2019.07.027

Chauhan, A. (2019). A review on various aspects of MongoDB databases. *International Journal of Engineering Research & Technology (IJERT),* 8(05), 90-92. https://www.ijert.org/a-review-on-various-aspects-of-mongodb-databases

Damodaran B, D., Salim, S., & Vargese, S. M. (2016). Performance Evaluation of MySQL and MongoDB Databases. International Journal on Cybernetics & Informatics, 5(2), 387–394. https://doi.org/10.5121/ijci.2016.5241

Eyada, M.M., Saber, W., El Genidy, M.M., & Amer, F.A. (2020). Performance Evaluation of IoT Data Management Using MongoDB Versus MySQL Databases in Different Cloud Environments. *IEEE Access,* 8, 110656–110668. https://doi.org/10.1109/ACCESS.2020.3002164

Fiona Fui-Hoon Nah (2004). A study on tolerable waiting time: how long are Web users willing to wait?, Behaviour & Information Technology, 23:3, 153–163, DOI: 10.1080/01449290410001669914

Gupta, K., Gauba, T., & Jain, S. (2017). Big data in hospitality industry: A survey. *International Research Journal of Engineering and Technology*, 4(11), 476–479. https://www.academia.edu/35353958/Big_Data_In_Hospitality_Industry_A_survey

Gyorodi C.A, Dumşe-Burescu DV., Győrödi R.Ş, Zmaranda D.R, Bandici L, Popescu D.E (2021). *Performance Impact of Optimization Methods on MySQL Document-Based and Relational Databases.* Applied Sciences. 11(15):6794. https://doi.org/10.3390/app11156794

HBase filtering. (2023). https://docs.cloudera.com/runtime/7.2.10/managing-hbase/topics/hbase-filtering.html

Hinai, A., & Hamed, A. S. A. (2016). A Performance Comparison of SQL and NoSQL Databases for Large Scale Analysis of Persistent Logs. *Examensarbete*. http://uu.diva-portal.org/smash/record.jsf?pid=diva2:957015

Jose B. Abraham S.(2020). Performance analysis of NoSQL and relational databases with MongoDB and MySQL. *Materials Today: Proceedings.* Vol 24, 3,2036-2043. DOI: https://doi.org/10.1016/j.matpr.2020.03.634.

Kailas, G. (2019). A Comparison Of NoSQL Database Systems: A Study On MongoDB and Apache HBase. https://www.researchgate.net/publication/330937764

Maaya G. D. (2022). MySQL Security Best practises. *IEEE Computer Association*. https://www.computer.org/publications/tech-news/trends/mysql-security-best-practices

MongoDB (2023). Optimize Query Performance. MongoDB https://www.mongodb.com/docs/manual/tutorial/optimize-query-performance-with-indexes-and-projections/

Patil, M., Hanni, A., Tejeshwar, C.H., & Patil, P.M. (2017). A qualitative analysis of the performance of MongoDB vs MySQL database based on insertion and retriewal operations using a web/android application to explore load balancing — Sharding in MongoDB and its advantages. *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, 325–330. https://ieeexplore.ieee.org/document/8058365

Powell, A.L., French, J.C., Callan, J., Connell, M.E., & Viles, C.L. (2000). The impact of database selection on distributed searching. *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1–8. https://doi.org/10.1145/345508.345584

Salva, S. (2021). Handling JSON in MongoDB. Section. https://www.section.io/engineering-education/handling-json-in-mongodb/

Sirish, S.B. & Akshay, K.C. (2019). Performance Analysis of Queries in RDBMS vs NoSQL. 2019 *2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, p. 1, 1283–1286. https://doi.org/10.1109/ICICICT46008.2019.8993394

Skokov, I. (2019). Introduction to Apache HBase(part 1). *Medium*. https://medium.com/@lagrang09/introduction-to-apache-hbase-755b40cc0f30

Stringam, B., & Gerdes, J. (2019). Service gap in hotel website load performance. *International* Hospitality Review, 33(1), 16-29. https://www.emerald.com/insight/content/doi/10.1108/IHR-09-2018-0012/full/html

What is Apache Hive? (2023). Amazon Web Services, Inc. https://aws.amazon.com/big-data/what-is-hive/

APPENDIX A: Dataset description of *merged_listings.csv*

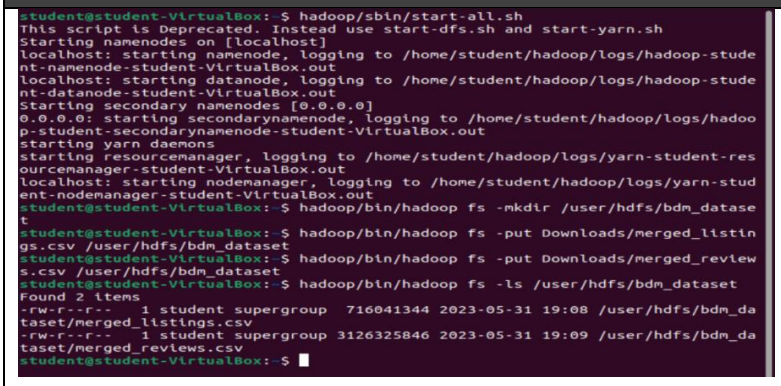
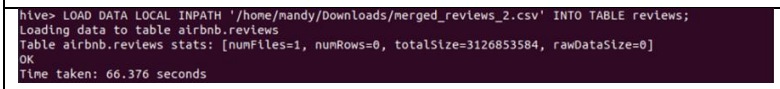
No	Columns	Descriptions
1	id	Unique ID of the listing
2	listing_url	URL of the listing
3	scrape_id	Unique ID of the web scrapping performed
4	last_scraped	Date of the web scrapping performed
5	name	Name of the listing
6	description	Description of the listing
7	neighborhood_overview	Description of the neighbourhood of the listing
8	picture_url	URL of the picture in the listing
9	host_id	Unique ID of the host
10	host_url	URL of the host
11	host_name	Name of the host
12	host_since	Start date of being a host
13	host_location	The area the host is currently based
14	host_about	Descriptions of the host
15	host_response_time	Response categories of the host
16	host_response_rate	The response rate of the host
17	host_acceptance_rate	The acceptance rate of the host
18	host_is_superhost	Airbnb Superhost status of the host
19	host_thumbnail_url	URL of the thumbnail of the listing
20	host_picture_url	URL of the picture of the host
21	host_neighbourhood	The neighbourhood of the host
22	host_listings_count	The number of listing the host has
23	host_total_listings_count	The total listing the host has
24	host_verifications	Communication options with the host
25	host_has_profile_pic	Availability of host's profile picture
26	host_identity_verified	Identity verification status of the host
27	neighbourhood	Neighbourhood of the listing
28	neighbourhood_cleansed	The specific neighbourhood of the listing
29	neighbourhood_group_cleansed	Categories of the neighbourhood of the listing
30	latitude	Geographical coordinates of the listing
31	longitude	Geographical coordinates of the listing
32	property_type	Type of property
33	room_type	Type of room the property has
34	accommodates	Number of persons the property can accommodate
35	bathrooms	Availability of bathrooms
36	bathrooms_text	Detailed number and type of bathrooms
37	bedrooms	Number of bedrooms

38	beds	Number of beds
39	amenities	Amenities available on the property
40	price	Price per night of the property
41	minimum_nights	Minimum nights
42	maximum_nights	Maximum nights
43	minimum_minimum_nights	Minimum minimum nights
44	maximum_minimum_nights	Maximum minimum nights
45	minimum_maximum_nights	Minimum maximum nights
46	maximum_maximum_nights	Maximum maximum nights
47	minimum_nights_avg_ntm	Average of minimum nights
48	maximum_nights_avg_ntm	Average of maximum nights
49	calendar_updated	No data
50	has_availability	Availability of the property
51	availability_30	Availability of up to 30 nights
52	availability_60	Availability of up to 60 nights
53	availability_90	Availability of up to 90 nights
54	availability_365	Availability of up to a year
55	calendar_last_scraped	Date of the web scrapping performed
56	number_of_reviews	Number of reviews on the listing
57	number_of_reviews_ltm	Number of reviews
58	number_of_reviews_l30d	Number of reviews within 130 days
59	first_review	Date of the first review
60	last_review	Date of the latest review
61	review_scores_rating	Review scores in terms of rating
62	review_scores_accuracy	Review scores in terms of accuracy
63	review_scores_cleanliness	Review scores in terms of cleanliness
64	review_scores_checkin	Review scores in terms of checking in
65	review_scores_communication	Review scores in terms of communication
66	review_scores_location	Review scores in terms of location
67	review_scores_value	Review scores in terms of value
68	license	License number
69	instant_bookable	Instant bookability
70	calculated_host_listings_count	Calculated field of the count of host's listings
71	calculated_host_listings_count_entire_homes	Calculated field of the count of entire homes in the host's listings
72	calculated_host_listings_count_private_rooms	Calculated field of the count of private rooms in the host's listings
73	calculated_host_listings_count_shared_rooms	Calculated field of the count of shared rooms in the host's listings
74	reviews_per_month	Number of reviews per month

APPENDIX B: Dataset description of *merged_reviews.csv*

No.	Attributes	Descriptions
1	listing_id	Unique ID of the listing
2	id	Unique ID of an unspecified attribute
3	date	Date of the review
4	reviewer_id	Unique ID of the reviewer
5	reviewer_name	Name of the reviewer
6	comments	Comments are given on the listing

APPENDIX C: Implementation in HIVE

Command Line	Narration
	• First, the Hadoop services are started using the start-all.sh • Under HDFS, the folder directory of /user/hdfs/bdm_dataset is created. • The merged_listings.csv and merged_reviews.csv files are uploaded into the /user/hdfs/bdm_dataset folder directory in HDFS.
	• Hive is initialized, and bdm_project database is used.
Reviews – Data Import	
	• A table named "reviews" is created in Hive with the following properties set as shown in the screenshot.
	• The merged_reviews.csv file is loaded into the "reviews" table.
Listings – Data Import	
	• A table named "listings" is created in Hive with the following properties set as shown in the screenshot.

```
hive> LOAD DATA LOCAL INPATH '/home/student/Downloads/merged_listings_2.csv' INTO TABLE listi
ngs;
Loading data to table airbnb.listings
Table airbnb.listings stats: [numFiles=1, numRows=0, totalSize=718380626, rawDataSize=0]
OK
Time taken: 12.577 seconds
```

- The merged_listings.csv file is loaded into the "listings" table.

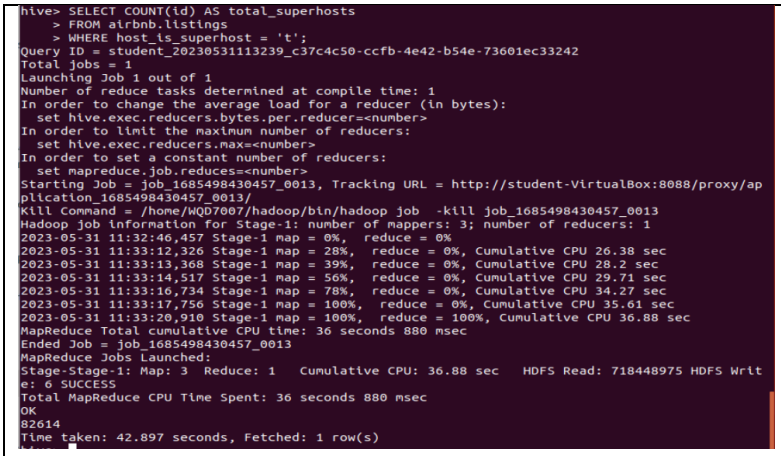
Listings – Queries

```
hive> SELECT id, name, price, review_scores_rating, host_response_time, host_response_rate
    > FROM listings
    > WHERE host_is_superhost = 't'
    > ORDER BY CAST(price AS DECIMAL) ASC, review_scores_rating DESC, host_response_time ASC,
 CAST(host_response_rate AS DECIMAL) DESC
    > LIMIT 10;
Query ID = student_20230531112925_cb1eb3fb-8e2e-45c1-8cc2-2dc4087c8f8f
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0011, Tracking URL = http://student-VirtualBox:8088/proxy/ap
plication_1685498430457_0011/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0011
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 1
2023-05-31 11:29:42,573 Stage-1 map = 0%,  reduce = 0%
2023-05-31 11:30:08,814 Stage-1 map = 22%,  reduce = 0%, Cumulative CPU 17.22 sec
2023-05-31 11:30:12,143 Stage-1 map = 39%,  reduce = 0%, Cumulative CPU 22.37 sec
2023-05-31 11:30:15,354 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 27.14 sec
2023-05-31 11:30:20,606 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 29.05 sec
MapReduce Total cumulative CPU time: 29 seconds 50 msec
Ended Job = job_1685498430457_0011
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 1   Cumulative CPU: 29.05 sec   HDFS Read: 718447246 HDFS Writ
e: 1016 SUCCESS
Total MapReduce CPU Time Spent: 29 seconds 50 msec
OK
49145147.0      Paradise awaits! C-203 Dual Master Suite No Fees         "Microwave"    8/5/2
021     within an hour  100%
46173360        420 Inside | Self Check-In | Sauna, Game Room, Gym       Amazon Prime Video"8
/11/2020        within an hour  100%
50854459.0      ☀Hot Tub & Yard - long term - 365 Day rental☀  $211.00 5.0
53309073.0      Garden Suite Minutes to Airport $70.00  5.0
51021962.0      Lovely 1BR condo w/ pool/spa, 4 min drive to beach       $145.00 5.0
22731814        Clean Mid-Century Modern in Long Beach CA       $58.00  5.0
49501235.0      尔湾光谱中心全新3房独栋别墅，24小时封闭社区，独立车库超大后院景观主卧，旅游度
假赴美生子最佳选择！     $300.00 5.0
45090236.0      Irvine with contemporary bedroom |Queen bed | WiFi       $50.00  5.0
45409025.0      Surf + stay beach townhome      $295.00 5.0
50013219.0      Beach Diamond Casita with AC,7 Blocks to Beach&Bay       $331.00 5.0
Time taken: 57.647 seconds, Fetched: 10 row(s)
```

- This is the Hive query for "Top 10 cheapest superhost hotels with the best rating and fastest response time and rate."

