Roshan Giri

18406492

TECHNICAL REPORT

**CSY3024 -- Databases 3**

Table of Contents

[Learning Diary Reflection 2](#_Toc43554756)

[Queries 3](#_Toc43554757)

[1. Show all the EPL teams involved in the season. 3](#_Toc43554758)

[2. How many matches were played on Mondays? 4](#_Toc43554759)

[3. Display the total number of goals “Liverpool” had scored and conceded in the season. 5](#_Toc43554760)

[4. Who refereed the most matches? 6](#_Toc43554761)

[5. Display all the matches that “Man United” lost. 6](#_Toc43554762)

[Big Data Analytics and the Role of NoSQL Databases 8](#_Toc43554763)

[*I.* *INTRODUCTION* 8](#_Toc43554764)

[*II.* *NoSQL Over SQL* 8](#_Toc43554765)

[*III.* *NoSQL DATA MANAGEMENT MODELS* 9](#_Toc43554766)

[*IV.* *NoSQL DATABASE CHARACTERISTICS* 10](#_Toc43554767)

[*V.* *BIGDATA CHALLENGES* 10](#_Toc43554768)

[*VI.* *DATA ANALYSIS APPROACHES* 11](#_Toc43554769)

[*VII.* *OPEN ISSUES IN BIG DATA* 12](#_Toc43554770)

[*VIII.* *CONCLUSION* 13](#_Toc43554771)

[*IX.* *REFERENCES* 13](#_Toc43554772)

[Appendix 14](#_Toc43554773)

[A. Learning Diary 14](#_Toc43554774)

[1. WEEK 13 - Cypher Query Language 14](#_Toc43554775)

[2. WEEK 16 – Introduction to MongoDB 23](#_Toc43554776)

[3. WEEK 17 – MongoDB Continuing 27](#_Toc43554777)

[4. WEEK 21 – Database Transaction Management 43](#_Toc43554778)

[B. Queries 45](#_Toc43554779)

[1. Show all the EPL teams involved in the season. 45](#_Toc43554780)

[2. How many matches were played on Mondays? 45](#_Toc43554781)

[3. Display the total number of goals “Liverpool” had scored and conceded in the season. 45](#_Toc43554782)

[4. Who refereed the most matches? 46](#_Toc43554783)

[5. Display all the matches that “Man United” lost. 46](#_Toc43554784)

[REFERENCES 47](#_Toc43554785)

# Learning Diary Reflection

This part provides information about my overall reflection on learning of this module which includes lab exercises, activities that I have learnt during my time via lectures. This part discusses on the overall course of the term. The knowledge and skills that I have gained over the course of this year is reflected in the diary log. There are still many things that need to be improved over the interval of time. I have almost completed the learning exercises that were provided to us.

To sum up my individual performance reflecting my work that includes the lectures, lab exercises, I have gained knowledge and skills for using the non-relational database management system i.e. NoSQL which includes the graph database (Neo4j) and the document-based database (MongoDB). I have developed critical understanding towards the NoSQL, which is better than the old relational database management system.

The learning diary logs are completed and I have had fewer problems while completing the exercises but most of the learning activities were completed without having any problems. I had problems doing the lab exercise regarding to work mongo DB with php and java. I also encounter fewer problems using mongo DB aggregate functions but with more practice and hard work I have developed critical understanding to solve the problems that stumble across my path.

During the time of course, the knowledge and skills that I have gained are:

* Better understanding of NoSQL and its advantages over the relational database system such as: scalability, high-performance, stores structured and un-structured data, less expensive, schema less and so on.
* Limitations of the old relational database management system regarding big data gathered from various sources
* Deep Understanding towards the graph model database of NoSQL and the advantages of graph database for big data companies
* Deep Understanding towards the document-based database model of NoSQL and advantages of this model such as high-performance, distributed architecture for storing of data, horizontal scaling and so on.
* The syntax development of NoSQL is appraisable because it does not use the join clauses, which makes read and write faster than the old relational database model.
* Developed knowledge and skills for performing smoothly in both graph model and document- based model of NoSQL
* Querying the graph model and document-based model database using various clauses, operators and so on.
* Gained knowledge and understanding on using MongoDB with PHP and Java.
* Better understanding the database transaction management.

There is still some gap for improving the knowledge and skills toward the NoSQL database management system which I will learn and improve over the course of time.

# Queries

We are provided with the data of English Premiere League 2018/19 season in the JSON file. It is a single document that consists of large data of the matches between home team and away team and different other factors involving in the matches 2018/19. Basically, it contains all the matches history of the EPL -2018/19.

To get started with querying with file, we have to know how the EPL works. There are 20 teams involved in the EPL every year and these teams have to compete against each other for at least once in-home ground and once in away ground over the season. Basically, every team have to play 38 matches in order to complete the EPL season of respective year. There are three results involved in match i.e. either a team loses, wins or the match is tied. If the team win a match then they would get 3 points, draws a match then they would get 1 point and if the loses they game then they would get 0 points. On the basis of the points the table are ordered in the EPL and if the points are equal then they would consider other factors such as goals conceded, goal differences and so on.

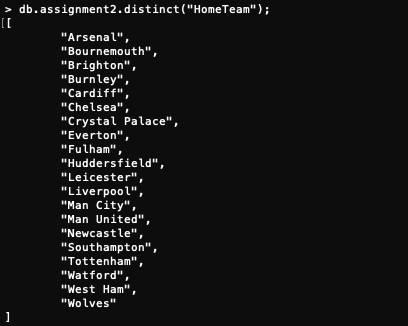
In order to query and analyze the document, we have to import the JSON file epl\_1819.json in collection assignment2.



## Show all the EPL teams involved in the season.

db.assignment2.distinct("HomeTeam");

There are 20 teams in the EPL which is listed below by the above query. This query uses distinct in the collection assignment 2 field HomeTeam. This resulted in showing distinct teams involved in EPL 2018/19.

****

## How many matches were played on Mondays?

db.assignment2.find().forEach(function(day) {

day.Date = new Date(day.Date);

db.match.save(day);

});

db.assignment2.aggregate(

[

{

$project:

{

Date: "$Date",

day: { $dayOfWeek: "$Date" },

}

},

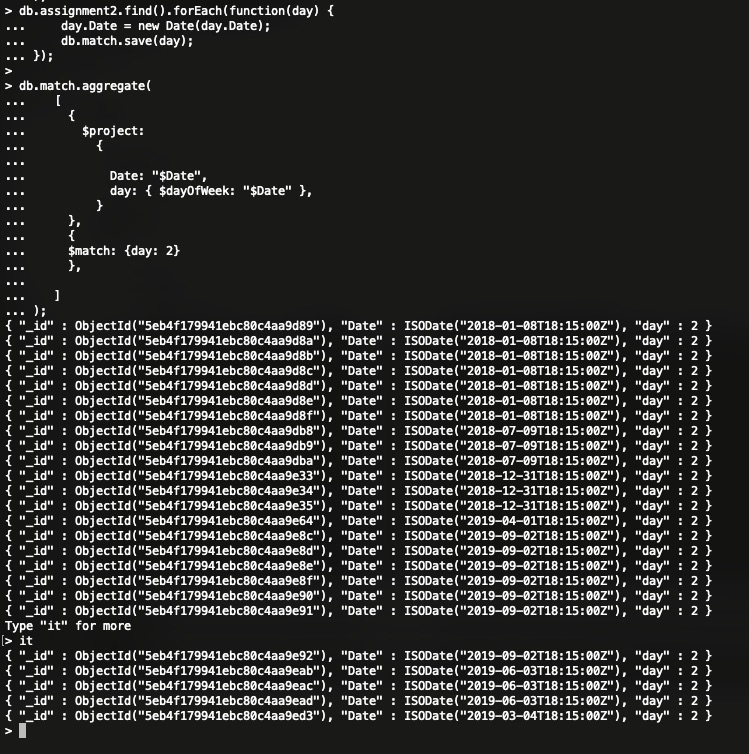
{

$match: { day: 2 }

},

]

);



I used the aggregation pipeline operator $dayOfWeek that returns day of week for Date. It returns the number of days of the week i.e. Sunday =1 whereas Saturday = 7. In order to find out, how many matches were played in EPL 2018/19, we have to use this operator that finds the day of week in the date. I used the day as 2, so it returns all the matches that were played in day 2 i.e. Monday. The $match returns all the matches that were played on monday.

## Display the total number of goals “Liverpool” had scored and conceded in the season.

db.assignment2.aggregate(

{ $match: { $or: [ { "HomeTeam": "Liverpool" }, { "AwayTeam": "Liverpool" } ] } },

{ "$project":

{

Club: "Liverpool", \_id: 0, "HomeTeam": 1, "AwayTeam": 1,

Total\_Goal\_Scored: { $cond: { if: { $eq: [ '$HomeTeam', 'Liverpool' ] }, then: '$FTHG', else: '$FTAG' } },

Total\_Goal\_Conceded: { $cond: { if: { $eq: [ '$AwayTeam', 'Liverpool' ] }, then: '$FTHG', else: '$FTAG' } }

}

},

{ $group:

{

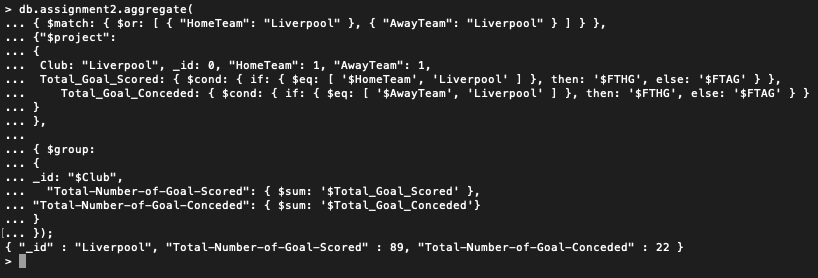
\_id: "$Club",

"Total-Number-of-Goal-Scored": { $sum: '$Total\_Goal\_Scored' },

"Total-Number-of-Goal-Conceded": { $sum: '$Total\_Goal\_Conceded'}

}

});



In this query we have to find the number of goals Liverpool had conceded and scored in the season. So, in order to find the number of goals Liverpool had scored and conceded, we have to calculate the number of full-time home goal (FTHG) and full-time away goals (FTAG) in HomeTeam and AwayTeam respectively. In the above query, we used the conditional expression $cond to calculate the number of goals scored and conceded by Liverpool. If Liverpool is the HomeTeam than we take the full-time home goal (FTHG) of the HomeTeam else we take the full-time away score to calculate the number of goals scored by Liverpool in the season. Similarly, the goals conceded can be calculated. At last, the sum of total goal scored and total goals conceded are calculated which shows that Liverpool had scored 89 goals and conceded 22 goals.

## Who refereed the most matches?

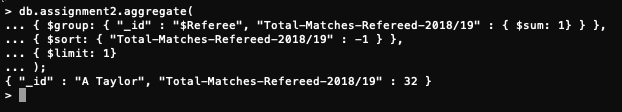
db.assignment2.aggregate(

{ $group: { "\_id" : "$Referee", "Total-Matches-Refereed-2018/19" : { $sum: 1 } } },

{ $sort: { "Total-Matches-Refereed-2018/19" : -1 } },

{ $limit: 1 }

);

****

In this query we used aggregate operation in order to see which referee refereed most of the matches in 2018/19. In this operation the group stage groups the document by referee field to calculate the sum of the total match refereed by a referee and sort the document displaying the value by using limit because we want only one result.

## Display all the matches that “Man United” lost.

db. assignment2.find( {

$or: [

{ "AwayTeam": "Man United", "FTR": "H" },

{ "HomeTeam": "Man United", "FTR": "A" }

],

});

****

The operator $or is used for logical operation. It is used in an array where there are two or more expression and selects at least one operation that satisfy the document. In the above query we used the $or operator because Man United play games both in home and away so there are two results i.e. one result in home ground and other result in away ground which means Man United can lose games in home and away. If they lose the game in home then the full-time result (FTR) will be “A” and if they lose the game in away ground then the FTR will be “H”. In this way we would be able to count the total number of matches that Man United lost during home and away game. In away game Man United lost to Brighton, West Ham, Man City, Liverpool, Arsenal, Wolves and Everton whereas in home game Man United lost to Tottenham, Man City and Cardiff.

Therefore, Man United lost a total of 10 matches which is displayed

# Big Data Analytics and the Role of NoSQL Databases

## *INTRODUCTION*

Data analytic can be defined as a process of finding meaningful or relevant information from the stored data to discover hidden patterns or potential. Big data is an expression, meaning huge volume of structured and unstructured data or information that is so huge. These types of data are difficult to process using the traditional database and software techniques. (Han Hu et al., 2014) According to the attributive definition Big data is characterized as “4Vs” meaning the volume, variety, velocity and value. The big data system architecture is divided into four parts i.e. data generation, data acquisition, data storage and data analytics.

(Han Hu et al., 2014) The first phase of bigdata is the generation of data. The data is generated from various sources such as social media, hospitals, business, education, communication, entertainment, transportation and various other sources. After the data is generated, the generated data is obtained from these sources and useful information is obtained in the data acquisition phase. After the data acquisition phase, the data is stored in a platform where it is organized and the organized data is used for exploring and extraction. After the third phase of data storage, the data is analyzed to extract required value or information and to find meaningful patterns and knowledge in the stored data. The data are analyzed into three different levels i.e. Descriptive Analytics (Concludes what happened on the basis of the historical data), Predictive Analytics (Predicts the future trend on the basis of concurrent data and testing data) and Prescriptive Analytics (addresses decision making and efficiency on the present situation) (Han Hu et al., 2014).

SQL stands for Structured Query Language. It is a relational database management system which uses SQL syntax for performing data operations such as storing, querying and retrieving data whereas NoSQL stands for non-relational database management system. NoSQL does not require fixed scheme than that of SQL. This schema does not use join queries and is easy to scale up.

This non-relational database management system can store the data in the form of structured, semi- structured, polymorphic and unstructured.

## *NoSQL Over SQL*

(Venkatraman et al., 2016) SQL was the best choice for the business to operate in the past. But as time passed down, there was too much data volume generated by the users, systems, sensors, organizations such as google, amazon, and so on due to which the querying of data became very hard and it loses its efficiency. The data was

increasing day by day and the complexity of data was increasing in internet, social media sites and various other systems which became difficult to manage. The managing of data was very hard to operate in the relational database. There were a lot of limitations in the relational database in the big data analytics. To solve the problem the cluster-based architecture was introduced

but the SQL database system was not designed to work with clusters which results in alternate solution for the problem. The SQL servers are weak for managing the memory footprint, security risks and in performance (Moniruzzaman and Hossain, 2013).

To overcome the limitations of SQL, NoSQL database was introduced which manages data flexibly and can be scaled up horizontally. The storage and retrieval of data in NoSQL is easy because of its structure than that of SQL. The database of the NoSQL can be scaled up horizontally with the help of clusters and servers that are cheap. The data can be managed automatically and faulty recovery can be done easily in the NoSQL database system which results in saving money. The limitations of the SQL are overcome by the non-relational database system along with addition of extra features. NoSQL does not provide tables that has fixed- column and works as BLOBs.

(Venkatraman et al., 2016) There are many advantages of NoSQL over SQL such as it does not require relational mapping and normalization of data. This database does not have complex queries, joins and ACID properties. The SQL have one DBMS products whereas the NoSQL have four categories databases i.e. key-value pair based, column-oriented graph, graph based and document-oriented. These four categories have their own attributes and limitations. The schema of non-relational database is dynamic, data can be stored together and allows modification of schemas whereas in the relational database the schemas are pre-defined with relationship with various tables of the entire schema and any alter or update query can alter the database. The non- relational database supports all types of data such as structured, semi-structured, unstructured and so on whereas in the relational database the data are stored in fixed row and columns in a table which has different relationship such as primary, unique, foreign and so on. The non- relational database can query the data efficiently due to object-oriented APIs whereas the relational database uses SQL for the data manipulation. The platform for operating the non-relational database system are mongo dB, Neo4j, HBase, Couchbase and so on whereas the relational database uses oracle, MySQL and so on. These differences between the relational database and non-relational database made the big companies to shift from SQL Language to NoSQL language. All of the limitations of the SQL were solved by the NoSQL language along with additional features.

## *NoSQL DATA MANAGEMENT MODELS*

(Venkatraman et al., 2016) and (Moniruzzaman and Hossain, 2013) There are many NoSQL databases but all of the databases fall over four data models i.e. Key- Value Store, Column Oriented Store, Document Store and Graph Store databases. Each of these models have their own attributes but the data models are different. (Han Hu et al., 2014) These models are highly used in the management of big bulk data for analytics.

*a. Key-Value Store Database –* It is a simple database which is efficient and powerful. In this database the data are stored in two parts i.e. key and value. The key is represented by a string whereas the value is represented by a data which in combination create a Key-Value. This database allows to store data that is structured or unstructured. The keys are used for retrieving and searching data. This database model is highly concurrent and has a large scalability. However, this model is not so consistent. These types of database are used for online shopping carts and used for online shopping websites where the user sessions are required to store. (Moniruzzaman and Hossain, 2013) These database models are incredibly fast, scalable for retrieving values for applications which manages the profile of users, sessions or retrieving the name of the product. Some of the example of this database model are DynamoDB of Amazon, Apache’s Cassandra, Voldemort of LinkedIn and so on. One of the main advantages of this model is that it inserts and read rate are higher than that of old RDBMS.



Figure 1. Example of Key-Value Store Database (Moniruzzaman and Hossain, 2013)

*b. Column Oriented Store Database* – (Venkatraman et al., 2016) In this model of database the columns are defined in each and every row. It does not predefine the table structure that has uniform size columns for each and every row. The Column Oriented store database have two-level aggregate structure i.e. it has a key and a row aggregate and is a group of columns. In this database model whatever column can be added in any row as well as the row can have different columns. It can also mean that each row has different columns that are being stored. This database model can also store data tables as segment of columns of data. In this model the data can be observed as row-oriented and as column- oriented. In the row-oriented the row is aggregate and, in the column-oriented the column is defined as a record type. This model has a high-performance rate, scalability rate because the data are stored in appropriated architecture. This model is used for data mining purpose and in the analysis of big data applications (Moniruzzaman and Hossain, 2013). Example of this database model are Facebook’s Cassandra, Apache HBase, Google Bigtable and high table data structure.



Figure 2. Example of Wide-Column Store Database (Moniruzzaman and Hossain, 2013)

*c. Document Store Database –* (Moniruzzaman and Hossain, 2013) The document store database is the database that are designed to manage and stored data in documents. These documents are encoded and represented in the form of XML, JSON or BSON. This model extends the basic key-value database and stores the value in column in document. The value in the document are semi-structured data that has specific name and value pairs. In this document store database thousands of attributes can be stored in a single column. In this model of database, the key and values can be searched throughout the document databases. This model is suitable for storing big data collection of blogs, texts, emails and so on. Some of the examples of this model are mongo DB, Apache, CouchDB and so on. This model of database is efficient, consistent, persistence but lack in resources.



Figure 3. Example of Document Store Database (Moniruzzaman and Hossain, 2013)

*d. Graph Store Database-* A Graph Store database is a database which stores the entities along with the relation to the entities. The entities are stored in the form of node along with the relationship and edges. In this model the node and edge represent a unique identifier and the node relationship is given by an edge. When comparing this model with the old relational database system the graph store database is multi- relational in nature where the old relational database is connected loosely. This store database model has high performance rate and is very fast which is why it is used in social networks, engineering, structural data and so on. Some of the example of this database is Neo4j, Titan and so on. Modern companies such as LinkedIn, Facebook, Instagram and so on produce a large amount of data which is why this database model is used because it is highly scalable and has a parallel structure and can process a large data without any limitations in the system.

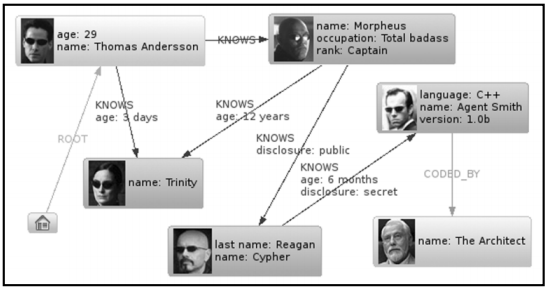


Figure 4. Example of Graph Store Database (Moniruzzaman and Hossain, 2013)

## *NoSQL DATABASE CHARACTERISTICS*

(Moniruzzaman and Hossain, 2013) In the old relational database management system, the integrity of data is maintained on the basis of the transactions and is consistent for the data management system. The old relational database system had ACID property which stands for Atomicity, Consistency, Isolation and Durability. So, while shifting from the transferable nature (ACID) system arises complications i.e. CAP theorem which can also be known as Brewer’s theorem. (Moniruzzaman and Hossain, 2013), (Han et al., 2011) and (Kambatla et al., 2014) This theorem said that it is impossible for a data store to provide two out of there characteristics i.e. Consistency, Availability and Partition Tolerance. The store data can only provide any two characteristics among the three.

Consistent - In this characteristic, the data should remain consistent throughout the execution of operations which means that it should be able to contain data if read request is done.

Availability - The data should be available at any time of request without facing any problems of downtime.

Partition Tolerance - In partition tolerance the system should be able to function even though the communication is unstable throughout the servers.

(Moniruzzaman and Hossain, 2013), (R. Cattell et al., 2011) and (Han et al., 2011) The NoSQL database have loosened the characteristics Consistent of the CAP theorem in order to achieve Availability and Partition Tolerance. This is the reason why the NoSQL system emerged to be a BASE system which stands for Basically, Available, Soft-state, Eventually consistent. This system does not operate as the old relational database rather this system introduced restraint on the data model in order to achieve partitioning schemes. This is the reason why the NoSQL databases are used in bulk of data processing, analytics of data, storing data into larger volumes and so on. The transfer from old system to the non-relational system was expensive but for new companies they don’t have to transfer their data. Today, NoSQL is less expensive and has more scalability for storing large amount of data. Facebook, Instagram, LinkedIn, Google uses the non-relational database to store their data because of its scalability and availability characteristics. NoSQL is also used for many other purposes such as processing large amounts of data, analytics of big structured or unstructured data and many other purposes.

## *BIGDATA CHALLENGES*

(Han Hu et al., 2014) Bigdata are growing rapidly day by day which is a challenging factor in the modern world. The data are collected in huge amount from various medium such as communication, transportation, education, entertainment and many other mediums. Millions of data from Facebook, twitter is collected in the form of texts, images, videos and so on. To manage these bigdata comes with great responsibility and challenges. (Malhotra et al., 2017) Some of the challenges in Big data are represented below:

*a. Collection and Integration*- Collection and Integration of data is very difficult. There are number of structured, semi-structured, unstructured data. The data should be collected from a various source and the integration is difficult. Facebook produces big data in millions in the form of messages, photos, videos and many other sources making the collection and integration of data extremely difficult (Han Hu et al., 2014).