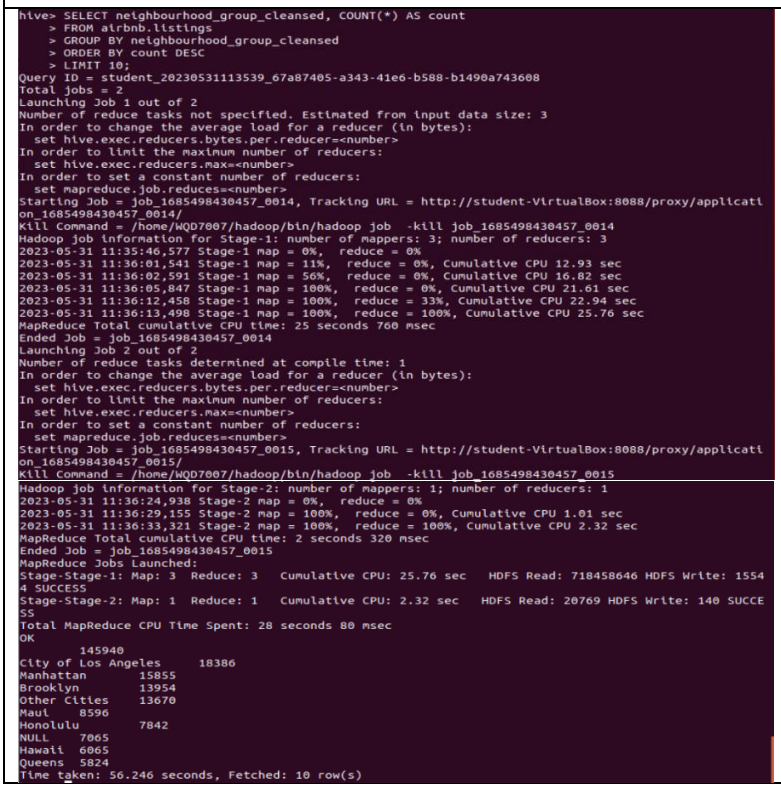
```
hive> SELECT id, name,price,review_scores_rating
    > FROM listings
    > WHERE host_is_superhost = 't' AND accommodates = 10 AND instant_bookable = 't'
    > ORDER BY CAST(price AS DECIMAL) ASC, review_scores_rating DESC
    > LIMIT 10;
Query ID = student_20230531113140_68441eae-7feb-4c37-aa28-7755a14714b7
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0012, Tracking URL = http://student-VirtualBox:8088/proxy/ap
plication_1685498430457_0012/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0012
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 1
2023-05-31 11:31:45,649 Stage-1 map = 0%,  reduce = 0%
2023-05-31 11:32:00,698 Stage-1 map = 39%,  reduce = 0%, Cumulative CPU 16.1 sec
2023-05-31 11:32:02,928 Stage-1 map = 56%,  reduce = 0%, Cumulative CPU 17.31 sec
2023-05-31 11:32:05,026 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 22.97 sec
2023-05-31 11:32:10,311 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 24.43 sec
MapReduce Total cumulative CPU time: 24 seconds 430 msec
Ended Job = job_1685498430457_0012
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 1   Cumulative CPU: 24.43 sec   HDFS Read: 718446986 HDFS Writ
e: 728 SUCCESS
Total MapReduce CPU Time Spent: 24 seconds 430 msec
OK
5.59666e+17     The Golden Glow Estate - socalholiday.com       $718.00 5.0
52995148.0      Beautiful & FULL of amenities for Kids & Adults!        $231.00 5.0
51214167.0      4BR Backyard Pool/Oasis -San Diego's best location      $693.00 5.0
50871025        Las Olas Designer retreat with a heated pool!   $513.00 5.0
50123097        Casa Marta -Hollywood Vacation Getaway  $332.00 5.0
42735396.0      Luxury villa w/private heated pool & hot tub, gas grill, game room, theater r
oom     $1,582.00       5.0
36671534.0      Modern Beach House Steps to Beach w/ Rooftop Deck       $1,036.00       5.0
48214776        Villa Tortuga | Pool & Jacuzzi | By the ocean   $380.00 5.0
46987415        RENT LUXE 5 STAR 4 BED + HTD POOL WALK TO BEACH!!       $399.00 5.0
47000374        RENT LUXE 5 STAR NEW 4 BED + HTD POOL STUNNING!!        $425.00 5.0
Time taken: 30.921 seconds, Fetched: 10 row(s)
```

- This is the Hive query: "Top 10 cheapest superhost hotels with the best rating accommodates 10 people and are instantly bookable."



```
hive> SELECT COUNT(id) AS total_superhosts
    > FROM airbnb.listings
    > WHERE host_is_superhost = 't';
Query ID = student_20230531113239_c37c4c50-ccfb-4e42-b54e-73601ec33242
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0013, Tracking URL = http://student-VirtualBox:8088/proxy/ap
plication_1685498430457_0013/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0013
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 1
2023-05-31 11:32:46,457 Stage-1 map = 0%,  reduce = 0%
2023-05-31 11:33:12,326 Stage-1 map = 28%,  reduce = 0%, Cumulative CPU 26.38 sec
2023-05-31 11:33:13,368 Stage-1 map = 39%,  reduce = 0%, Cumulative CPU 28.2 sec
2023-05-31 11:33:14,517 Stage-1 map = 56%,  reduce = 0%, Cumulative CPU 29.71 sec
2023-05-31 11:33:16,734 Stage-1 map = 78%,  reduce = 0%, Cumulative CPU 34.27 sec
2023-05-31 11:33:17,756 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 35.61 sec
2023-05-31 11:33:20,910 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 36.88 sec
MapReduce Total cumulative CPU time: 36 seconds 880 msec
Ended Job = job_1685498430457_0013
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 1   Cumulative CPU: 36.88 sec   HDFS Read: 718448975 HDFS Writ
e: 6 SUCCESS
Total MapReduce CPU Time Spent: 36 seconds 880 msec
OK
82614
Time taken: 42.897 seconds, Fetched: 1 row(s)
```

- This is the Hive query for "Total count of superhost."



```
hive> SELECT neighbourhood_group_cleansed, COUNT(*) AS count
    > FROM airbnb.listings
    > GROUP BY neighbourhood_group_cleansed
    > ORDER BY count DESC
    > LIMIT 10;
Query ID = student_20230531113539_67a8740S-a343-41e6-b588-b1490a743608
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0014, Tracking URL = http://student-VirtualBox:8088/proxy/applicati
on_1685498430457_0014/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0014
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 3
2023-05-31 11:35:46,577 Stage-1 map = 0%,  reduce = 0%
2023-05-31 11:36:01,541 Stage-1 map = 11%,  reduce = 0%, Cumulative CPU 12.93 sec
2023-05-31 11:36:02,591 Stage-1 map = 56%,  reduce = 0%, Cumulative CPU 16.82 sec
2023-05-31 11:36:05,847 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 21.61 sec
2023-05-31 11:36:12,458 Stage-1 map = 100%,  reduce = 33%, Cumulative CPU 22.94 sec
2023-05-31 11:36:13,498 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 25.76 sec
MapReduce Total cumulative CPU time: 25 seconds 760 msec
Ended Job = job_1685498430457_0014
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0015, Tracking URL = http://student-VirtualBox:8088/proxy/applicati
on_1685498430457_0015/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0015
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2023-05-31 11:36:24,938 Stage-2 map = 0%,  reduce = 0%
2023-05-31 11:36:29,155 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 1.01 sec
2023-05-31 11:36:33,321 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 2.32 sec
MapReduce Total cumulative CPU time: 2 seconds 320 msec
Ended Job = job_1685498430457_0015
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 3   Cumulative CPU: 25.76 sec   HDFS Read: 718458646 HDFS Write: 1554
4 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.32 sec   HDFS Read: 20769 HDFS Write: 140 SUCCE
SS
Total MapReduce CPU Time Spent: 28 seconds 80 msec
OK
        145940
City of Los Angeles     18386
Manhattan       15855
Brooklyn        13954
Other Cities    13670
Maui    8596
Honolulu        7842
NULL    7065
Hawaii  6065
Queens  5824
Time taken: 56.246 seconds, Fetched: 10 row(s)
```

- This is the Hive query for "Top 10 neighbourhood group locations with the count."

<table>
<tr>
<td>

```
hive> SELECT host_location, COUNT(*) AS count
    > FROM airbnb.listings
    > GROUP BY host_location
    > ORDER BY count DESC
    > LIMIT 10;
Query ID = student_20230531113913_b77a2b91-6b25-4729-a080-83ae76d71a0b
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0016, Tracking URL = http://student-VirtualBox:8088/proxy/applicati
on_1685498430457_0016/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0016
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 3
2023-05-31 11:39:19,532 Stage-1 map = 0%,  reduce = 0%
2023-05-31 11:39:35,638 Stage-1 map = 11%,  reduce = 0%, Cumulative CPU 13.04 sec
2023-05-31 11:39:36,783 Stage-1 map = 22%,  reduce = 0%, Cumulative CPU 16.71 sec
2023-05-31 11:39:39,089 Stage-1 map = 39%,  reduce = 0%, Cumulative CPU 22.01 sec
2023-05-31 11:39:40,141 Stage-1 map = 78%,  reduce = 0%, Cumulative CPU 23.2 sec
2023-05-31 11:39:41,169 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 24.73 sec
2023-05-31 11:39:49,688 Stage-1 map = 100%,  reduce = 33%, Cumulative CPU 26.35 sec
2023-05-31 11:39:51,785 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 29.63 sec
MapReduce Total cumulative CPU time: 29 seconds 630 msec
Ended Job = job_1685498430457_0016
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0017, Tracking URL = http://student-VirtualBox:8088/proxy/applicati
on_1685498430457_0017/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0017
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2023-05-31 11:40:03,533 Stage-2 map = 0%,  reduce = 0%
2023-05-31 11:40:07,729 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 1.08 sec
2023-05-31 11:40:12,839 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 2.26 sec
MapReduce Total cumulative CPU time: 2 seconds 260 msec
Ended Job = job_1685498430457_0017
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 3   Cumulative CPU: 29.63 sec   HDFS Read: 718459057 HDFS Write: 3680
70 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.26 sec   HDFS Read: 373243 HDFS Write: 343 SUCC
ESS
Total MapReduce CPU Time Spent: 31 seconds 890 msec
OK
US      29569
New York, New York, United States       24763
Los Angeles, California, United States  15247
Austin, Texas, United States    9178
San Diego, California, United States    8217
San Francisco, California, United States        7297
Las Vegas, Nevada, United States        6083
NULL    5395
Chicago, Illinois, United States        4817
New Orleans, Louisiana, United States   4368
Time taken: 60.34 seconds, Fetched: 10 row(s)
```

</td>
<td>

- This is the Hive query for “Top 10 host locations with the count.”

</td>
</tr>
<tr>
<td>

```
hive> SELECT property_type, accommodates, COUNT(*) AS count
    > FROM airbnb.listings
    > GROUP BY property_type, accommodates
    > ORDER BY count DESC
    > LIMIT 5;
Query ID = student_20230531114117_4d346010-f43b-44ed-841b-bfef0e1170da
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 3
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0018, Tracking URL = http://student-VirtualBox:8088/proxy/applicati
on_1685498430457_0018/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0018
Hadoop job information for Stage-1: number of mappers: 3; number of reducers: 3
2023-05-31 11:41:22,861 Stage-1 map = 0%,  reduce = 0%
2023-05-31 11:41:37,675 Stage-1 map = 11%,  reduce = 0%, Cumulative CPU 13.42 sec
2023-05-31 11:41:38,736 Stage-1 map = 39%,  reduce = 0%, Cumulative CPU 17.06 sec
2023-05-31 11:41:40,848 Stage-1 map = 56%,  reduce = 0%, Cumulative CPU 22.35 sec
2023-05-31 11:41:41,879 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 23.17 sec
2023-05-31 11:41:51,588 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 29.09 sec
MapReduce Total cumulative CPU time: 29 seconds 90 msec
Ended Job = job_1685498430457_0018
Launching Job 2 out of 2
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685498430457_0019, Tracking URL = http://student-VirtualBox:8088/proxy/applicati
on_1685498430457_0019/
Kill Command = /home/WQD7007/hadoop/bin/hadoop job  -kill job_1685498430457_0019
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2023-05-31 11:42:03,103 Stage-2 map = 0%,  reduce = 0%
2023-05-31 11:42:08,293 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 1.23 sec
2023-05-31 11:42:13,493 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 2.67 sec
MapReduce Total cumulative CPU time: 2 seconds 670 msec
Ended Job = job_1685498430457_0019
MapReduce Jobs Launched:
Stage-Stage-1: Map: 3  Reduce: 3   Cumulative CPU: 29.09 sec   HDFS Read: 718460254 HDFS Write: 2724
29 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.67 sec   HDFS Read: 277940 HDFS Write: 124 SUCC
ESS
Total MapReduce CPU Time Spent: 31 seconds 760 msec
OK
Entire rental unit      2       21200
Entire rental unit      4       17531
Private room in home    2       14053
Entire condo    4       13474
Entire home     6       11241
Time taken: 56.794 seconds, Fetched: 5 row(s)
```

</td>
<td>

- This is the Hive query for “Top 5 property types and accommodates with the count.”