*b. Management and Storage-* The collected huge amount of data needs to be stored and managed properly. One of the other challenging problem is to manage the collected big data and storing in proper system. The management and storage of big data should be done properly so that the big data can be accessed quickly. Meaning the big data should be stored and managed properly so that the information can be analyzed and accessed quickly without compromising the security and privacy of the data (Labrinidis and Jagadish, 2012).

*c. Designing, visualization, development and prediction-* One of the major challenges is designing, visualization, development and prediction of big data. The big data analytics should be done effectively in various level actual time counting the development, designing visualization and prediction in-order to improve the outcome of the system. This will help in decision making and gaining information.

## *DATA ANALYSIS APPROACHES*

(Han Hu et al., 2014) and (Malhotra et al., 2017) As the traditional relational database system is not appropriate for the analysis of huge data. The appropriate analysis of large data is done by NoSQL, Hbase, Hadoop framework, HIVE, Sqoop, and so many other programs. These solutions for the analysis of big data is appropriate because it is cost effective, scalable, elastic. NoSQL provides a distributed server of management of file systems (Howard et al., 1988) for the storing of big data (R. Cattell et al., 2011) Similarly, Hadoop Framework provides parallel processing of large amount of data along with scalable system, blunder receptive and so on (Han Hu et al., 2014) and (Malhotra et al., 2017). Hadoop is a data analysis open source programming framework which is used for mainly two purposes. The two purposes are for storing big data and processing these big data (Herodotou et al., 2011).

*1. Hadoop Framework*

(Malhotra et al., 2017) The Hadoop programming framework is used for processing of large distributed datasets crosswise array. The core component of this framework is the Hadoop Distributed File System (HDFS) along with the MapReduce programming model. Hadoop framework are used in various activities such as data mining, analytics of marketing, web surfing and processing of image, audio, video. It is used for analysis of stocks in marketing and finance. There are many features of this framework. Some of them are:

*a. Scalability:* The Hadoop framework is the base for the scalable hardware. In this framework the infrastructure of the system can be scaled up and down. If the data is big then the infrastructure can be scaled up and vice-versa. By scaling the infrastructure, the data formats would not change. According to the hardware changes the data and calculation activity is naturally adjusted.

*b. Cost Effective:* The Hadoop framework is cost effective due to it parallel computation. The big data generated day by day can be maintained by parallel computation with the Hadoop framework. Due to its server capacity, it can commode a large amount of machine with low price tags which helps in addition of storage and convert power to the entire array system. This is why the storage of data in Hadoop is cheaper and cost effective.

*c. Flexible:* The Hadoop Framework is flexible because it can consume any data from various number of sources. Therefore, making the system flexible due to the absorption of data from any number of resources.

*d. Fault Tolerance*: This framework can recover data if the data is lost by a computation error or a network failure therefore, making the system fault tolerance.

*2. MapReduce Framework*

(Han Hu et al., 2014) and (Malhotra et al., 2017) This framework uses algorithm for processing of big data counting for their generation. This framework is a combination of map procedures and reduce procedures. The map procedure processes the big data and finally generates the Key-Value pair. After this process, the output is sent to the reduce procedures where it produces the final outcome. In this framework the processing of data is done via parallel form. There are number of programming languages that are defined from the MapReduce framework athenaeum. Apache Hadoop is one of the examples of the implementation of the MapReduce framework libraries. Besides, this framework has very high fault tolerance i.e. the data are not lost if there is computation error or network failure problem (Malhotra et al., 2017). This framework is cost efficiency and is faster and better than relational database management system. This frame work is mostly suitable for cloud status system. It is also applicable for joining and sorting of the datasets in the database. Mainly this is important for searching of pattern-based items (Dean and Ghemawat, 2008).

*3. HIVE*

Hive is also one of the major big data analysis solution. It was first introduced by Facebook because the number of Facebook users were growing rapidly resulting in production of big amount of data. Millions gigabyte of data are produced as time passed on and the management of these data cannot be done by the old relational database management system. So, to solve the problem of big data Facebook started using Hive in 2006 (Malhotra et al., 2017).

This system is a data repository and is built inside a Hadoop file system. This system operates big data as the traditional database could not solve the problem. HIVE provides a podium for customers to be able to use queries called HiveQL which as similar to the SQL. The Hive Query is only used for managing structure data only. HIVE removes the complexity of Hadoop because in HIVE we do not have to learn MapReduce which was compulsory to operate in Hadoop programming framework. In this system the user does not have to learn the JAVA programming language along with the application platform interface of the Hadoop programming language.

There are many advantages of Hive such as it is used for mining of data. HIVE uses log for processing of data which results in dividing the data in the form of table and becomes easy for analyzing. In HIVE large documents are indexed. Since, HIVE operates in queries of its own known as HiveQL, the old user using SQL queries would be able to operate properly in this system. Since, Hive is a data repository and is built inside the Hadoop file system because of its reliability, scalability, flexibility. The stored data is in the form of hive tables so the assessment of data is easy (Han Hu et al., 2014) and (Malhotra et al., 2017).

However, there are some disadvantages of HIVE. It can only process structured data which means it cannot process the data which is semi- structured or unstructured.

*4. NoSQL*

As discussed earlier NoSQL means not only structured query language. This is designed for large company who generates huge amount of data in a single day. The large companies used this database system because of its scalable platform. This non-relational database does not require any functionalities of the traditional database such as join operations. In this non-relational database, the data are stored horizontally. This database uses the CAP theorem in which only two functionalities can be achieved by losing one functionality. Most of the databases loses the consistency in order to gain the availability and partition tolerance of the data. The NoSQL database also uses the availability and partition tolerance of the CAP theorem. As discussed earlier in the characteristics of the NoSQL, it emerges as a BASE system which use the portioning scheme. This system is cost efficient, flexible, scalable, and can be portioned into various places to store data. The data is not lost if the server break down due to technical issues or computer issues. This non-relational database system is less complicated as the complication in the SQL were high (Malhotra et al., 2017).

Even though, NoSQL are a breakthrough from the relational database, there are some disadvantages of NoSQL comparing to the SQL. In NoSQL the database is scalable but can only provide availability and partition tolerance by giving consistency property. When compared to the SQL database the NoSQL does not have a standard query language. It is difficult for new users to interact with NoSQL database because it does not have an ideal interface due to which the database is difficult to maintain.

(Venkatraman et al., 2016) However, NoSQL plays a vital role in the management of big data analytics. There are many advantages of NoSQL in the analytics of big data. This database has a property for adaptability meaning when a business wants to change its data model, the big data analytics is flexible and can adapt any conditions and situations. Another important factor of NoSQL in big data analytics is that it has a distributed scalable system where the user can perform the queries smoother and faster. The database is simpler to use for big data analytics and is very efficient for major data analytic functions i.e. it can perform both read and write schema functionality. So, NoSQL plays a vital role in the Big Data analytics therefore, many companies are shifting toward the NoSQL non- relational database because this database is suitable for the analytics of big data.

## *OPEN ISSUES IN BIG DATA*

*A. Platform and Framework*

*a. Congestion on Data Analytics System-* There will be congestion of big data on the analytics system because of the system where the data is analyzed or collected, working environment, input data such as structured, un-structured and semi-structured. There will be high increase of data for computation, which results in putting big load on the data analysis system. To bypass the congestion of the data analytics system, more resources for computation of data analysis system should be added and splitting the analysis of data into various computing nodes (Tsai, Lai, Chao and Vasilakos, 2015).

*b. Communication between systems-* (Tsai, Lai, Chao and Vasilakos, 2015) Communication between the system and big data analytics is one of the open issues on the basis of platform and framework perspective. Mostly the system designing for the big data analytics will be parallel computing. The users work on the cloud system or search engine-based system which result in the overall working operation of the Knowledge Discovery in Databases. The main issue in the communication between the system and data analytics is that the communication cost will increase. (Tsai et al., 2015) Also, the communication of big data analytics with other system is an issue because we do not know how they will communicate with one another. The communication between the different system and big data analytics should decisive.

*c. Security Issues-* Data are collected from various sources such as communication, transportation, health, education, entertainment and so on. These data are collected for data analytic process. One of the important factors to protect these big data analytics is security. The analytic system should secure the collected data in order to make the system reliable. The security of the system should be one of the important factors to be considered while collection of big data analytics. The security issues of big data analytics can be branched into four factors. The fours factors are input, analysis of data, output and the communication of big data analytics with other systems (Tsai et al., 2015). Input are those data which are gathered from various sources such as sensor input, IOT and so on. These inputs sources should not be volatile to attacks and the communication factor with other system should be considered for big data analytics. Hence, security issues are one of the important problems of big data analytics (Tsai, Lai, Chao and Vasilakos, 2015).

*B. Data Mining*

On the basis of data mining perspective there are some open issues such as the privacy issues, algorithm for MapReduce solution, congestion on data mining algorithm and so on. The privacy issues are one of the important issues on data mining because other people information should not be leaked while analyzation of the data. The company should take responsibility, liability in order for big data analytics and use of those data should be done properly. Since, the data mining is used in big data analytics, the personal and protective data of the people can be revealed to other authorities or organization after the analysis of the big data.

## *CONCLUSION*

In this essay we discussed about the perception on big data and the big data system architecture. We discussed about the data generation, data acquisition, data storage and analysis of the big data life cycle. The different type of analytics was presented and discussed above in the essay. This essay also presented about the relational database management system. There were several disadvantage factors of SQL compared to the non-relational database management system i.e. NoSQL. We discussed how NoSQL is better than the traditional relation database and different advantages of NoSQL over SQL. We presented why the non- relational database management system i.e. NoSQL is chosen for the big data analytics rather than the old relational database management system. There were several advantages of choosing NoSQL over SQL for the big data analytics which is discussed and presented in this essay. This is why NoSQL played a vital role for the big data analytics because of its advantages over the relational database management system.

This essay presented the different data model of NoSQL i.e. Key-value store, Document store, Column store and Graph store databases. We discussed about how these NoSQL model databases operates and the features, advantages and disadvantages. We discussed about several characteristics of NoSQL and the effect of CAP theorem in NoSQL. There were several challenges for the big data collection and analysis which is discussed and presented. We discussed about the several data analysis approaches among them are Hadoop Framework, MapReduce Framework, Hive and NoSQL. These data analysis approaches were discussed and its features, advantages and disadvantages for the big data analytics were presented. Finally, Big Data open issues were discussed on the basis of platform and framework and data mining. The issues were discussed and presented with solutions.

Therefore, NoSQL plays a vital role for the big data analytics which is discussed in the above essay. NoSQL has many features that are suitable for the big data analytic research. Its high performance, scalability, flexibility makes this database suitable for Big data analytics.

## *REFERENCES*

1. Han Hu, Yonggang Wen, Tat-Seng Chua and Xuelong Li, 2014. Toward Scalable Systems for Big Data Analytics: A Technology Tutorial. IEEE Access, 2, pp.652-687.
2. Venkatraman, S., Kaspi, K. and Venkatraman, R., 2016. SQL Versus NoSQL Movement with Big Data Analytics. *International Journal of Information Technology and Computer Science*, 8(12), pp.59-66
3. Moniruzzaman, A B M & Hossain, Syed. (2013). NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison. Int J Database Theor Appl. 6.
4. R. Cattell, ‘‘Scalable SQL and NoSQL datastores,’’SIGMOD Rec. vol.39, no. 4, pp. 12–27, 2011.
5. J. Dean and S. Ghemawat, ‘‘Mapreduce: Simplified data processing on large clusters,’’ Commun. ACM, vol. 51, no. 1, pp. 107–113, 2008.
6. A. Labrinidis and H. V. Jagadish, ‘‘Challenges and opportunities with big data,’’ Proc. VLDB Endowment, vol. 5, no. 12, pp. 2032–2033, Aug. 2012.
7. Han, J., Haihong, E., Le, G., & Du, J. (2011, October). Survey on nosql database. In Pervasive Computing and Applications (ICPCA), 2011 6th International Conference on (pp. 363-366). IEEE.
8. Malhotra, S., Doja, M., Alam, B. and Alam, M., 2017. Bigdata analysis and comparison of bigdata analytic approches. *2017 International Conference on Computing, Communication and Automation (ICCCA)*,.
9. L. Neumeyer, B. Robbins, A. Nair, and A. Kesari, ‘‘S4: Distributed stream computing platform,’’ in Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW), Dec. 2010, pp. 170–177.
10. John H. Howard, Michael L. Kazar, Sherri G. Menees, David A. Nichols, M. Stayanayanan, Robert N. Sidebotham, and Michael J. West. [Scale and Performance in a Distributed File System](http://pages.cs.wisc.edu/~cs736-1/papers/afs.pdf) ACM Trans. on Computer Systems 6(1), February 1988, pp. 51-81.
11. Herodotou, H., Lim, H., Luo, G., Borisov, N., Dong, L., Cetin, F. and Babu, S., 2011. Starfish: A Self-tuning System for Big Data Analytics. *CIDR*, pp.261-272.
12. T. Sandholm and K. Lai, ‘‘Dynamic proportional share scheduling in Hadoop,’’ in Job Scheduling Strategies for Parallel Processing. Berlin, Germany: Springer-Verlag, 2010, pp. 110–131.
13. Kambatla, K., Kollias, G., Kumar, V. and Grama, A., 2014. Trends in big data analytics. *Journal of Parallel and Distributed Computing*, 74(7), pp.2561-2573.
14. Tsai, C., Lai, C., Chao, H. and Vasilakos, A., 2015. Big data analytics: a survey. *Journal of Big Data*, 2(1).

# Appendix

# Learning Diary

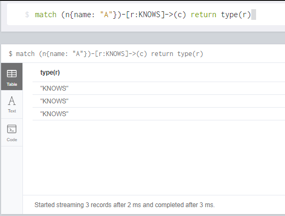
## WEEK 13 - Cypher Query Language

**CYPHER QUERY - 7**

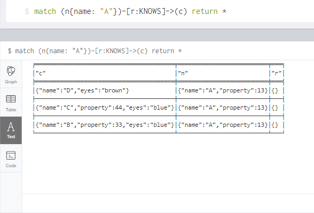
**RETURN**

****

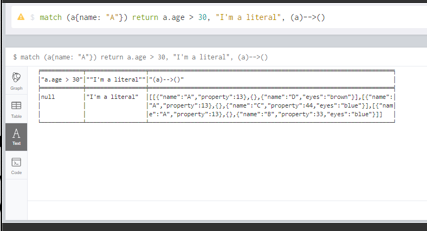
**RETURN RELATIONSHIPS**

****

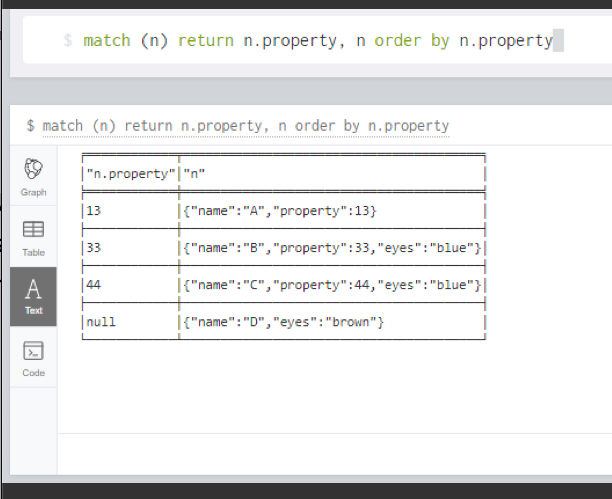
**RETURN ALL ELEMENTS**

****

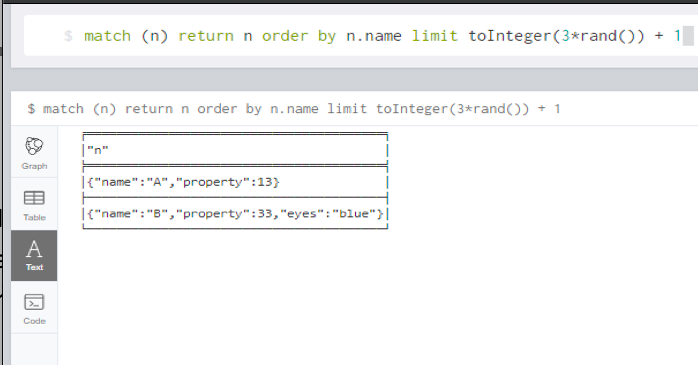
**ANY EXPRESSION CAN BE USED AS A RETURN ITEM**