</td>
</tr>
</table>

```
hive> SELECT host_id, host_name, host_location, host_total_listings_count
    > FROM airbnb_listings
    > WHERE host_location = 'Tijuana, Baja California, Mexico' AND host_total
_listings_count = 3
    > LIMIT 10;
OK
260274485       Sonia   Tijuana, Baja California, Mexico        3
260274485       Sonia   Tijuana, Baja California, Mexico        3
153767576       Sandra  Tijuana, Baja California, Mexico        3
238969028       Beatriz Eugenia Tijuana, Baja California, Mexico        3
39363032        Lilian Victoria Tijuana, Baja California, Mexico        3
153767576       Sandra  Tijuana, Baja California, Mexico        3
167860812       Eunice  Tijuana, Baja California, Mexico        3
207219072       Martha  Tijuana, Baja California, Mexico        3
227931731       Edmundo Tijuana, Baja California, Mexico        3
227931731       Edmundo Tijuana, Baja California, Mexico        3
Time taken: 2.519 seconds, Fetched: 10 row(s)
```

- This is the Hive query for "10 hosts who are located in Tijuana, Baja, California, Mexico and have 'host_total_listings_count ' of 3."

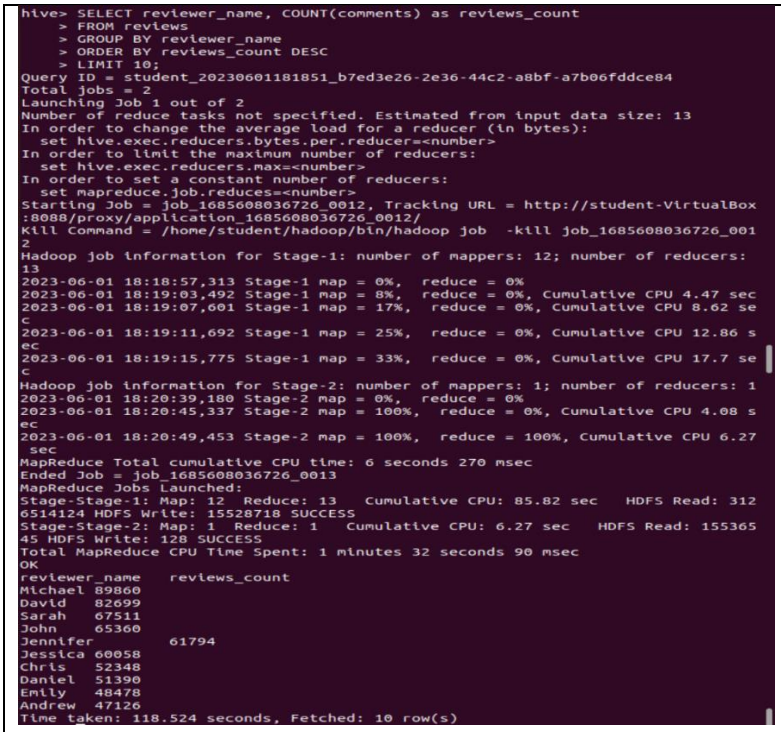
Reviews – Queries

```
hive> SELECT listing_id, COUNT(comments) as reviews_count
    > FROM reviews
    > GROUP BY listing_id
    > ORDER BY reviews_count DESC
    > LIMIT 10;
Query ID = student_20230601180501_61217c1c-ff6e-42ff-8cb1-fad908f53528
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 13
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685608036726_0008, Tracking URL = http://student-VirtualBox
:8088/proxy/application_1685608036726_0008/
Kill Command = /home/student/hadoop/bin/hadoop job  -kill job_1685608036726_000
8
Hadoop job information for Stage-1: number of mappers: 12; number of reducers:
13
2023-06-01 18:05:05,973 Stage-1 map = 0%,  reduce = 0%
2023-06-01 18:05:11,150 Stage-1 map = 8%,  reduce = 0%, Cumulative CPU 3.82 sec
2023-06-01 18:05:15,286 Stage-1 map = 17%,  reduce = 0%, Cumulative CPU 7.82 se
c
2023-06-01 18:05:19,382 Stage-1 map = 25%,  reduce = 0%, Cumulative CPU 11.15 s
ec
2023-06-01 18:05:23,462 Stage-1 map = 33%,  reduce = 0%, Cumulative CPU 15.03 s
ec
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2023-06-01 18:06:47,865 Stage-2 map = 0%,  reduce = 0%
2023-06-01 18:06:54,046 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 4.18 s
ec
2023-06-01 18:06:59,229 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 6.3
sec
MapReduce Total cumulative CPU time: 6 seconds 300 msec
Ended Job = job_1685608036726_0009
MapReduce Jobs Launched:
Stage-Stage-1: Map: 12  Reduce: 13   Cumulative CPU: 80.79 sec   HDFS Read: 312
6513922 HDFS Write: 40247157 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 6.3 sec   HDFS Read: 4025497
2 HDFS Write: 129 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 27 seconds 90 msec
OK
listing_id      reviews_count
44799007        2890
29819757        2295
34071681        2020
46534   1857
8357    1689
6652991 1520
42409434        1512
35158303        1438
8356380 1438
815639  1419
Time taken: 119.0 seconds, Fetched: 10 row(s)
```

- This is the Hive query for "Top 10 listings with most reviews".

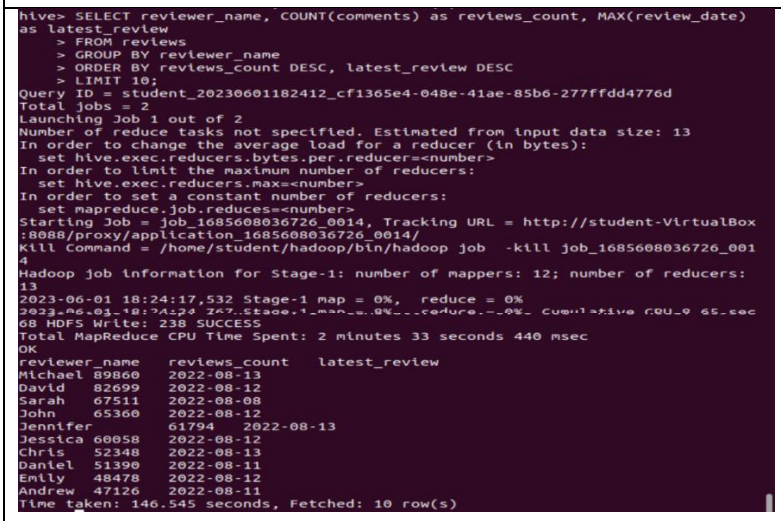
```
hive> SELECT listing_id, COUNT(comments) as reviews_count, MAX(review_date) as
latest_review
    > FROM reviews
    > GROUP BY listing_id
    > ORDER BY reviews_count DESC, latest_review DESC
    > LIMIT 10;
Query ID = student_20230601181137_809f6357-51d7-4cce-9823-338f65dbe9d5
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 13
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685608036726_0010, Tracking URL = http://student-VirtualBox
:8088/proxy/application_1685608036726_0010/
Kill Command = /home/student/hadoop/bin/hadoop job  -kill job_1685608036726_001
0
Hadoop job information for Stage-1: number of mappers: 12; number of reducers:
13
2023-06-01 18:11:42,276 Stage-1 map = 0%,  reduce = 0%
2023-06-01 18:11:49,479 Stage-1 map = 8%,  reduce = 0%, Cumulative CPU 8.12 sec
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2023-06-01 18:13:37,726 Stage-2 map = 0%,  reduce = 0%
2023-06-01 18:13:43,908 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 4.37 s
ec
2023-06-01 18:13:49,080 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 6.8
sec
MapReduce Total cumulative CPU time: 6 seconds 800 msec
Ended Job = job_1685608036726_0011
MapReduce Jobs Launched:
Stage-Stage-1: Map: 12  Reduce: 13   Cumulative CPU: 119.49 sec   HDFS Read: 31
26522533 HDFS Write: 40879348 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 6.8 sec   HDFS Read: 4088761
2 HDFS Write: 239 SUCCESS
Total MapReduce CPU Time Spent: 2 minutes 6 seconds 290 msec
OK
listing_id      reviews_count   latest_review
44799007        2890    2022-05-31
29819757        2295    2022-06-06
34071681        2020    2021-11-22
46534   1857    2022-06-21
8357    1689    2022-06-19
6652991 1520    2022-06-14
42409434        1512    2022-05-31
8356380 1438    2022-06-10
35158303        1438    2022-06-06
815639  1419    2022-06-03
Time taken: 133.808 seconds, Fetched: 10 row(s)
```

- This is the Hive query for "top 10 listing with most reviews and were recently commented".



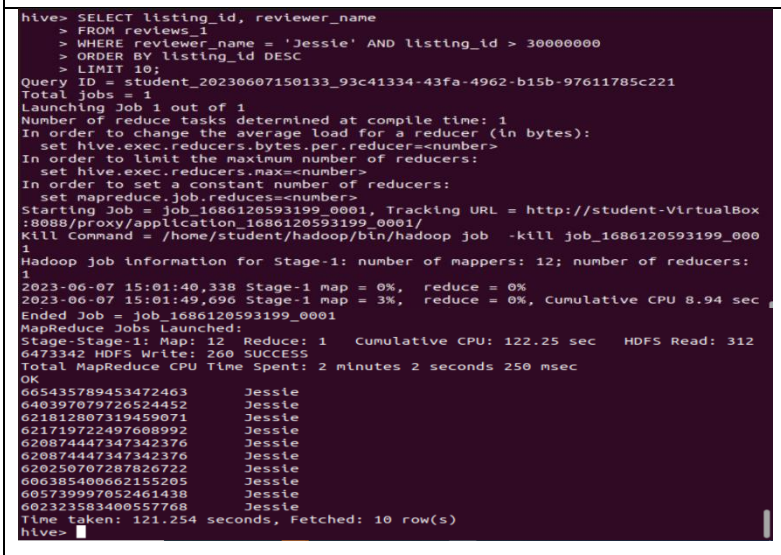
```
hive> SELECT reviewer_name, COUNT(comments) as reviews_count
    > FROM reviews
    > GROUP BY reviewer_name
    > ORDER BY reviews_count DESC
    > LIMIT 10;
Query ID = student_20230601181851_b7ed3e26-2e36-44c2-a8bf-a7b06fddce84
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 13
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685608036726_0012, Tracking URL = http://student-VirtualBox
:8088/proxy/application_1685608036726_0012/
Kill Command = /home/student/hadoop/bin/hadoop job  -kill job_1685608036726_001
2
Hadoop job information for Stage-1: number of mappers: 12; number of reducers:
13
2023-06-01 18:18:57,313 Stage-1 map = 0%,  reduce = 0%
2023-06-01 18:19:03,492 Stage-1 map = 8%,  reduce = 0%, Cumulative CPU 4.47 sec
2023-06-01 18:19:07,601 Stage-1 map = 17%,  reduce = 0%, Cumulative CPU 8.62 se
c
2023-06-01 18:19:11,692 Stage-1 map = 25%,  reduce = 0%, Cumulative CPU 12.86 s
ec
2023-06-01 18:19:15,775 Stage-1 map = 33%,  reduce = 0%, Cumulative CPU 17.7 se
c
Hadoop job information for Stage-2: number of mappers: 1; number of reducers: 1
2023-06-01 18:20:39,180 Stage-2 map = 0%,  reduce = 0%
2023-06-01 18:20:45,337 Stage-2 map = 100%,  reduce = 0%, Cumulative CPU 4.08 s
ec
2023-06-01 18:20:49,453 Stage-2 map = 100%,  reduce = 100%, Cumulative CPU 6.27
 sec
MapReduce Total cumulative CPU time: 6 seconds 270 msec
Ended Job = job_1685608036726_0013
MapReduce Jobs Launched:
Stage-Stage-1: Map: 12  Reduce: 13   Cumulative CPU: 85.82 sec   HDFS Read: 312
6514124 HDFS Write: 15528718 SUCCESS
Stage-Stage-2: Map: 1   Reduce: 1   Cumulative CPU: 6.27 sec   HDFS Read: 155365
45 HDFS Write: 128 SUCCESS
Total MapReduce CPU Time Spent: 1 minutes 32 seconds 90 msec
OK
reviewer_name   reviews_count
Michael 89860
David   82699
Sarah   67511
John    65360
Jennifer        61794
Jessica 60058
Chris   52348
Daniel  51390
Emily   48478
Andrew  47126
Time taken: 118.524 seconds, Fetched: 10 row(s)
```