****

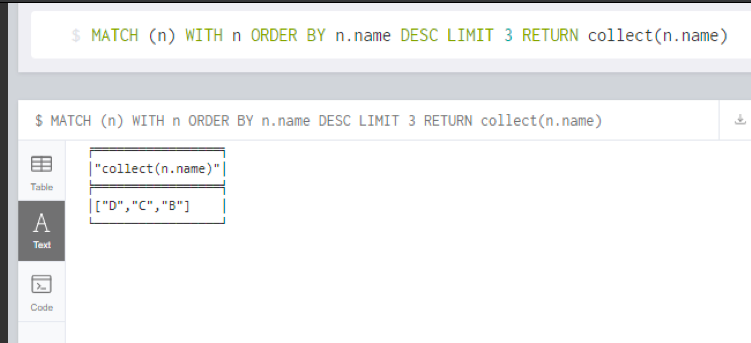
**ORDERING NULL**

****

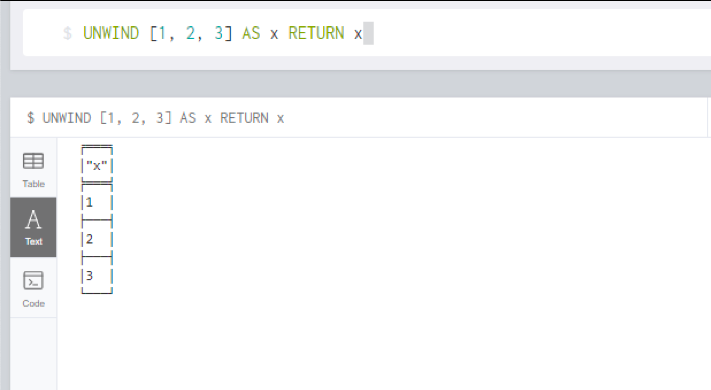
**LIMIT WITH EXPRESSION**

****

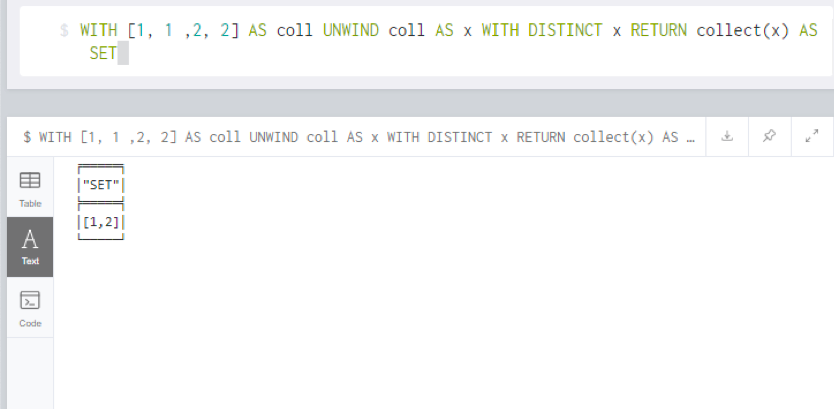
**LIMIT BRANCHING OF PATH SEARCH**

****

**UNWIND**

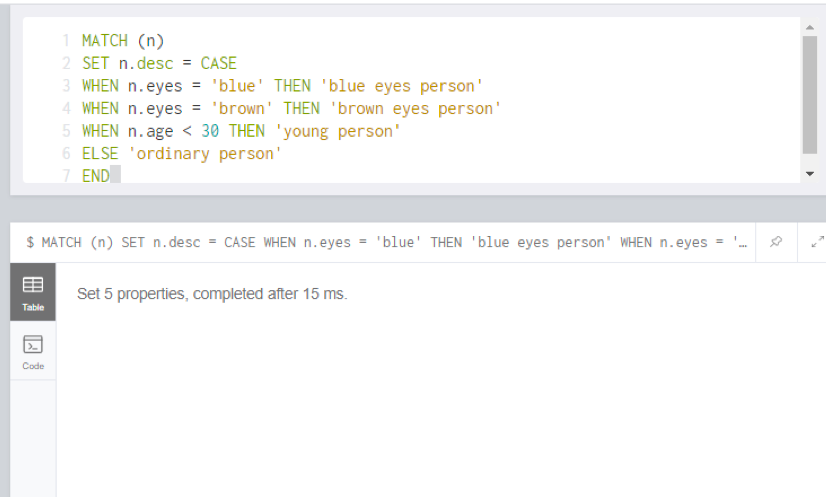
****

**CREATING DISTINCT LIST**

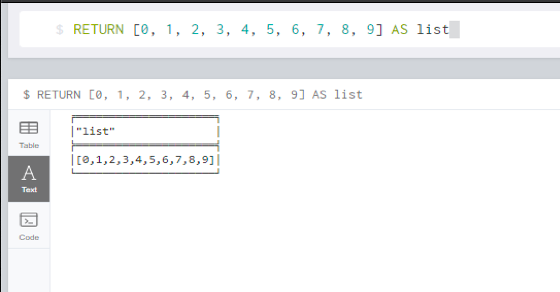
****

**CYPHER QUERY – 8**

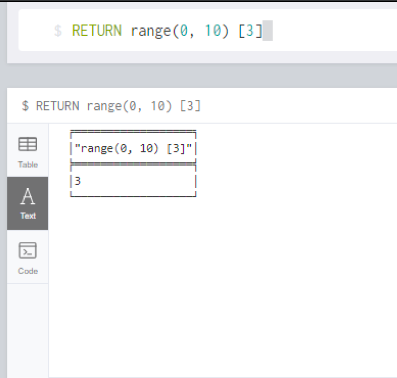
**CASE EXPRESSION**

****

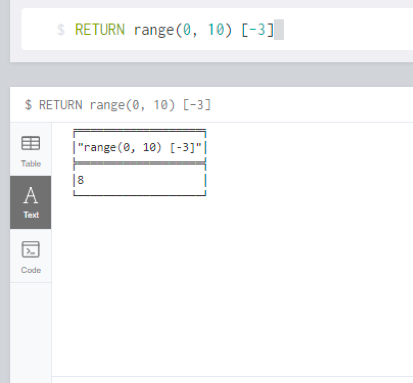
**LISTS**

****

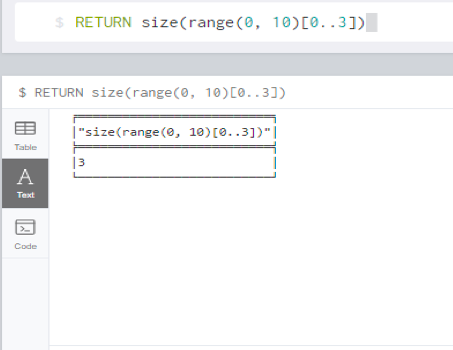
**RANGE OF A LIST**

****

**RANGE OF A LIST**

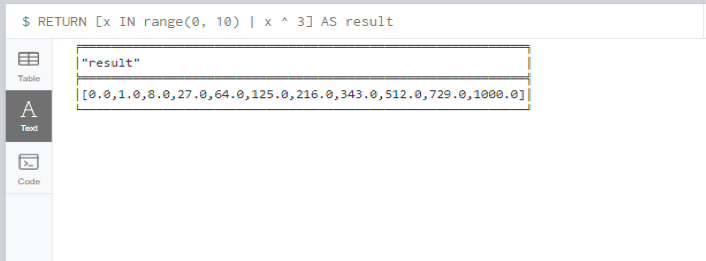
****

**OUT-OF-BOUND-LIST**

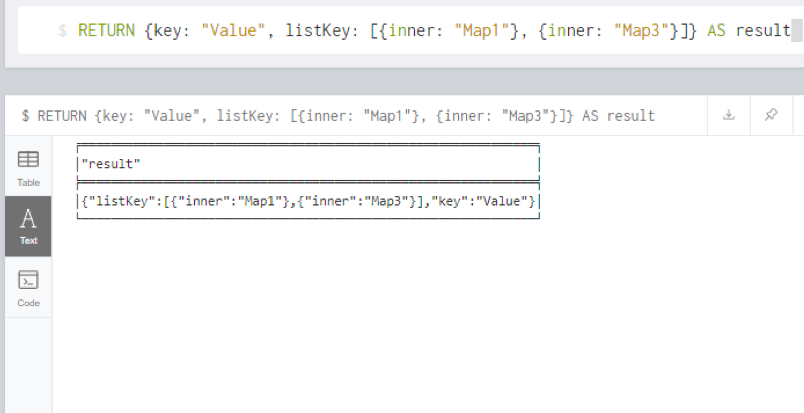
****

**LIST COMPREHENSION**

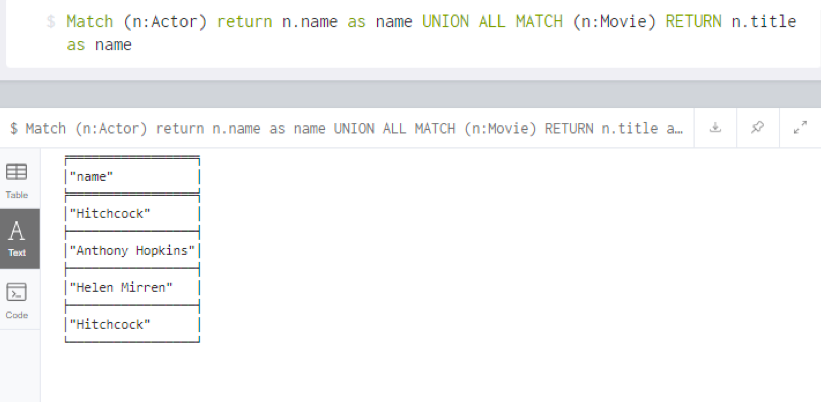
****

****

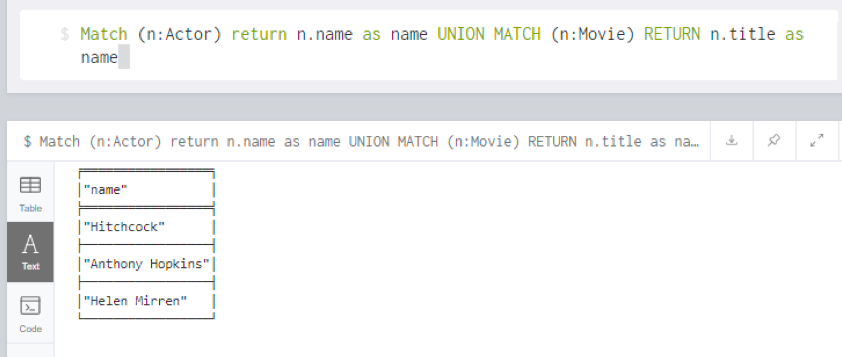
**LITERAL MAPS**

****

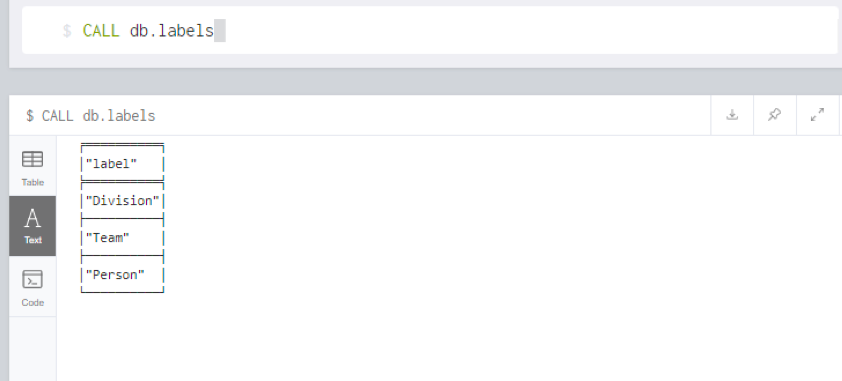
**COMBINATION OF TWO QUERIES**

****

**COMBINATION OF TWO QUERIES AND DUPLICATES REMOVEL**

****

**CALLING A PROCEDURE**

****

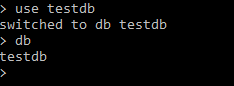
**Reflection**

In this cypher query language exercise, we learned about some general clauses and cypher syntax such as return, order by, ordering null, limit with and without expression, skip, with, applying filters on aggregate function results, unwind and so on. We approached this task by learning about how to use these clauses properly in queries. We use the unwind clause and saw that it could transform list backward into respective row. We observed and learned to use these general clauses, syntax which is logged in the learning diary. The learning process went smoothly, there were no problems and difficulties with this exercise.

The things that need to be improved is to use these clauses and syntax in more complex high-level queries and practice more to become familiar on how to use it effectively.

## WEEK 16 – Introduction to MongoDB

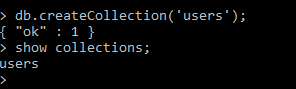
**a. Create a MongoDB database and switch to the database**

****

**b. Drop a Database**

****

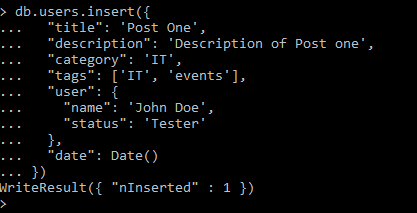
**c. Create Collection**

****

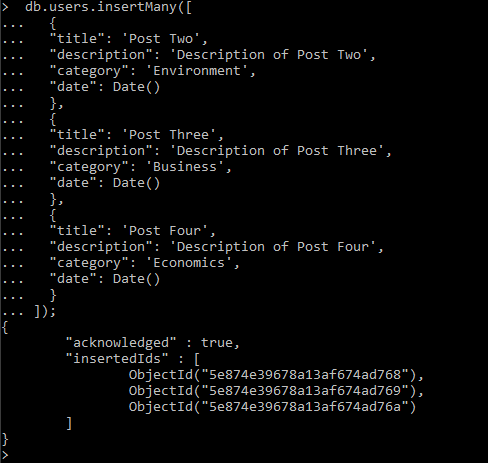
**d. Drop Collection**

****

**e. Inserting a single document**

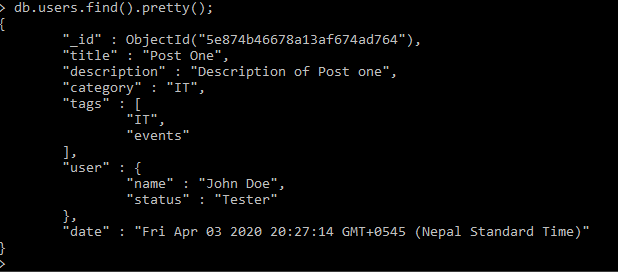
****

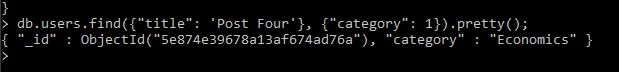
**f. Inserting many document**

****

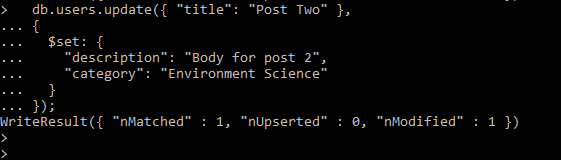
**g. Performing read, update, delete operations on document**

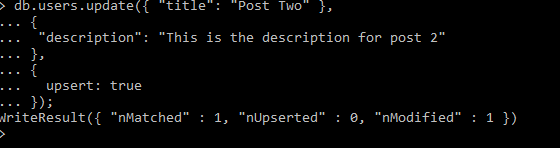
**Read Operation**

****

****

**Update Operation**

****

****

**Delete Operation**



**Reflection**

Before approaching to the tutorial of this week we learned about the mongo DB which is the leading solution of the non-relational database management system. It is a document-based database system that uses BSON documents and have JSON-like field and value pairs. This is a NoSQL database model that used document to store records in a collection. We learned that this model has high performance because it does not use the joins. This database model is scalable and is available due to replicated servers. This model can store all types of data and does not need any pre-defined schema.

After we learned about the mongo DB, we installed the database system in our personal computers and began to learn the functionalities of the system. After installing the system, we performed basic operations such as using the database, creating collection within the database, dropping database, dropping collection, inserting single document and many documents and performing different operations in the document such as read, update, delete and so on. There were no difficulties doing these tasks and was easy. It was like the SQL but the syntax was different. In order to get use to this database system, performing a lot of operations is needed to get use to the system functionalities.