- This is the Hive query for "Top 10 reviewers with most reviews."



```
hive> SELECT reviewer_name, COUNT(comments) as reviews_count, MAX(review_date)
as latest_review
    > FROM reviews
    > GROUP BY reviewer_name
    > ORDER BY reviews_count DESC, latest_review DESC
    > LIMIT 10;
Query ID = student_20230601182412_cf1365e4-048e-41ae-85b6-277ffdd4776d
Total jobs = 2
Launching Job 1 out of 2
Number of reduce tasks not specified. Estimated from input data size: 13
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1685608036726_0014, Tracking URL = http://student-VirtualBox
:8088/proxy/application_1685608036726_0014/
Kill Command = /home/student/hadoop/bin/hadoop job  -kill job_1685608036726_001
4
Hadoop job information for Stage-1: number of mappers: 12; number of reducers:
13
2023-06-01 18:24:17,532 Stage-1 map = 0%,  reduce = 0%
2023-06-01 18:24:24,767 Stage-1 map = 0%,  reduce = 0%, Cumulative CPU 0.65 sec
68 HDFS Write: 238 SUCCESS
Total MapReduce CPU Time Spent: 2 minutes 33 seconds 440 msec
OK
reviewer_name   reviews_count   latest_review
Michael 89860   2022-08-13
David   82699   2022-08-12
Sarah   67511   2022-08-08
John    65360   2022-08-12
Jennifer        61794   2022-08-13
Jessica 60058   2022-08-12
Chris   52348   2022-08-13
Daniel  51390   2022-08-11
Emily   48478   2022-08-12
Andrew  47126   2022-08-11
Time taken: 146.545 seconds, Fetched: 10 row(s)
```

- This is the Hive query for "top 10 reviewers with most reviews and recently commented".



```
hive> SELECT listing_id, reviewer_name
    > FROM reviews_1
    > WHERE reviewer_name = 'Jessie' AND listing_id > 30000000
    > ORDER BY listing_id DESC
    > LIMIT 10;
Query ID = student_20230607150133_93c41334-43fa-4962-b15b-97611785c221
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1686120593199_0001, Tracking URL = http://student-VirtualBox
:8088/proxy/application_1686120593199_0001/
Kill Command = /home/student/hadoop/bin/hadoop job  -kill job_1686120593199_000
1
Hadoop job information for Stage-1: number of mappers: 12; number of reducers:
1
2023-06-07 15:01:40,338 Stage-1 map = 0%,  reduce = 0%
2023-06-07 15:01:49,696 Stage-1 map = 3%,  reduce = 0%, Cumulative CPU 8.94 sec
Ended Job = job_1686120593199_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 12  Reduce: 1   Cumulative CPU: 122.25 sec   HDFS Read: 312
6473342 HDFS Write: 260 SUCCESS
Total MapReduce CPU Time Spent: 2 minutes 2 seconds 250 msec
OK
665435789453472463      Jessie
640397079726524452      Jessie
621812807319459071      Jessie
621719722497608992      Jessie
620874447347342376      Jessie
620874447347342376      Jessie
620250707287826722      Jessie
606385400662155205      Jessie
605739997052461438      Jessie
602323583400557768      Jessie
Time taken: 121.254 seconds, Fetched: 10 row(s)
hive>
```

- This is the Hive query for "Find 10 reviewers named "Jessie" after the company has listed more than 30m Airbnb."

APPENDIX D: Implementation in HBASE

Command Line	Narration

`student@student-VirtualBox:~/Downloads$ hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.columns="HBASE_ROW_KEY,reviews:id,reviews:listing_id,reviews:date,reviews:reviewer_id,reviews:reviewer_name,reviews:comments" -Dimporttsv.separator='|' airrev /tmp/reviews2_sep.csv`
`student@student-VirtualBox:~$ hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.columns="HBASE_ROW_KEY,listings:id,listings:listing_url,listings:scrape_id,listings:last_scraped,listings:name,listings:description,listings:neighborhood_overview,listings:picture_url,listings:host_id,listings:host_url,listings:host_name,listings:host_since,listings:host_location,listings:host_about,listings:host_response_time,listings:host_response_rate,listings:host_acceptance_rate,listings:host_is_superhost,listings:host_thumbnail_url,listings:host_picture_url,listings:host_neighbourhood,listings:host_listings_count,listings:host_total_listings_count,listings:host_verifications,listings:host_has_profile_pic,listings:host_identity_verified,listings:neighbourhood,listings:neighbourhood_cleansed,listings:neighbourhood_group_cleansed,listings:latitude,listings:longitude,listings:property_type,listings:room_type,listings:accommodates,listings:bathrooms,listings:bathrooms_text,listings:bedrooms,listings:beds,listings:amenities,listings:price,listings:minimum_nights,listings:maximum_nights,listings:minimum_minimum_nights,listings:maximum_minimum_nights,listings:minimum_maximum_nights,listings:maximum_maximum_nights,listings:minimum_nights_avg_ntm,listings:maximum_nights_avg_ntm,listings:calendar_updated,listings:has_availability,listings:availability_30,listings:availability_60,listings:availability_90,listings:availability_365,listings:calendar_last_scraped,listings:number_of_reviews,listings:number_of_reviews_ltm,listings:number_of_reviews_l30d,listings:first_review,listings:last_review,listings:review_scores_rating,listings:review_scores_accuracy,listings:review_scores_cleanliness,listings:review_scores_checkin,listings:review_scores_communication,listings:review_scores_location,listings:review_scores_value,listings:license,listings:instant_bookable,listings:calculated_host_listings_count,listings:calculated_host_listings_count_entire_homes,listings:calculated_host_listings_count_private_rooms,listings:calculated_host_listings_count_shared_rooms,listings:reviews_per_month,listings:a" -Dimporttsv.separator='|' airlist /tmp/listings_sep.csv`
Reviews – Data Import
	• The reviews.csv file is imported into the table name 'airrev' which was created in Hbase, and all the column names in reviews.csv are under the column family named 'reviews'.
	• The listings.csv file is imported into the table name 'airlist' which was created in Hbase, and all the column names in listings.csv are under the column family named 'listings'.

`2023-06-06 13:58:39,378 INFO [main] mapreduce.Job: Running job: job_1686030516625_0001`
`2023-06-06 13:58:45,812 INFO [main] mapreduce.Job: Job job_1686030516625_0001 running in uber mode : false`
`2023-06-06 13:58:45,813 INFO [main] mapreduce.Job: map 0% reduce 0%`
`2023-06-06 13:58:52,868 INFO [main] mapreduce.Job: map 4% reduce 0%`
`2023-06-06 13:58:58,895 INFO [main] mapreduce.Job: map 9% reduce 0%`
`2023-06-06 13:59:02,916 INFO [main] mapreduce.Job: map 13% reduce 0%`
`2023-06-06 13:59:08,955 INFO [main] mapreduce.Job: map 17% reduce 0%`
`2023-06-06 13:59:14,986 INFO [main] mapreduce.Job: map 22% reduce 0%`
`2023-06-06 13:59:21,012 INFO [main] mapreduce.Job: map 26% reduce 0%`
`2023-06-06 13:59:27,040 INFO [main] mapreduce.Job: map 30% reduce 0%`
`2023-06-06 13:59:32,059 INFO [main] mapreduce.Job: map 35% reduce 0%`
`2023-06-06 13:59:38,080 INFO [main] mapreduce.Job: map 39% reduce 0%`
`2023-06-06 13:59:44,103 INFO [main] mapreduce.Job: map 43% reduce 0%`
`2023-06-06 13:59:49,125 INFO [main] mapreduce.Job: map 48% reduce 0%`
`2023-06-06 13:59:54,138 INFO [main] mapreduce.Job: map 52% reduce 0%`
`2023-06-06 14:00:00,155 INFO [main] mapreduce.Job: map 57% reduce 0%`
`2023-06-06 14:00:05,175 INFO [main] mapreduce.Job: map 61% reduce 0%`
`2023-06-06 14:00:12,213 INFO [main] mapreduce.Job: map 65% reduce 0%`
`2023-06-06 14:00:18,239 INFO [main] mapreduce.Job: map 70% reduce 0%`
`2023-06-06 14:00:23,265 INFO [main] mapreduce.Job: map 74% reduce 0%`
`2023-06-06 14:00:30,297 INFO [main] mapreduce.Job: map 78% reduce 0%`
`2023-06-06 14:00:36,325 INFO [main] mapreduce.Job: map 83% reduce 0%`
`2023-06-06 14:00:41,354 INFO [main] mapreduce.Job: map 87% reduce 0%`
`2023-06-06 14:00:48,379 INFO [main] mapreduce.Job: map 91% reduce 0%`
`2023-06-06 14:00:53,398 INFO [main] mapreduce.Job: map 96% reduce 0%`
`2023-06-06 14:00:58,415 INFO [main] mapreduce.Job: map 100% reduce 0%`
`2023-06-06 14:00:59,422 INFO [main] mapreduce.Job: Job job_1686030516625_0001`
`Total megabyte-milliseconds taken by all map tasks=6172262`
`Map-Reduce Framework`
`Map input records=10460000`
`Map output records=10459938`
`Input split bytes=2461`
`Spilled Records=0`
`Failed Shuffles=0`
`Merged Map outputs=0`
`GC time elapsed (ms)=2712`
`CPU time spent (ms)=93800`
`Physical memory (bytes) snapshot=5676404736`
`Virtual memory (bytes) snapshot=44289409024`
`Total committed heap usage (bytes)=2740453376`
`ImportTsv`
`Bad Lines=62`
`File Input Format Counters`
`Bytes Read=3089117973`
`File Output Format Counters`
`Bytes Written=0`
	• Here is the MapReduce process and time taken of importation for the reviews.csv file.

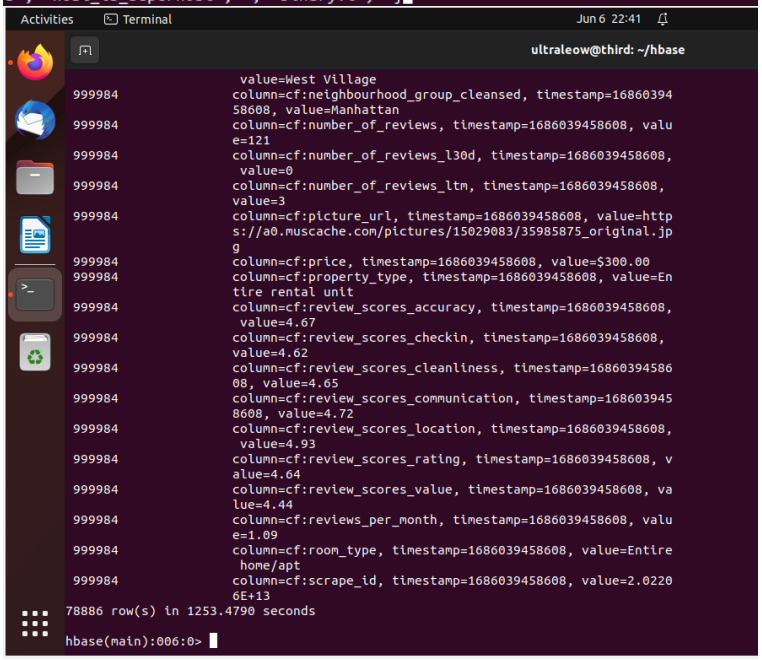
Listings – Data Import

```
2023-06-06 11:40:09,548 INFO  [main] mapreduce.Job: Running job: job_1686022390
229_0001
2023-06-06 11:40:14,613 INFO  [main] mapreduce.Job: Job job_1686022390229_0001
running in uber mode : false
2023-06-06 11:40:14,614 INFO  [main] mapreduce.Job:  map 0% reduce 0%
2023-06-06 11:40:23,685 INFO  [main] mapreduce.Job:  map 14% reduce 0%
2023-06-06 11:40:24,694 INFO  [main] mapreduce.Job:  map 17% reduce 0%
2023-06-06 11:40:32,728 INFO  [main] mapreduce.Job:  map 31% reduce 0%
2023-06-06 11:40:33,733 INFO  [main] mapreduce.Job:  map 33% reduce 0%
2023-06-06 11:40:40,768 INFO  [main] mapreduce.Job:  map 50% reduce 0%
2023-06-06 11:40:48,806 INFO  [main] mapreduce.Job:  map 64% reduce 0%
2023-06-06 11:40:49,809 INFO  [main] mapreduce.Job:  map 67% reduce 0%
2023-06-06 11:40:57,843 INFO  [main] mapreduce.Job:  map 83% reduce 0%
2023-06-06 11:41:01,856 INFO  [main] mapreduce.Job:  map 100% reduce 0%
2023-06-06 11:41:01,861 INFO  [main] mapreduce.Job: Job job_1686022390229_0001
completed successfully
```

- Here shows the MapReduce process and time taken of importation for the listings.csv file.

Listings – Queries

```
hbase(main):010:0> scan 'airlist', {FILTER => "SingleColumnValueFilter('listing
s', 'host_is_superhost', =, 'binary:t') "}
```



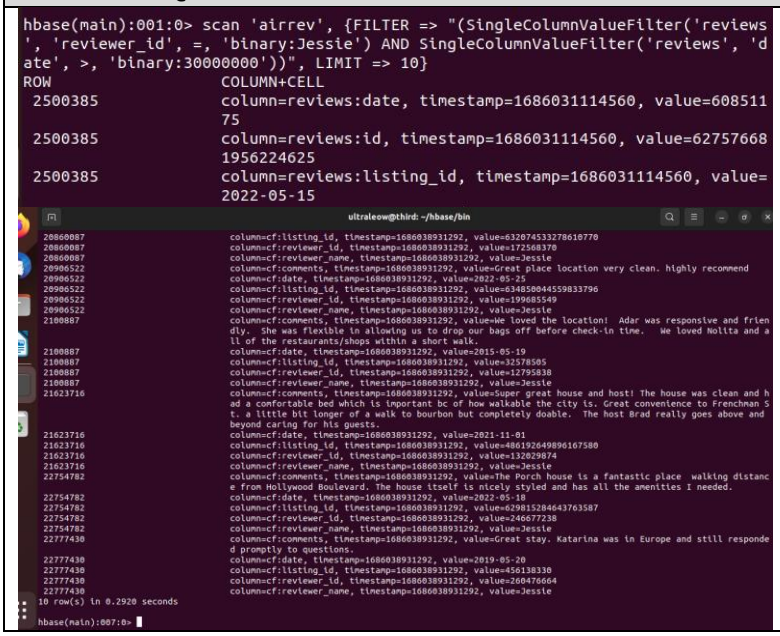

```
                        value=West Village
999984                  column=cf:neighbourhood_group_cleansed, timestamp=16860394
                        58608, value=Manhattan
999984                  column=cf:number_of_reviews, timestamp=1686039458608, valu
                        e=121
999984                  column=cf:number_of_reviews_l30d, timestamp=1686039458608,
                         value=0
999984                  column=cf:number_of_reviews_ltm, timestamp=1686039458608,
                        value=3
999984                  column=cf:picture_url, timestamp=1686039458608, value=http
                        s://a0.muscache.com/pictures/15029083/35985875_original.jp
                        g
999984                  column=cf:price, timestamp=1686039458608, value=$300.00
999984                  column=cf:property_type, timestamp=1686039458608, value=En
                        tire rental unit
999984                  column=cf:review_scores_accuracy, timestamp=1686039458608,
                         value=4.67
999984                  column=cf:review_scores_checkin, timestamp=1686039458608,
                        value=4.62
999984                  column=cf:review_scores_cleanliness, timestamp=16860394586
                        08, value=4.65
999984                  column=cf:review_scores_communication, timestamp=168603945
                        8608, value=4.72
999984                  column=cf:review_scores_location, timestamp=1686039458608,
                         value=4.93
999984                  column=cf:review_scores_rating, timestamp=1686039458608, v
                        alue=4.64
999984                  column=cf:review_scores_value, timestamp=1686039458608, va
                        lue=4.44
999984                  column=cf:reviews_per_month, timestamp=1686039458608, valu
                        e=1.09
999984                  column=cf:room_type, timestamp=1686039458608, value=Entire
                         home/apt
999984                  column=cf:scrape_id, timestamp=1686039458608, value=2.0220
                        6E+13
78886 row(s) in 1253.4790 seconds

hbase(main):006:0>
```

- This is the HBase query for "Total count of superhost".

```
hbase(main):004:0> scan 'airlist', {FILTER => "(SingleColumnValueFilter('info',
 'host_location', =, 'binary:Tijuana, Baja California, Mexico') AND SingleColum
nValueFilter('info', 'host_total_listings_count', =, 'binary:3'))", LIMIT => 10
}
ROW                     COLUMN+CELL
 10000070               column=listings:accommodates, timestamp=1686022805803, val
                        ue=
 10000070               column=listings:amenities, timestamp=1686022805803, value=
                        $80.00
 10000070               column=listings:availability_30, timestamp=1686022805803,
                        value=0
 10000070               column=listings:availability_365, timestamp=1686022805803,
                         value=4/6/2022
 10000070               column=listings:availability_60, timestamp=1686022805803,
                        value=6
 10000070               column=listings:availability_90, timestamp=1686022805803,
                        value=221
 10000070               column=listings:bathrooms, timestamp=1686022805803, value=
                        1 shared bath
 10000070               column=listings:bathrooms_text, timestamp=1686022805803, v
                        alue=1.0
 10000070               column=listings:bedrooms, timestamp=1686022805803, value=1
                        .0
 10005528               column=listings:review_scores_location, timestamp=16860228
                        05803, value=4.79
 10005528               column=listings:review_scores_rating, timestamp=1686022805
                        803, value=4.9
 10005528               column=listings:review_scores_value, timestamp=16860228058
                        03, value=
 10005528               column=listings:reviews_per_month, timestamp=1686022805803
                        , value=
 10005528               column=listings:room_type, timestamp=1686022805803, value=
                        2
 10005528               column=listings:scrape_id, timestamp=1686022805803, value=
                        15/6/2022
10 row(s) in 0.3500 seconds
```

- This is the Hbase query for "10 hosts who are located in Tijuana, Baja California, Mexico and have 'host_total_listings_count' of 3".

Reviews – Queries



- This is the HBase query for “Find 10 reviewers named “Jessie” after the company has listed more than 30m Airbnb”.

APPENDIX E: Implementation in MongoDB

Command Line

```
student@student-VirtualBox:~$ mongosh
Current Mongosh Log ID: 647afe3d070ed400d82c8d73
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS
=2000&appName=mongosh+1.9.1
Using MongoDB:          6.0.6
Using Mongosh:          1.9.1

For mongosh info see: https://docs.mongodb.com/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodically
(https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2023-06-03T16:33:38.387+08:00: Using the XFS filesystem is strongly recommended with the Wired
Tiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
   2023-06-03T16:33:38.675+08:00: Access control is not enabled for the database. Read and write
access to data and configuration is unrestricted
   2023-06-03T16:33:38.676+08:00: vm.max_map_count is too low
------

test> show dbs
admin   40.00 KiB
config  12.00 KiB
local   40.00 KiB
test> use hotel
switched to db hotel
```

- MongoDB was initialized with the `mongosh` keyword without any need for Hadoop.
- Database 'hotel' was created.

Reviews – Data Import

```
student@student-VirtualBox:~$ time mongoimport --db hotel --collection reviews --type csv --heade
rline --file 7007_data/merged_reviews.csv
2023-06-05T02:32:56.617+0800    connected to: mongodb://localhost/
2023-06-05T02:32:59.618+0800    [........................] hotel.reviews        5.85MB/2.91GB (0.
2%)
2023-06-05T02:33:02.617+0800    [........................] hotel.reviews        12.7MB/2.91GB (0.
4%)
2023-06-05T02:33:05.619+0800    [........................] hotel.reviews        20.2MB/2.91GB (0.
7%)
2023-06-05T02:33:08.618+0800    [........................] hotel.reviews        27.5MB/2.91GB (0.
9%)
2023-06-05T02:33:11.618+0800    [........................] hotel.reviews        34.0MB/2.91GB (1.
1%)
2023-06-05T02:33:14.618+0800    [........................] hotel.reviews        39.1MB/2.91GB (1.
3%)
2023-06-05T02:33:17.618+0800    [........................] hotel.reviews        43.9MB/2.91GB (1.
5%)
2023-06-05T02:33:20.618+0800    [........................] hotel.reviews        48.3MB/2.91GB (1.
```

```
hotel> db.reviews.count()
10459999
```

```
hotel> db.reviews.find().limit(2)
[
  {
    _id: ObjectId("647b39c39670b5ea74d87919"),
    listing_id: 109,
    id: 449036,
    date: '2011-08-15',
    reviewer_id: 927861,
    reviewer_name: 'Edwin',
    comments: 'The host canceled my reservation the day before arrival.'
  },
  {
    _id: ObjectId("647b39c39670b5ea74d8791a"),
    listing_id: 45392,
    id: 270172,
    date: '2011-05-18',
    reviewer_id: 92047,
    reviewer_name: 'Kristin',
    comments: "Olivia helped me with directions to arrive & leave, since i failed to get her addr
ess. she is very welcoming & friendly, as are her nice dogs. the room is very private with it's o
wn bathroom & entrance. she also accommodated me with a reading light by the bed & tea in the mor
ning. very comfortable bed too!"
  }
]
```

- To import the data, the MongoDB shell must be exited.
- The data import is done via the `mongoimport` utility, while the `time` utility is to get the execution time of the data import.
- There is no need to create a table first. A MongoDB table will be created with the first line as a header automatically.
- The content of the table can be shown using the `find()` function.

Listings – Data Import

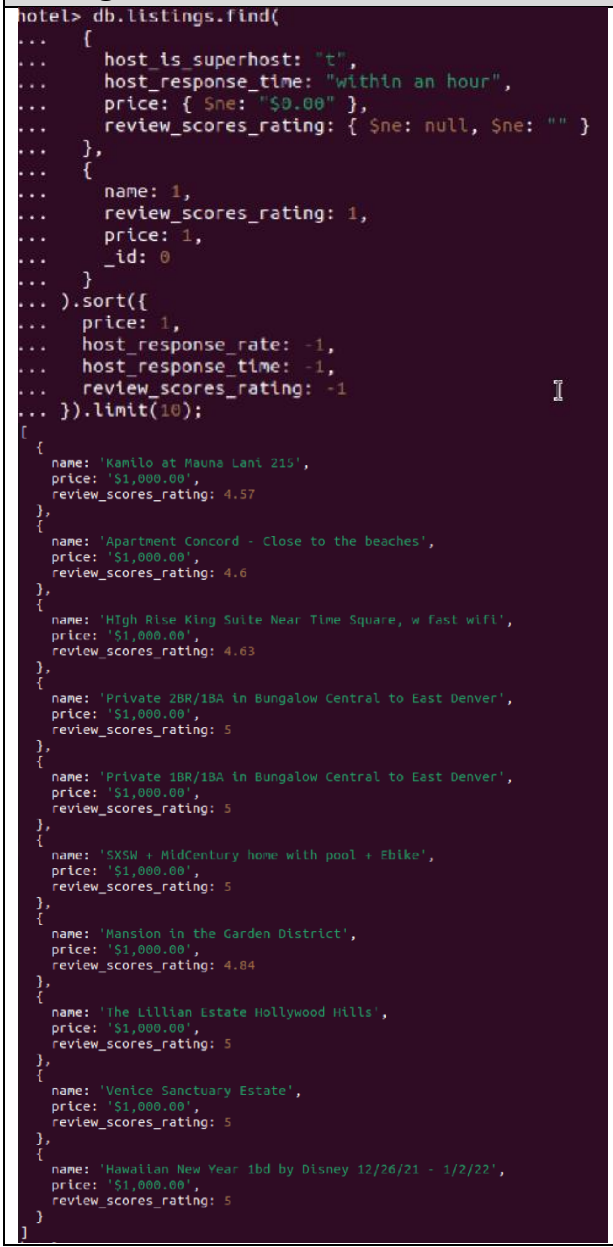
```
student@student-VirtualBox:~$ time mongoimport --db hotel --collection listings --type csv --head
erline --file 7007_data/merged_listings.csv
2023-06-03T20:56:23.231+0800    connected to: mongodb://localhost/
2023-06-03T20:56:26.232+0800    [........................] hotel.listings       26.6MB/683MB (3.9
%)
2023-06-03T20:56:29.238+0800    [#.......................] hotel.listings       53.8MB/683MB (7.9
%)
2023-06-03T20:56:32.231+0800    [##......................] hotel.listings       80.4MB/683MB (11.
8%)
2023-06-03T20:56:35.231+0800    [###.....................] hotel.listings       107MB/683MB (15.7
%)
2023-06-03T20:56:38.231+0800    [####....................] hotel.listings       138MB/683MB (20.2
%)
2023-06-03T20:56:41.232+0800    [#####...................] hotel.listings       166MB/683MB (24.3
%)
2023-06-03T20:56:44.231+0800    [######..................] hotel.listings       194MB/683MB (28.4
%)
2023-06-03T20:56:47.232+0800    [#######.................] hotel.listings       219MB/683MB (32.1
%)
```

```
hotel> db.listings.count()
261538
```

- Like the `Reviews` table, the data import for `Listings` is done via the `mongoimport` utility, while the `time` utility is to get the execution time of the data import

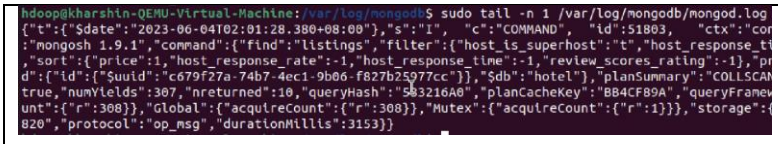
```
hotel> db.listings.find().limit(2)
[
  {
    _id: ObjectId("647b3877db9248ab0a989ce1"),
    id: 45392,
    listing_url: 'https://www.airbnb.com/rooms/45392',
    scrape_id: 20220600000000,
    last_scraped: '6/6/2022',
    name: 'Cute Home in Mount Washington',
    description: "<b>The space</b><br />Cute house in Mount Washington near Griffith Park we have
 one room available (Red Bedroom) Continental Breakfast, closets are 90% empty, it  can accommoda
te 3 people max. for a professional person, working person, or grad student, no heavy drinking. M
ust like dogs - have one indoor dog. House is fully furnished, street parking. Close to shops. Ea
sy access to freeways. <br /><br />Amenities: <br />- High speed wireless internet <br />- Cable
 TV<br />- Full size bed<br />- Fully equipped kitchen, ready to cook and serve <br />- TOWELS and
 SHEETS<br /><br />NO SMOKING inside but you can smoke outside on our patio. It's a safe and frie
ndly neighborhood.<br /><br />Bonuses: <br />- 24/7 support! Our resident' s knowledge is here fo
r you! <br /><br />Driving,<br />10 mins from China town, downtown LA.<br />5 mins from Silver La
ke and Eagle Rock and 20 mins from Pasadena.<br /><br />20 to 25 mins from Hollywood. <br /><br /
>Metro stop<br />Lincon/Cypress<br /><br />Close",
    neighborhood_overview: '',
    picture_url: 'https://a0.muscache.com/pictures/miso/Hosting-45392/original/c453103b-6752-4655
-b0dc-161e10bae98c.jpeg',
    host_id: 201514,
    host_url: 'https://www.airbnb.com/users/show/201514',
    host_name: 'Olivia & Alexey',
    host_since: '14/8/2010',
    host_location: 'Los Angeles, California, United States',
    host_about: 'We love outdoors, museums, art, film, travel, photography,\n' +
      'languages, hockey, soccer.\n',
    host_response_time: 'within a day',
    host_response_rate: '100%',
    host_acceptance_rate: '86%',
    host_is_superhost: 'f',
    host_thumbnail_url: 'https://a0.muscache.com/im/users/201514/profile_pic/1344289344/original
```