## WEEK 17 – MongoDB Continuing

* **Write a MongoDB query to display all the documents in the collection restaurants.**

****

* **Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine for all the documents in the collection restaurant.**

****

* **Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine, but exclude the field \_id for all the documents in the collection restaurant.**

****

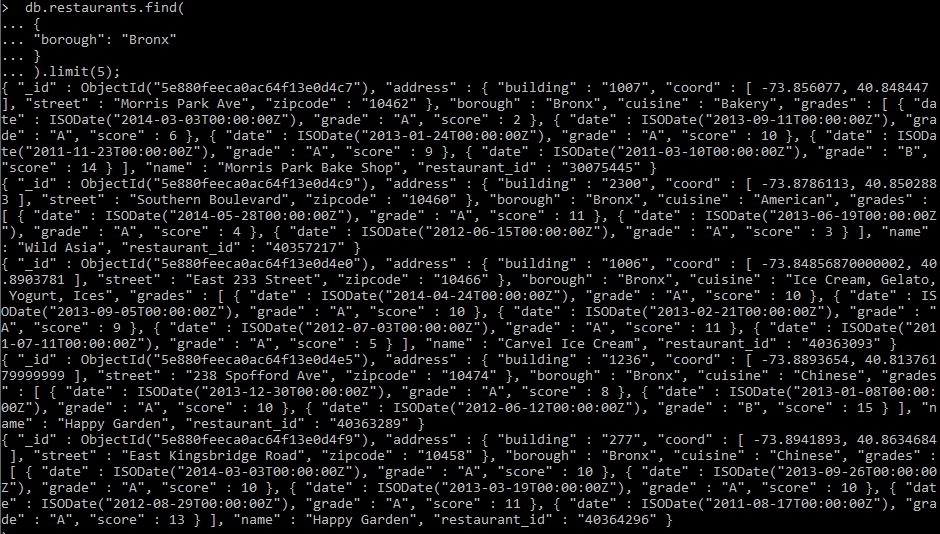
* **Write a MongoDB query to display the fields restaurant\_id, name, borough and zip code, but exclude the field \_id for all the documents in the collection restaurant.**

****

* **Write a MongoDB query to display all the restaurant which is in the borough Bronx**

****

* **Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx.**

****

* **Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.**

****

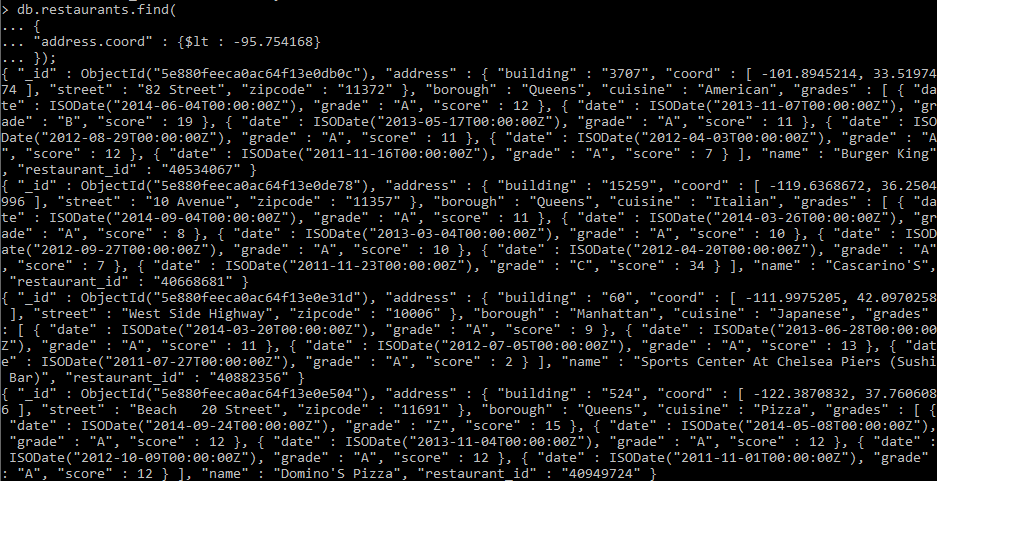
* **Write a MongoDB query to find the restaurants who achieved a score more than 90.**

****

* **Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100.**

****

* **Write a MongoDB query to find the restaurants which locate in latitude value less than -95.754168.**

****

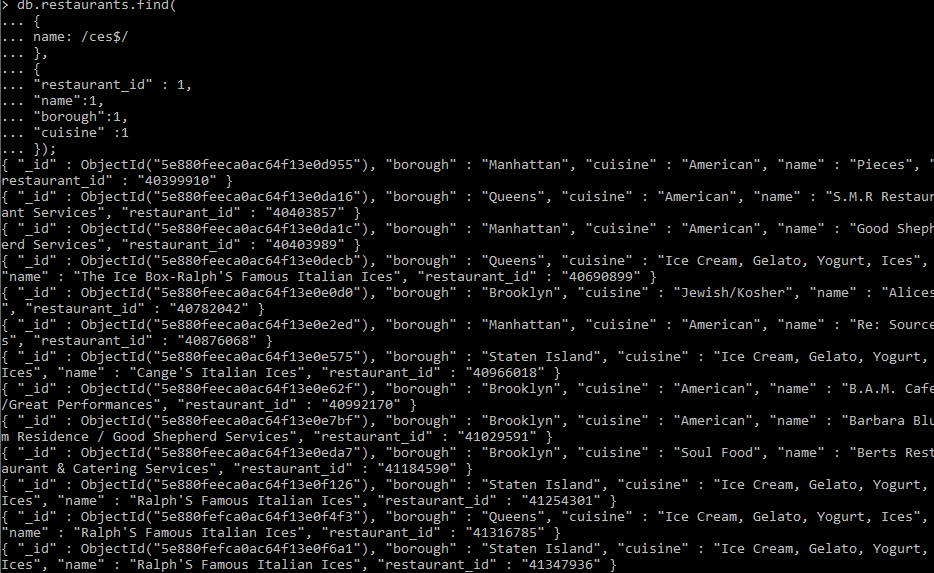
* **Write a MongoDB query to find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.**

****

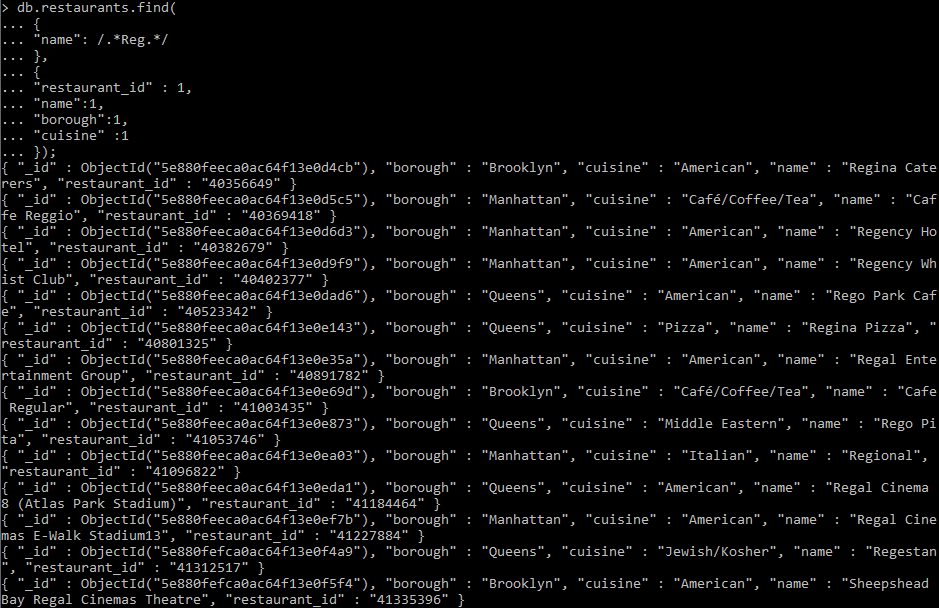
* **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Wil' as first three letters for its name.**

****

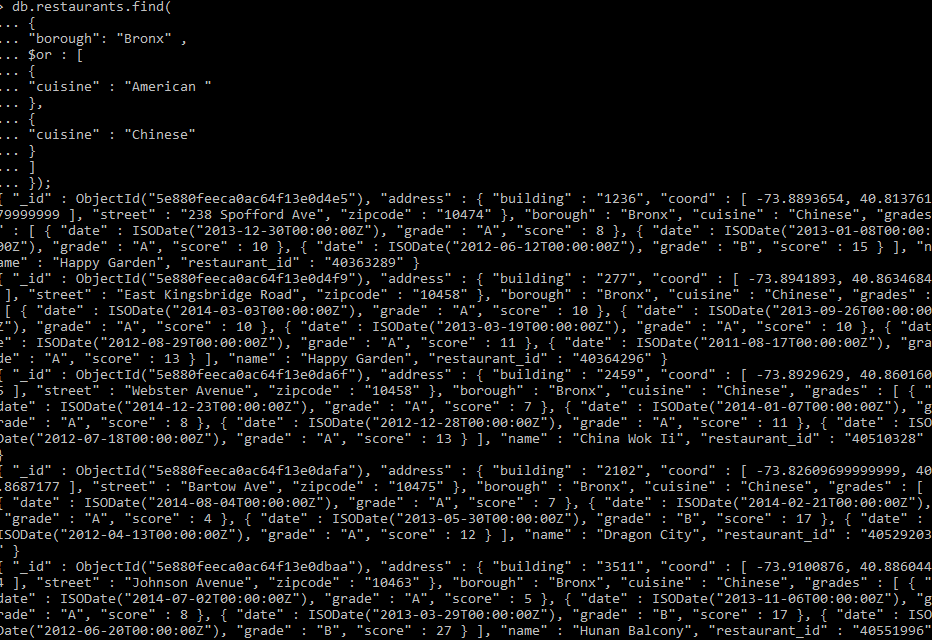
* **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.**

****

* **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name.**

****

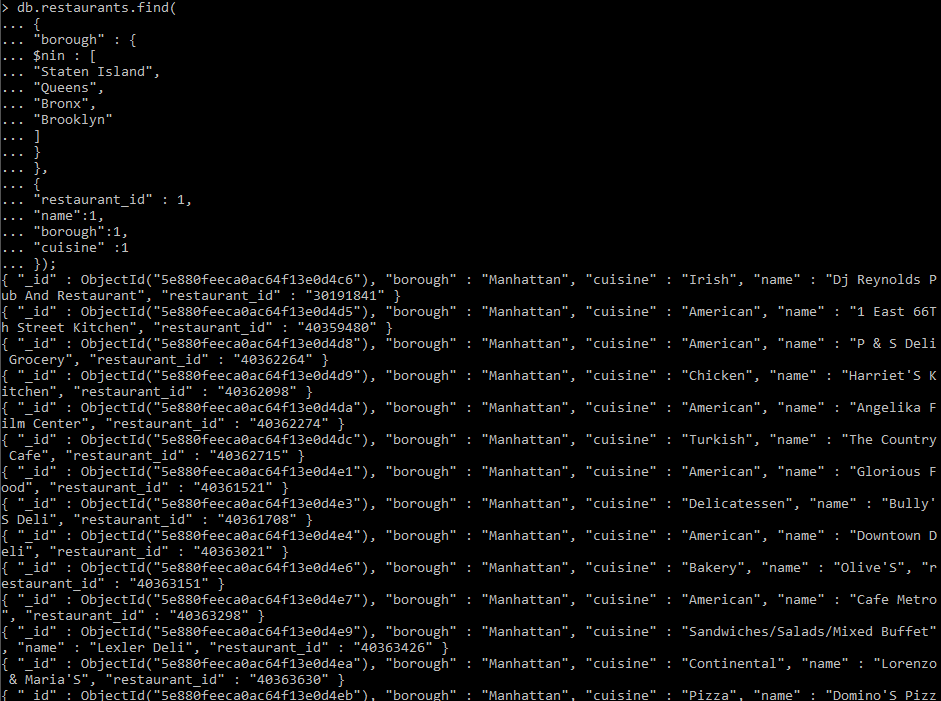
* **Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish.**