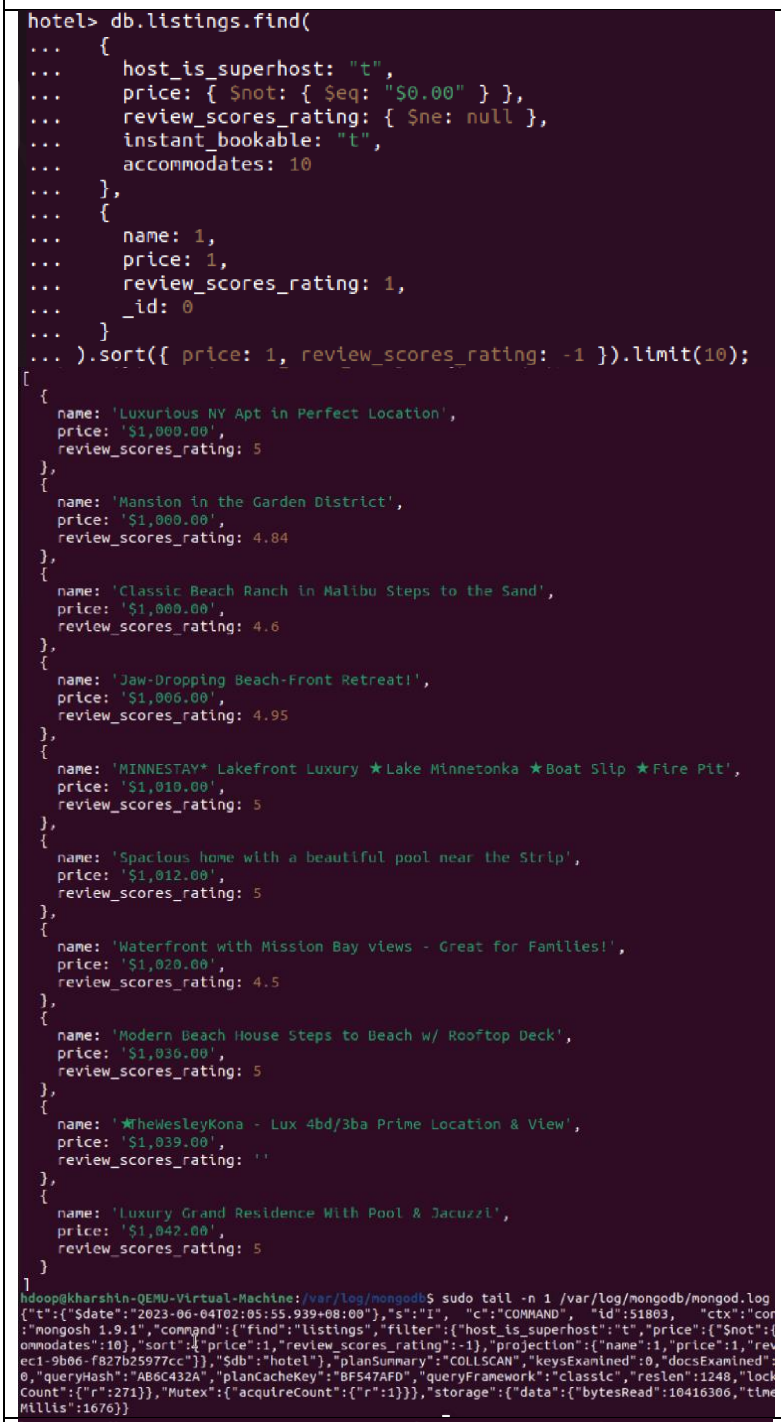
Listings – Queries



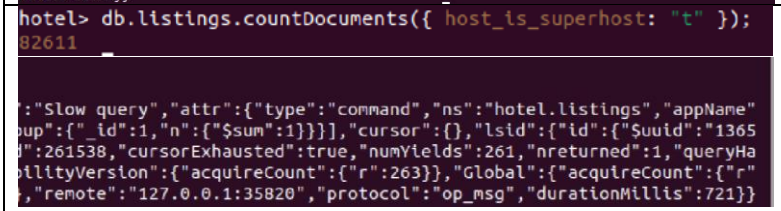
```
hotel> db.listings.find(
...   {
...     host_is_superhost: "t",
...     host_response_time: "within an hour",
...     price: { $ne: "$0.00" },
...     review_scores_rating: { $ne: null, $ne: "" }
...   },
...   {
...     name: 1,
...     review_scores_rating: 1,
...     price: 1,
...     _id: 0
...   }
... ).sort({
...   price: 1,
...   host_response_rate: -1,
...   host_response_time: -1,
...   review_scores_rating: -1
... }).limit(10);
[
  {
    name: 'Kamilo at Mauna Lani 215',
    price: '$1,000.00',
    review_scores_rating: 4.57
  },
  {
    name: 'Apartment Concord - Close to the beaches',
    price: '$1,000.00',
    review_scores_rating: 4.6
  },
  {
    name: 'HIgh Rise King Suite Near Time Square, w fast wifi',
    price: '$1,000.00',
    review_scores_rating: 4.63
  },
  {
    name: 'Private 2BR/1BA in Bungalow Central to East Denver',
    price: '$1,000.00',
    review_scores_rating: 5
  },
  {
    name: 'Private 1BR/1BA in Bungalow Central to East Denver',
    price: '$1,000.00',
    review_scores_rating: 5
  },
  {
    name: 'SXSW + MidCentury home with pool + Ebike',
    price: '$1,000.00',
    review_scores_rating: 5
  },
  {
    name: 'Mansion in the Garden District',
    price: '$1,000.00',
    review_scores_rating: 4.84
  },
  {
    name: 'The Lillian Estate Hollywood Hills',
    price: '$1,000.00',
    review_scores_rating: 5
  },
  {
    name: 'Venice Sanctuary Estate',
    price: '$1,000.00',
    review_scores_rating: 5
  },
  {
    name: 'Hawaiian New Year 1bd by Disney 12/26/21 - 1/2/22',
    price: '$1,000.00',
    review_scores_rating: 5
  }
]
```

- This is the MongoDB query for “Top 10 cheapest superhost hotels with the best rating and fastest response time and rate”.





- This is the MongoDB query for “Top 10 cheapest superhost hotels with the best rating and accommodates 10 person and are instantly bookable”.



- This is the MongoDB query for “Total count of superhost”.

<table>
<tr>
<td>

```
hotel> db.listings.aggregate([ { $match: { neighbourhood_group_cleansed: { $
ne: '' } } }, { $group: { _id: "$neighbourhood_group_cleansed", count: { $su
m: 1 } } }, { $sort: { count: -1 } }, { $limit: 10 }]);
[
  { _id: 'City of Los Angeles', count: 18386 },
  { _id: 'Manhattan', count: 15855 },
  { _id: 'Brooklyn', count: 13954 },
  { _id: 'Other Cities', count: 13671 },
  { _id: 'Maui', count: 8596 },
  { _id: 'Honolulu', count: 7842 },
  { _id: 'Hawaii', count: 6065 },
  { _id: 'Queens', count: 5824 },
  { _id: 'Kauai', count: 3842 },
  { _id: 'Unincorporated Areas', count: 3271 }
]
hdoop@kharshin-QEMU-Virtual-Machine:/var/log/mongodb$ sudo tail -n 1 /var/log/mongodb/mongod.log
{"t":{"$date":"2023-06-04T02:36:13.970-08:00"},"s":"I",  "c":"COMMAND",  "id":51803,   "ctx":"conn8","msg":"Sl
:"mongosh 1.9.1","command":{"aggregate":"listings","pipeline":[{"$match":{"neighbourhood_group_cleansed":{"$ne
"$sum":1}}},{"$sort":{"count":-1}},{"$limit":10}],"cursor":{},"lsid":{"id":{"$uuid":"13658380-d10b-4087-af07-9
ed":0,"docsExamined":261538,"hasSortStage":true,"cursorExhausted":true,"numYields":261,"nreturned":10,"queryHa
eslen":490,"locks":{"FeatureCompatibilityVersion":{"acquireCount":{"r":263}},"Global":{"acquireCount":{"r":263
":15707195,"timeReadingMicros":12133}},"remote":"127.0.0.1:35820","protocol":"op_msg","durationMillis":420}}
```

</td>
<td>

- This is the MongoDB query for the “Top 10 neighbourhood group with count”.

</td>
</tr>
<tr>
<td>

```
hotel> db.listings.aggregate([ { $match: { host_location: { $ne: '' } } }, {
 $group: { _id: "$host_location", count: { $sum: 1 } } }, { $sort: { count:
-1 } }, { $limit: 10 }]);
[
  { _id: 'US', count: 29569 },
  { _id: 'New York, New York, United States', count: 24763 },
  { _id: 'Los Angeles, California, United States', count: 15247 },
  { _id: 'Austin, Texas, United States', count: 9178 },
  { _id: 'San Diego, California, United States', count: 8217 },
  { _id: 'San Francisco, California, United States', count: 7297 },
  { _id: 'Las Vegas, Nevada, United States', count: 6083 },
  { _id: 'Chicago, Illinois, United States', count: 4817 },
  { _id: 'New Orleans, Louisiana, United States', count: 4368 },
  { _id: 'Brooklyn, New York, United States', count: 4335 }
]
hdoop@kharshin-QEMU-Virtual-Machine:/var/log/mongodb$ sudo tail -n 1 /var/log/mongodb/mongod.log
{"t":{"$date":"2023-06-04T02:39:18.969+08:00"},"s":"I",  "c":"COMMAND",  "id":51803,   "ctx":"con
:"mongosh 1.9.1","command":{"aggregate":"listings","pipeline":[{"$match":{"host_location":{"$ne":
-1}},{"$limit":10}],"cursor":{},"lsid":{"id":{"$uuid":"13658380-d10b-4087-af07-94088a952c03"}},"$
asSortStage":true,"cursorExhausted":true,"numYields":288,"nreturned":10,"queryHash":"A46BFB3E","p
mpatibilityVersion":{"acquireCount":{"r":290}},"Global":{"acquireCount":{"r":290}},"Mutex":{"acqu
:77450}},"remote":"127.0.0.1:35820","protocol":"op_msg","durationMillis":2478}}
```

</td>
<td>

- This is the MongoDB query for “Top 10 host location with count”.

</td>
</tr>
<tr>
<td>

```
hotel> db.listings.aggregate([ { $group: { _id: { property_type: "$property_
type", accommodates: "$accommodates" }, count: { $sum: 1 } } }, { $sort: { c
ount: -1 } }, { $limit: 5 }]);
[
  {
    _id: { property_type: 'Entire rental unit', accommodates: 2 },
    count: 21200
  },
  {
    _id: { property_type: 'Entire rental unit', accommodates: 4 },
    count: 17531
  },
  {
    _id: { property_type: 'Private room in home', accommodates: 2 },
    count: 14053
  },
  {
    _id: { property_type: 'Entire condo', accommodates: 4 },
    count: 13474
  },
  {
    _id: { property_type: 'Entire home', accommodates: 6 },
    count: 11241
  }
]
hdoop@kharshin-QEMU-Virtual-Machine:/var/log/mongodb$ sudo tail -n 1 /var/log/mongodb/mongod.log
{"t":{"$date":"2023-06-04T02:42:26.414+08:00"},"s":"I",  "c":"COMMAND",  "id":51803,   "ctx":"con
:"mongosh 1.9.1","command":{"aggregate":"listings","pipeline":[{"$group":{"_id":{"property_type":
{"count":-1}},{"$limit":5}],"cursor":{},"lsid":{"id":{"$uuid":"13658380-d10b-4087-af07-94088a952c
61538,"hasSortStage":true,"cursorExhausted":true,"numYields":262,"nreturned":5,"queryHash":"1FBA8
:{"acquireCount":{"r":264}},"Global":{"acquireCount":{"r":264}},"Mutex":{"acquireCount":{"r":2}}}
127.0.0.1:35820","protocol":"op_msg","durationMillis":980}}
```

</td>
<td>

- This is the MongoDB query for “Top 5 property type and accommodates with count”.