****

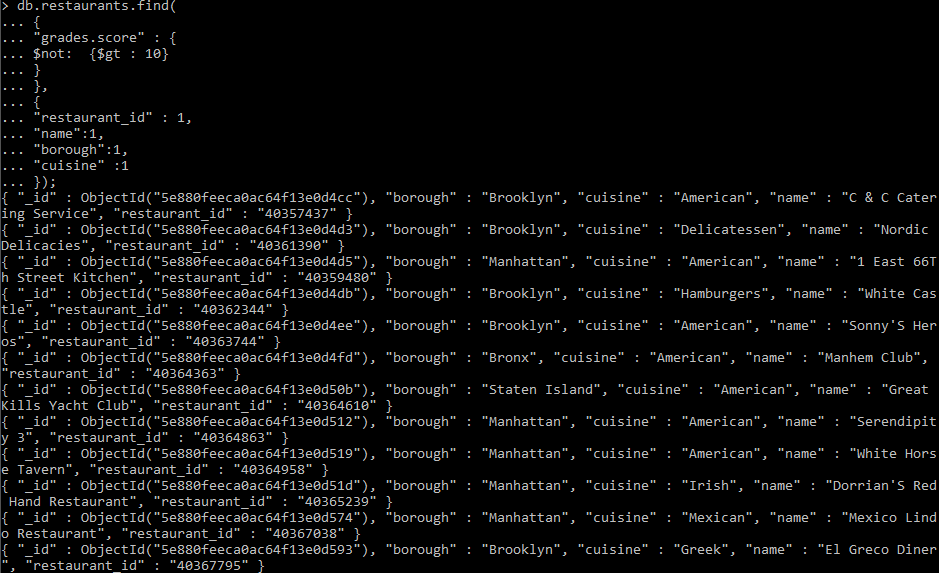
* **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronxor Brooklyn.**

****

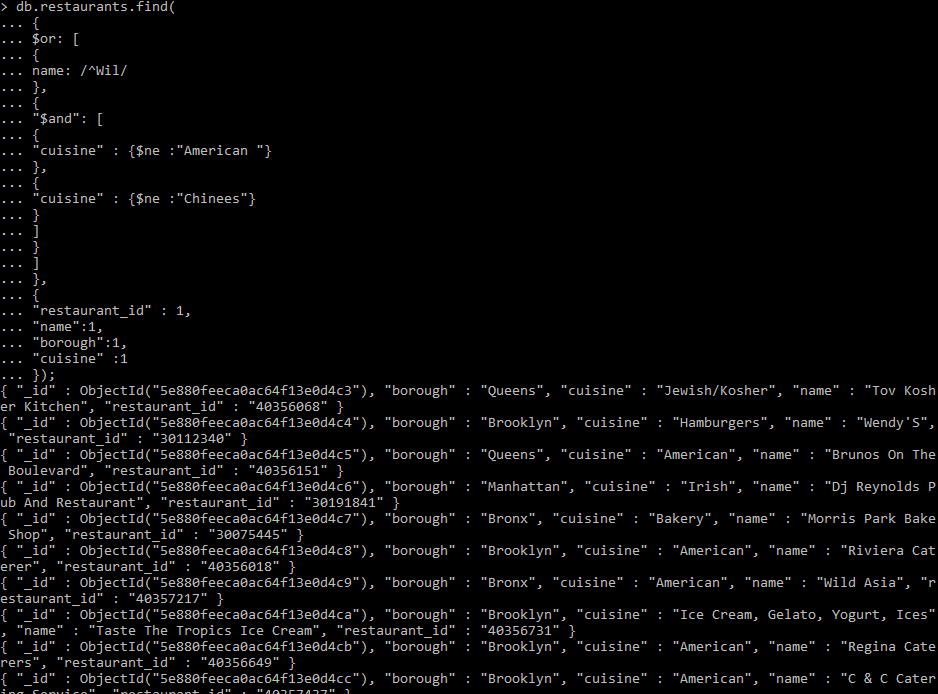
* **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.**

****

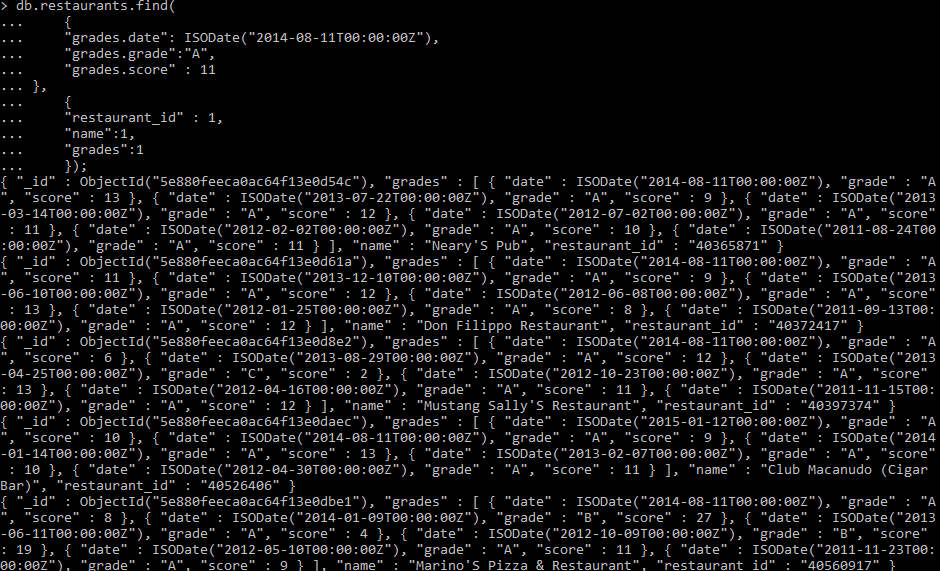
* **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score which is not more than 10.**

****

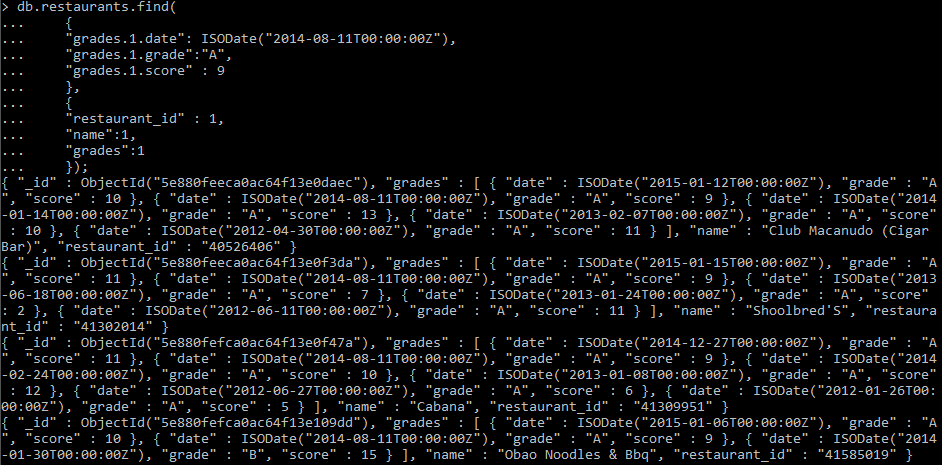
* **Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinese' or restaurant's name begins with letter 'Wil'.**

****

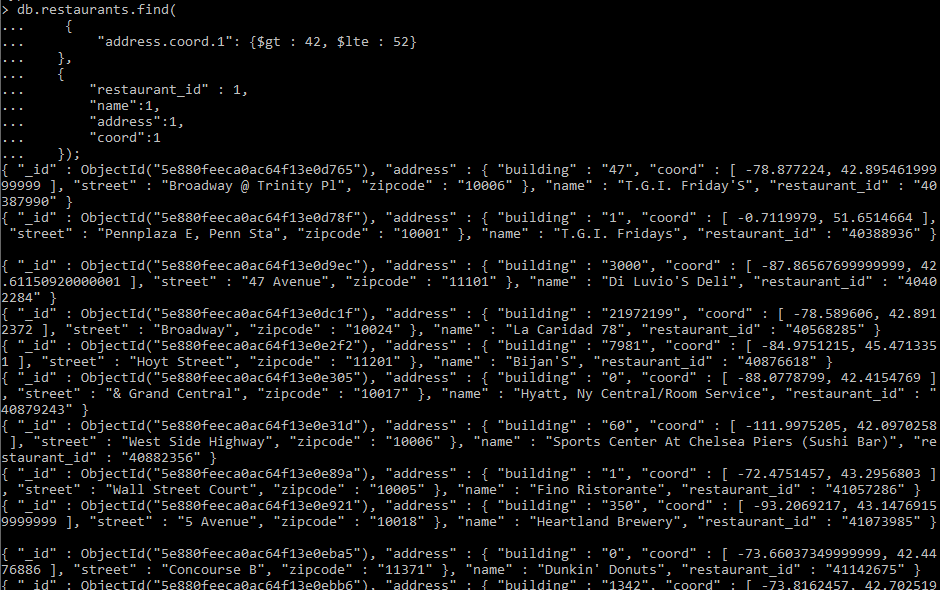
* **Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates.**

****

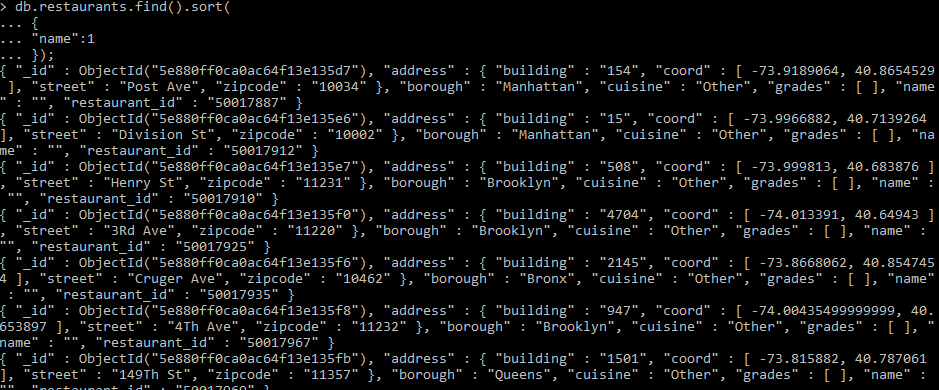
* **Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".**

****

* **Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and up to 52**

****

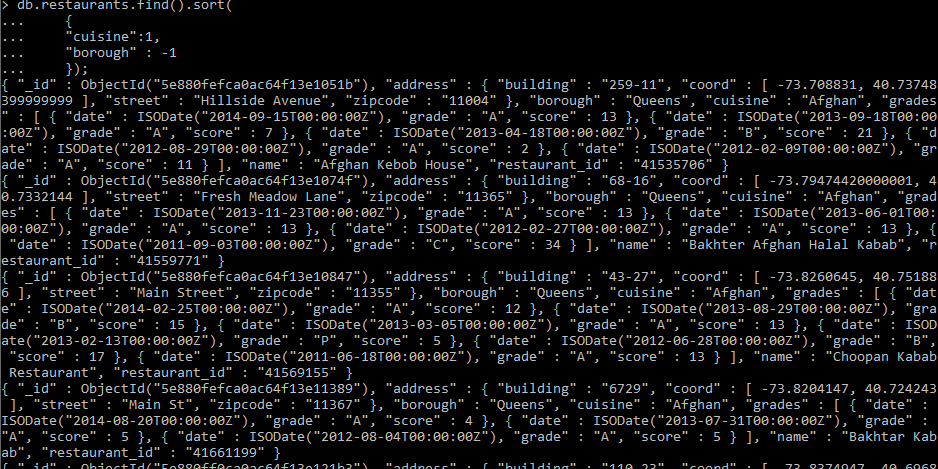
* **Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.**