</td>
</tr>
</table>

```
hotel> db.listings.aggregate([
...   {
...     $match: {
...       host_location: "Tijuana, Baja California, Mexico",
...       host_total_listings_count: 3
...     }
...   },
...   {
...     $group: {
...       _id: "$host_name"
...     }
...   },
...   {
...     $limit: 10
...   }
... ])
[
  { _id: 'Martha' },
  { _id: 'Lu' },
  { _id: 'Beatriz Eugenia' },
  { _id: 'Sonia' },
  { _id: 'Edmundo' },
  { _id: 'Eunice' },
  { _id: 'Emiliano' },
  { _id: 'Guillermo' },
  { _id: 'Lilian Victoria' },
  { _id: 'Blanca Flor' }
]
hdoop@kharshin-QEMU-Virtual-Machine:/var/log/mongodb$ sudo tail -n 1 /var/log/mo
ngodb/mongod.log
{"t":{"$date":"2023-06-05T15:13:04.167+08:00"},"s":"I",  "c":"COMMAND",  "id":51
803,   "ctx":"conn12","msg":"Slow query","attr":{"type":"command","ns":"hotel.li
stings","appName":"mongosh 1.9.1","command":{"aggregate":"listings","pipeline":[
{"$match":{"host_location":"Tijuana, Baja California, Mexico","host_total_listin
gs_count":3}},{"$group":{"_id":"$host_name"}},{"$limit":10}],"cursor":{},"lsid":
{"id":{"$uuid":"99f29136-a922-4932-af1e-1bb4d96c2af8"}},"$db":"hotel"},"planSumm
ary":"COLLSCAN","keysExamined":0,"docsExamined":261538,"cursorExhausted":true,"n
umYields":262,"nreturned":10,"queryHash":"1E98B1F7","planCacheKey":"1E98B1F7","q
ueryFramework":"sbe","reslen":367,"locks":{"FeatureCompatibilityVersion":{"acqui
reCount":{"r":264}},"Global":{"acquireCount":{"r":264}},"Mutex":{"acquireCount":
{"r":2}}},"storage":{"data":{"bytesRead":15068286,"timeReadingMicros":5899}},"re
mote":"127.0.0.1:44394","protocol":"op_msg","durationMillis":641}}
```

- This is the MongoDB query for “10 hosts who are located in Tijuana, Baja, California, Mexico and have ‘host_total_listings_count’ of 3”.

Reviews – Queries

```
hotel> db.reviews.aggregate([
...   {
...     $group: {
...       _id: "$listing_id",
...       count: { $sum: 1 }
...     }
...   },
...   {
...     $sort: { count: -1 }
...   },
...   {
...     $limit: 10
...   }
... ])
[
  { _id: 44799007, count: 2890 },
  { _id: 29819757, count: 2295 },
  { _id: 34071681, count: 2020 },
  { _id: 46534, count: 1857 },
  { _id: 8357, count: 1689 },
  { _id: 6652991, count: 1520 },
  { _id: 42409434, count: 1512 },
  { _id: 35158303, count: 1438 },
  { _id: 8356380, count: 1438 },
  { _id: 815639, count: 1419 }
]
executionStats: {
  executionSuccess: true,
  nReturned: 208658,
  executionTimeMillis: 11417,
  totalKeysExamined: 0,
  totalDocsExamined: 10459999,
```

- This is the MongoDB query for “top 10 listings with most reviews”.

```
hotel> db.reviews.aggregate([
...   {
...     $group: {
...       _id: "$listing_id",
...       count: { $sum: 1 },
...       lastReviewDate: { $max: "$date" }
...     }
...   },
...   {
...     $sort: { count: -1, lastReviewDate: -1 }
...   },
...   {
...     $limit: 10
...   }
... ])
[
  { _id: 44799007, count: 2890, lastReviewDate: '2022-05-31' },
  { _id: 29819757, count: 2295, lastReviewDate: '2022-06-06' },
  { _id: 34071681, count: 2020, lastReviewDate: '2021-11-22' },
  { _id: 46534, count: 1857, lastReviewDate: '2022-06-21' },
  { _id: 8357, count: 1689, lastReviewDate: '2022-06-19' },
  { _id: 6652991, count: 1520, lastReviewDate: '2022-06-14' },
  { _id: 42409434, count: 1512, lastReviewDate: '2022-05-31' },
  { _id: 8356380, count: 1438, lastReviewDate: '2022-06-10' },
  { _id: 35158303, count: 1438, lastReviewDate: '2022-06-06' },
  { _id: 815639, count: 1419, lastReviewDate: '2022-06-03' }
]
```

```
executionStats: {
  executionSuccess: true,
  nReturned: 208658,
  executionTimeMillis: 17333,
  totalKeysExamined: 0,
  totalDocsExamined: 10459999,
```

- This is the MongoDB query for the "top 10 listing with most reviews and were recently commented".

```
hotel> db.reviews.aggregate([
...   {
...     $group: {
...       _id: "$reviewer_id",
...       count: { $sum: 1 }
...     }
...   },
...   {
...     $sort: { count: -1 }
...   },
...   {
...     $limit: 10
...   }
... ])
[
  { _id: 390718049, count: 260 },
  { _id: 423057302, count: 197 },
  { _id: 422425318, count: 178 },
  { _id: 34853175, count: 149 },
  { _id: 107488231, count: 137 },
  { _id: 55490072, count: 129 },
  { _id: 43829658, count: 112 },
  { _id: 197156399, count: 106 },
  { _id: 134772735, count: 96 },
  { _id: 27728102, count: 92 }
]
```

```
executionStats: {
  executionSuccess: true,
  nReturned: 7456963,
  executionTimeMillis: 415977,
  totalKeysExamined: 0,
  totalDocsExamined: 10459999,
```

- This is the MongoDB, the query for "top 10 reviewers with most reviews."

<table>
<tr>
<td>

```
hotel> db.reviews.aggregate([
...   {
...     $group: {
...       _id: "$reviewer_id",
...       count: { $sum: 1 },
...       lastCommentDate: { $max: "$date" }
...     }
...   },
...   {
...     $sort: { count: -1, lastCommentDate: -1 }
...   },
...   {
...     $limit: 10
...   }
... ])
[
  { _id: 390718049, count: 260, lastCommentDate: '2019-01-22' },
  { _id: 423057302, count: 197, lastCommentDate: '2022-06-03' },
  { _id: 422425318, count: 178, lastCommentDate: '2022-06-10' },
  { _id: 34853175, count: 149, lastCommentDate: '2022-05-30' },
  { _id: 107488231, count: 137, lastCommentDate: '2020-03-12' },
  { _id: 55490072, count: 129, lastCommentDate: '2019-11-30' },
  { _id: 43829658, count: 112, lastCommentDate: '2022-05-22' },
  { _id: 197156399, count: 106, lastCommentDate: '2022-05-21' },
  { _id: 134772735, count: 96, lastCommentDate: '2022-02-16' },
  { _id: 27728102, count: 92, lastCommentDate: '2022-05-23' }
]
executionStats: {
  executionSuccess: true,
  nReturned: 7456963,
  executionTimeMillis: 413139,
  totalKeysExamined: 0,
  totalDocsExamined: 10459999,
```

</td>
<td>

- This is the MongoDB query for “top 10 reviews with most reviews and recently commented”.

</td>
</tr>
<tr>
<td>

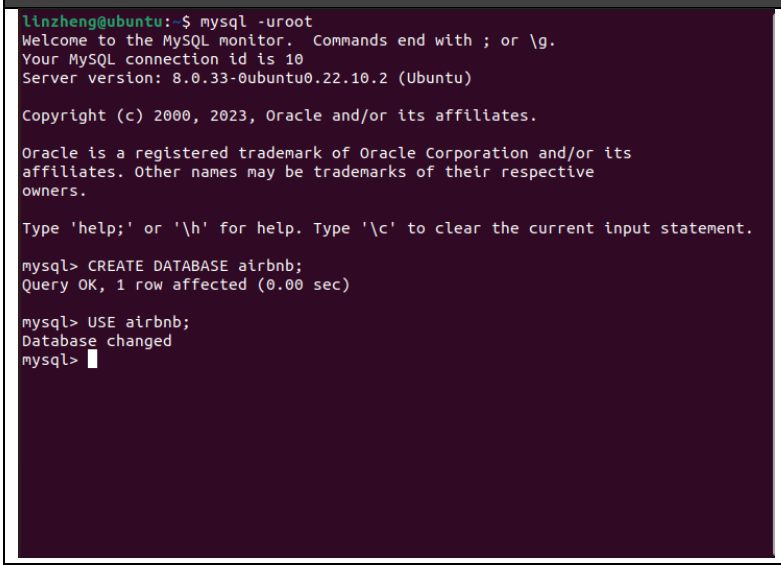
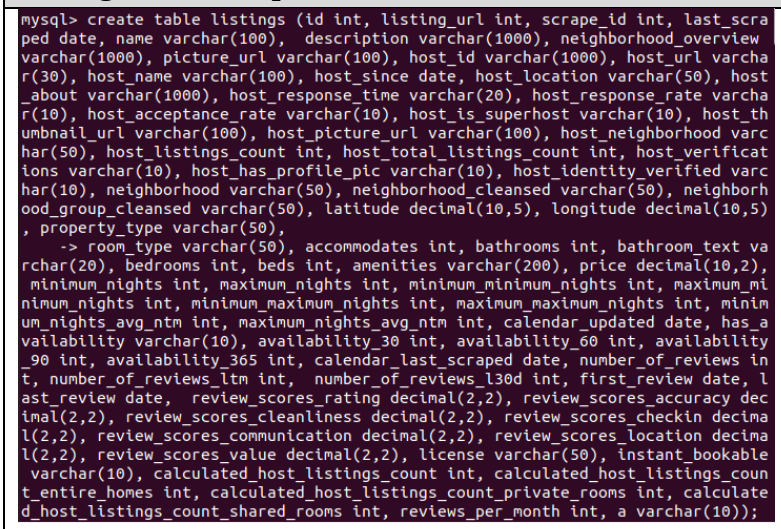
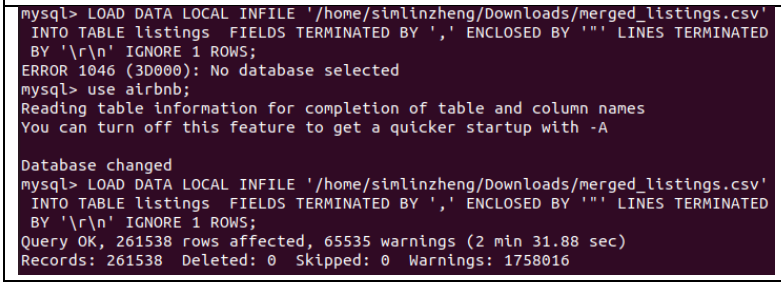
```
hotel> db.reviews.find( { reviewer_name: "Jessie", listing_id: { $gt: 30000000 } }, { reviewer_name: 1, reviewer_id: 1, _id: 0 }).limit(10);
[
  { reviewer_id: 271407793, reviewer_name: 'Jessie' },
  { reviewer_id: 64517667, reviewer_name: 'Jessie' },
  { reviewer_id: 319405334, reviewer_name: 'Jessie' },
  { reviewer_id: 111206424, reviewer_name: 'Jessie' },
  { reviewer_id: 178865406, reviewer_name: 'Jessie' },
  { reviewer_id: 151947858, reviewer_name: 'Jessie' },
  { reviewer_id: 21144671, reviewer_name: 'Jessie' },
  { reviewer_id: 262260878, reviewer_name: 'Jessie' },
  { reviewer_id: 33773793, reviewer_name: 'Jessie' },
  { reviewer_id: 12815209, reviewer_name: 'Jessie' }
]
executionStats: {
  executionSuccess: true,
  nReturned: 10,
  executionTimeMillis: 461,
  totalKeysExamined: 0,
  totalDocsExamined: 893760,
```

</td>
<td>

- This is the MongoDB query for “Find 10 reviewers named “Jessie” after the company has listed more than 30 million properties”.

</td>
</tr>
</table>

APPENDIX F: Implementation in MySQL

Command Line	Narration
	• MySQL was started without the need for Hadoop. • Database 'Airbnb' was created.
Listings – Data Import	
	• Table 'listings' was created, and data types were specified for the 75 attributes.
	• The merged_listings.csv was successfully imported.

```
| 45392 | https://www.airbnb.com/rooms/45392 | 2147483647 | 6/6/2022     | Cute
 Home in Mount Washington | <b>The space</b><br />Cute house in Mount Washingto
n near Griffith Park we have one room available (Red Bedroom) Continental Break
fast, closets are 90% empty, it  can accommodate 3 people max. for a profession
al person, working person, or grad student, no heavy drinking. Must like dogs -
 have one indoor dog. House is fully furnished, street parking. Close to shops.
 Easy access to freeways. <br /><br />Amenities: <br />- High speed wireless in
ternet <br />- Cable TV<br />- Full size bed<br />- Fully equipped kitchen, rea
dy to cook and serve <br />- TOWELS and SHEETS<br /><br />NO SMOKING inside but
 you can smoke outside on our patio. It's a safe and friendly neighborhood.<br
/><br />Bonuses: <br />- 24/7 support! Our resident' s knowledge is here for yo
u! <br /><br />Driving,<br />10 mins from China town, downtown LA.<br />5 mins
from Silver Lake and Eagle Rock and 20 mins from Pasadena.<br /><br />20 to 25
mins from Hollywood. <br /><br />Metro stop<br />Lincon/Cypress<br /><br />Clos
e |                    | https://a0.muscache.com/pictures/miso/Hosting-45392
/original/c453103b-6752-4655-b0dc-161e10bae98c.jp | 201514  | https://www.airbn
b.com/users/show/201514 | Olivia & Alexey | 14/8/2010  | Los Angeles, Californi
a, United States | We love outdoors, museums, art, film, travel, photography,
languages, hockey, soccer.
 | within a day       | 100%              | 86%                 | f
       | https://a0.muscache.com/im/users/201514/profile_pic/1344289344/origina
l.jpg?aki_policy=profile_small | https://a0.muscache.com/im/users/201514/profil
e_pic/1344289344/original.jpg?aki_policy=profile_x_med | Glassell Park     |
                1 |                          1 | ['email', 'phone'] | t
          | t                    |             | Mount Washington      | Ci
ty of Los Angeles         | 34.10632 | -118.22361 | Private room in home | Priv
ate room |           3 |         0 | 1 private bath |        1 |    2 | ["Hang
ers", "Free parking on premises", "Essentials", "Iron", "Air conditioning", "Fr
```

- The first record was displayed to ensure the CSV data has been populated accordingly into the table ‘listings’.

Listings – Query

```
mysql> SELECT id, host_is_superhost, price, review_scores_rating, host_response
_time, host_response_rate
    -> FROM listings
    -> WHERE host_is_superhost = 't'
    -> ORDER BY CAST(price as DECIMAL) ASC, review_scores_rating DESC,
    -> host_response_time ASC, CAST(host_response_rate AS DECIMAL) DESC
    -> LIMIT 10;
```

```
| id         | host_is_superhost | price   | review_scores_rating | host_respon
se_time | host_response_rate |
+------------+-------------------+---------+----------------------+------------
--------+--------------------+
| 2147483647 | t                 | $65.00  |                    5 | a few days
or more | 40%                |
|   48285229 | t                 | $123.00 |                    5 | a few days
or more | 40%                |
|   48096848 | t                 | $175.00 |                    5 | a few days
or more | 33%                |
|   39756724 | t                 | $70.00  |                    5 | a few days
or more | 33%                |
|   51303634 | t                 | $160.00 |                    5 | a few days
or more | 25%                |
|   48334110 | t                 | $180.00 |                    5 | a few days
or more | 25%                |
|   40861631 | t                 | $131.00 |                    5 | a few days
or more | 20%                |
|   41295559 | t                 | $196.00 |                    5 | a few days
or more | 20%                |
|   46313400 | t                 | $350.00 |                    5 | a few days
or more | 20%                |
|   48350489 | t                 | $499.00 |                    5 | a few days
or more | 20%                |
+------------+-------------------+---------+----------------------+------------
--------+--------------------+
10 rows in set, 65535 warnings (3.17 sec)

mysql>
```

- These are the top 10 cheapest super hotels with the besting rating and fastest response times and rates.

```
mysql> SELECT id, host_is_superhost, price, review_scores_rating, accommodates,
 instant_bookable
    -> FROM listings
    -> WHERE host_is_superhost = 't' AND accommodates = 10 AND instant_bookable
 = 't'
    -> ORDER BY CAST(price AS DECIMAL) ASC, review_scores_rating DESC
    -> LIMIT 10;
```

- These are the top 10 cheapest superhost hotels with the best rating, accommodate 10 persons, and can be booked instantly.

```
| id          | host_is_superhost | price     | review_scores_rating | accommoda
tes | instant_bookable |
+------------+------------------+-----------+----------------------+----------
---+-----------------+
| 2147483647 | t                 | $289.00   |                    5 |
 10 | t                |
|   50813114 | t                 | $501.00   |                    5 |
 10 | t                |
|   46127191 | t                 | $300.00   |                    5 |
 10 | t                |
|   35194836 | t                 | $448.00   |                    5 |
 10 | t                |
|   53576948 | t                 | $1,042.00 |                    5 |
 10 | t                |
|   50180898 | t                 | $7,201.00 |                    5 |
 10 | t                |
|   53396951 | t                 | $1,807.00 |                    5 |
 10 | t                |
| 2147483647 | t                 | $299.00   |                    5 |
 10 | t                |
|   51117850 | t                 | $471.00   |                    5 |
 10 | t                |
| 2147483647 | t                 | $1,375.00 |                    5 |
 10 | t                |
+------------+------------------+-----------+----------------------+----------
---+-----------------+
10 rows in set, 2570 warnings (0.63 sec)
```

```
mysql> SELECT COUNT(host_is_superhost)
    -> FROM listings
    -> WHERE host_is_superhost = 't';
+--------------------------+
| COUNT(host_is_superhost) |
+--------------------------+
|                    82611 |
+--------------------------+
1 row in set (0.65 sec)
```

- Total number of superhost in the dataset.

```
mysql> SELECT neighborhood_group_cleansed, COUNT(neighborhood_group_cleansed) A
S count
    -> FROM listings
    -> GROUP BY neighborhood_group_cleansed
    -> ORDER BY count DESC
    -> LIMIT 10 OFFSET 1;
+----------------------------+-------+
| neighborhood_group_cleansed | count |
+----------------------------+-------+
| City of Los Angeles         | 18386 |
| Manhattan                   | 15855 |
| Brooklyn                    | 13954 |
| Other Cities                | 13670 |
| Maui                        |  8596 |
| Honolulu                    |  7842 |
| Hawaii                      |  6065 |
| Queens                      |  5824 |
| Kauai                       |  3842 |
| Unincorporated Areas        |  3271 |
+----------------------------+-------+
10 rows in set (0.72 sec)
```

- The top 10 neighbourhoods in the dataset.

```
mysql> SELECT host_location, COUNT(host_location) AS count
    -> FROM listings
    -> GROUP BY host_location
    -> ORDER BY count DESC
    -> LIMIT 10;
+----------------------------------------+-------+
| host_location                          | count |
+----------------------------------------+-------+
| US                                     | 29569 |
| New York, New York, United States      | 24763 |
| Los Angeles, California, United States | 15247 |
| Austin, Texas, United States           |  9178 |
| San Diego, California, United States   |  8217 |
| San Francisco, California, United States |  7297 |
| Las Vegas, Nevada, United States       |  6083 |
| Chicago, Illinois, United States       |  4817 |
| New Orleans, Louisiana, United States  |  4368 |
| Brooklyn, New York, United States      |  4335 |
+----------------------------------------+-------+
10 rows in set (1.14 sec)
```

- Top 10 host locations based on the number of listings in these host locations.

```
mysql> SELECT property_type, COUNT(property_type) AS count, accommodates
    -> FROM listings
    -> GROUP BY property_type, accommodates
    -> ORDER BY count DESC, accommodates DESC
    -> LIMIT 5;
+---------------------+-------+--------------+
| property_type       | count | accommodates |
+---------------------+-------+--------------+
| Entire rental unit  | 21200 |            2 |
| Entire rental unit  | 17531 |            4 |
| Private room in home | 14053 |            2 |
| Entire condo        | 13474 |            4 |
| Entire home         | 11241 |            6 |
+---------------------+-------+--------------+
5 rows in set (1.22 sec)

mysql>
```

- Top 5 most common property types followed by the number of persons accommodated.

```
mysql> SELECT host_name, host_total_listings_count
    -> FROM listings
    -> WHERE host_location IN ('Tijuana', 'Baja', 'California', 'Mexico')
    -> AND host_total_listings_count = 3
    -> LIMIT 10;
+-----------+---------------------------+
| host_name | host_total_listings_count |
+-----------+---------------------------+
| Cathy     |                         3 |
| Cathy     |                         3 |
| Cathy     |                         3 |
| Jordan    |                         3 |
| Jordan    |                         3 |
| Jordan    |                         3 |
| Gino      |                         3 |
| Gino      |                         3 |
| Gino      |                         3 |
| Gloria    |                         3 |
+-----------+---------------------------+
10 rows in set (0.18 sec)

mysql>
```

- These are the 10 hosts located at the specified locations and have a total listing of 3.

Reviews – Data Import

```
mysql> load data infile '/var/lib/mysql-files/review.csv' INTO TABLE reviews FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\n' IGNORE 1 LINES (@listing_id,@ID,@date,@reviewer_id,@reviewer_name,@comments) SET listing_id=IF(listing_id='','0',@listing_id),ID=IF(ID='','0',@ID),date=IF(date='','0',@date),reviewer_id=IF(reviewer_id='','0',@reviewer_id),reviewer_name=IF(reviewer_name='','0',@reviewer_name),comments=IF(comments='','0',@comments);
Query OK, 10460010 rows affected (8 min 29.66 sec)
Records: 10460010  Deleted: 0  Skipped: 0  Warnings: 0
```

```
mysql> Select * from reviews Limit 1;
+------------+-------+------------+-------------+---------------+----------+
| listing_id | id    | date       | reviewer_id | reviewer_name | comments |
+------------+-------+------------+-------------+---------------+----------+
| 109        | 449036 | 2011-08-15 | 927861      | Edwin         | "0       The host canceled my reservation the day before...
1          Olivia helped me with directions to arrive & l...
2          Olivia was incredibly helpful and knowledgable...
3          Me and two friends stayed for four and a half ...
4          i had a wonderful stay. Everything from start ...
```

- Merged_reviews.csv was uploaded and imported into the ‘reviews’ table without any exception.
- The first row was displayed to ensure data were imported correctly.

Reviews – Queries

```
mysql> SELECT listing_id, COUNT(comments) as reviews_count
    -> FROM reviews
    -> GROUP BY listing_id
    -> ORDER BY reviews_count DESC
    -> LIMIT 10;
+------------+----------------+
| listing_id | reviews_count  |
+------------+----------------+
| 44799007   |           2890 |
| 29819757   |           2295 |
| 34071681   |           2020 |
| 46534      |           1857 |
| 8357       |           1689 |
| 6652991    |           1520 |
| 42409434   |           1512 |
| 8356380    |           1438 |
| 35158303   |           1438 |
| 815639     |           1419 |
+------------+----------------+
10 rows in set (1 min 26.10 sec)
```

- These are the listings that have the most reviews.

```
mysql> SELECT listing_id, COUNT(comments) as reviews_count, MAX(date) as Latest_review_date FROM reviews GROUP BY listing_id ORDER BY reviews_count DESC, Latest_review_date DESC LIMIT 10;
+------------+---------------+--------------------+
| listing_id | reviews_count | Latest_review_date |
+------------+---------------+--------------------+
| 44799007   |          2890 | 2022-05-31         |
| 29819757   |          2295 | 2022-06-06         |
| 34071681   |          2020 | 2021-11-22         |
| 46534      |          1857 | 2022-06-21         |
| 8357       |          1689 | 2022-06-19         |
| 6652991    |          1520 | 2022-06-14         |
| 42409434   |          1512 | 2022-05-31         |
| 8356380    |          1438 | 2022-06-10         |
| 35158303   |          1438 | 2022-06-06         |
| 815639     |          1419 | 2022-06-03         |
+------------+---------------+--------------------+
10 rows in set (1 min 11.09 sec)
```

- These are the listings that have the most recent reviews.

```
mysql> SELECT reviewer_name, COUNT(comments) as reviews_count
    -> FROM reviews
    -> GROUP BY reviewer_name
    -> ORDER BY reviews_count DESC
    -> LIMIT 10;
+---------------+---------------+
| reviewer_name | reviews_count |
+---------------+---------------+
| Michael       |         90013 |
| David         |         82766 |
| Sarah         |         67515 |
| John          |         65375 |
| Jennifer      |         61798 |
| Jessica       |         60146 |
| Chris         |         52349 |
| Daniel        |         51451 |
| Emily         |         48480 |
| Andrew        |         47132 |
+---------------+---------------+
10 rows in set (1 min 30.42 sec)
```

- These are the 10 reviewers that gave the most reviews.

```
mysql> SELECT reviewer_name, COUNT(comments) as review_count, MAX(date) as latest_review
    -> FROM reviews
    -> GROUP BY reviewer_name
    -> ORDER BY review_count, latest_review DESC
    -> LIMIT 10;
+--------------------------------------------------------------------------+--------------+--------------------+
| reviewer_name                                                            | review_count | latest_review      |
+--------------------------------------------------------------------------+--------------+--------------------+
| Michael,"""0          The host canceled my reservation the day befor... |            0 | 2012-10-01         |
| NULL                                                                     |            0 |  dtype: object""" |
| Diondre M                                                                |            1 | 2022-08-13         |
| Sareaza                                                                  |            1 | 2022-08-13         |
| Tumeakia                                                                 |            1 | 2022-08-12         |
| Le'Ah                                                                    |            1 | 2022-08-12         |
| Krillin                                                                  |            1 | 2022-08-11         |
| 雷风                                                                     |            1 | 2022-08-11         |
| DJ-Fredy                                                                 |            1 | 2022-08-11         |
| Ahnia                                                                    |            1 | 2022-08-11         |
+--------------------------------------------------------------------------+--------------+--------------------+
10 rows in set (1 min 36.45 sec)
```

- The output shows the 10 names that gave the most recent reviews on the listings they have rented.

```
mysql> SELECT listing_id, reviewer_name
    -> FROM reviews
    -> WHERE listing_id > 30000000 AND reviewer_name ='Jessie'
    -> ORDER by listing_id DESC
    -> LIMIT 10;
+--------------------+---------------+
| listing_id         | reviewer_name |
+--------------------+---------------+
| 665435789453472463 | Jessie        |
| 640397079726524452 | Jessie        |
| 621812807319459071 | Jessie        |
| 621719722497608992 | Jessie        |
| 620874447347342376 | Jessie        |
| 620874447347342376 | Jessie        |
| 620250707287826722 | Jessie        |
| 606385400662155205 | Jessie        |
| 605739997052461438 | Jessie        |
| 602323583400557768 | Jessie        |
+--------------------+---------------+
10 rows in set (12.62 sec)
```

- The output shows the 10 reviewers named ‘Jessie’ after the company have listed more than 30m Airbnb

```
mysql> SELECT table_name, round(((data_length + index_length) / 1024 / 1024), 2) AS "Table Size (MB)"
    -> FROM information_schema.TABLES
    -> WHERE table_schema = 'hotel'
    -> AND table_name = 'reviews';
+------------+-----------------+
| TABLE_NAME | Table Size (MB) |
+------------+-----------------+
| reviews    |         8911.00 |
+------------+-----------------+
1 row in set (0.00 sec)
```

- Output of the size of the ‘reviews’ table in MySQL