****

* **Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.**

****

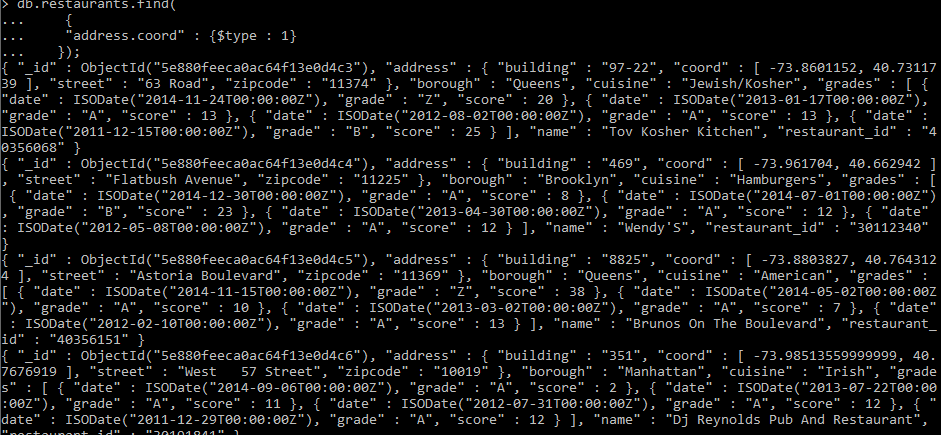
* **Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.**

****

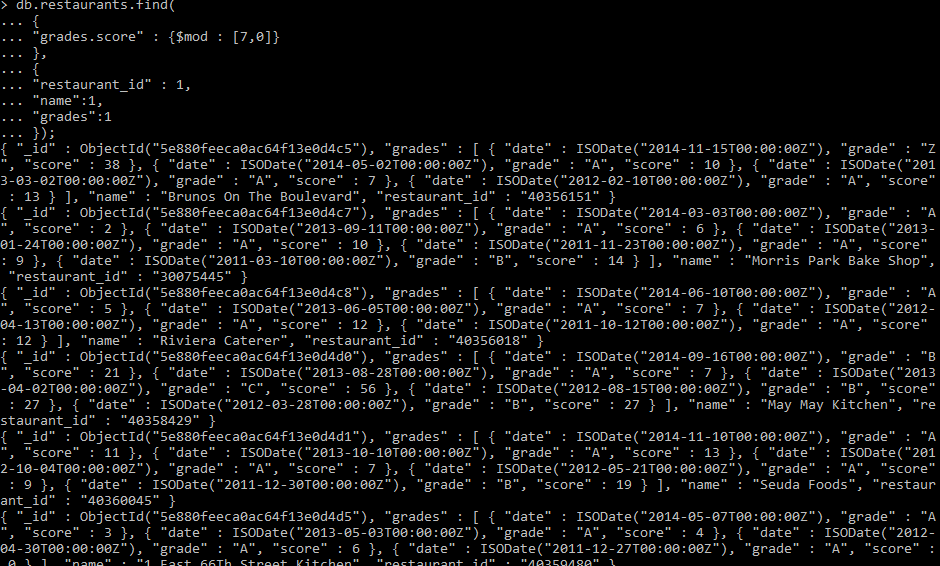
* **Write a MongoDB query to know whether all the addresses contains the street or not.**

****

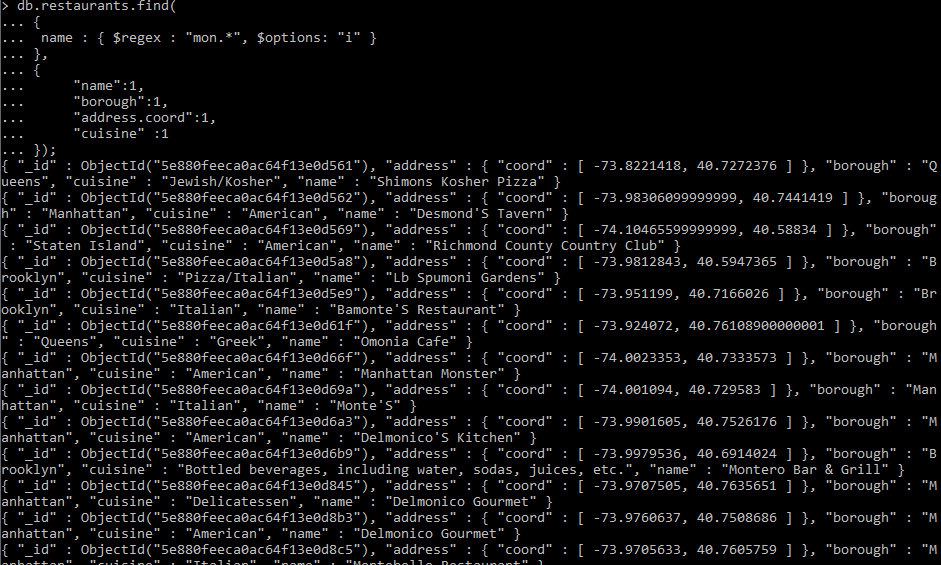
* **Write a MongoDB query which will select all documents in the restaurants collection where the coord field value is Double.**

****

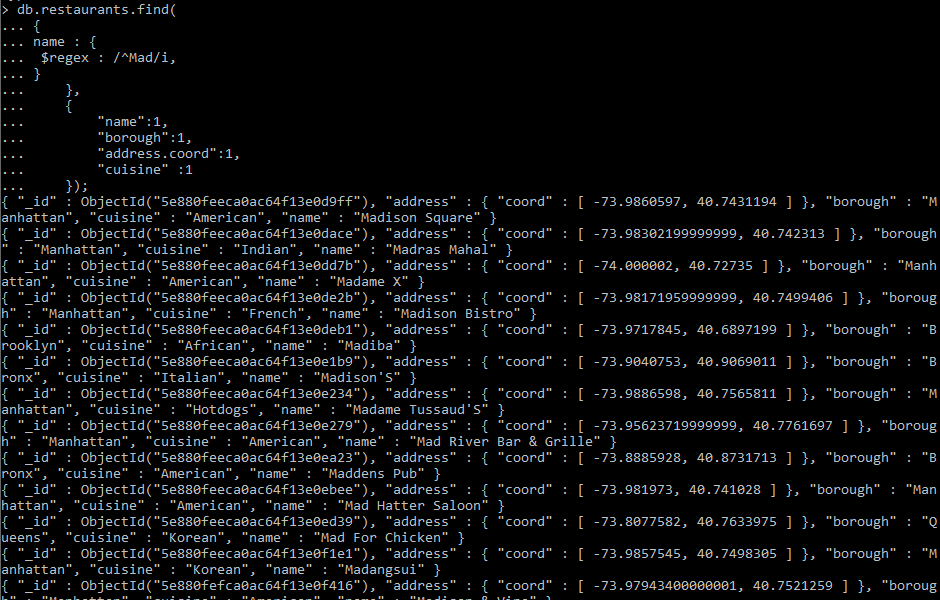
* **Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.**

****

* **Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.**

****

* **Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain 'Mad' as first three letters of its name.**

****

**Reflection**

Before approaching to this task, we have to learn about how to insert a JSON document into the database system. The process was really easy and after inserting the document in the system, we started doing queries in the document. The queries logic was as similar as we use to do in the SQL but the syntax was different but after we get use to the syntax the querying of the documents became easy. In mongo DB all the queries address a single collection. We used to find, limit, sort, update and so on. We used different operators such as increase, decrease, equals to, not equals to, multiply, minimum, maximum, and other operators to find the optimal solution to the queries. We used and, or operator and use like operators which id presented in the above log diary.

Once we get used to the syntax, it was all about logic and how we use that logic for querying the document and finding optimal solution. Personally, for me It was challenging to use the aggregate functions to query but once I got used using the aggregate functions more, it became easy operating on the aggregate function.

I still need to focus on using the functions, operators properly in order to gain knowledge and understanding when to use what while doing queries.

## WEEK 21 – Database Transaction Management

1. **List some database transaction properties, using an appropriate example to illustrate what they mean and why they are important.**

A transaction is a process by which groups of coherent changes takes place as a single change. The database is consistent when all the related data items are updated simultaneously. After the transaction is committed the changes should be visible and permanent. ACID property is the transaction properties. ACID property is: Atomicity, Consistency, Isolation and Durability.

1. **Atomicity:** While performing the transactions, it should take place at once or it does not take place at all.

For example: While withdrawing money from the ATM machine, an error occurs during the transaction than the system should abort the withdrawing operation process and restart or the operation should be cancelled.

1. **Consistency:** When moving from one state to another state during an operation, the previous operations should be completed in order to shift to another operation.For example: When withdrawing money from the ATM, the operation is not completed up and until the withdrawer record are added to the database.
2. **Isolation:** This transaction property says that a sequence of transaction or a set of concurrent transaction must have the same overall effect. This transaction property never allows for conflicting of two or more transactions simultaneously.

For example: When two transaction occurs on a database at the same interval of time than the first transaction will is not halted by the second transaction.

1. **Durability:** When a transaction is performed and stored in a database than the transaction should not be lost if the system crashes or fails i.e. the system should remain durable when an action is performed on the database.

For example: When a user completes a transaction via ATM, the system comes to an execute state and all the operation happened in the transaction will be recorded and stored long-lasting in the database management system.

1. **The following is a serial schedule, which involves actions in several transactions.**

*W*3(*z*) → *R*4(*z*) → *W*4(*z*) → *R*1(*x*) → *R*2(*y*) → *W*3(*y*) → *R*1(*z*) → *W*5(*x*) → *R*5(*y*) → *R*4(*y*) → *W*4(*u*) → *R*5(*v*)

1. **Identify the conflict pairs in the above schedule.**

The following are the conflicting pairs in the above schedule:

*<W*3(*z*), *R*4(*z*)>,

*<W*3(*z*), *W*4(*z*)>,

*<W*3(*z*), *R*1(*z*)>,

*<W*4(*z*), *R*1(*z*)>,

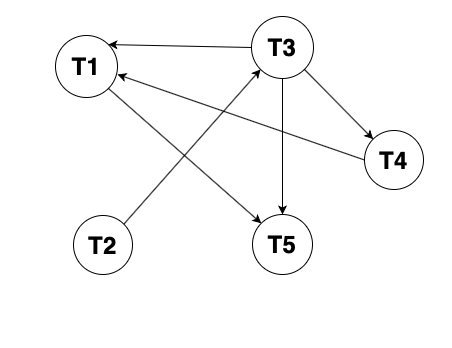
*<R*1(*x*), *W*5(*x*)>,

*<R*2(*y*), *W*3(*y*)>,

*<W*3(*y*), *R*5(*y*)>,

*<W*3(*y*), *R*4(*y*)>

1. **Draw precedence graph.**

****

**Reflection**

We learned about the database transaction management and its properties i.e. ACID property. We learned about different transaction states such as active transaction, partially committed transaction, committed transaction, failed transaction and aborted transaction. The transaction manager helps us to move from one consistent state to another consistent state during a transaction while the scheduler is used to provide strategy for execution of transaction and the recovery manager is used to recover the database to the consistent state after the system failure. We learned about scheduling of transactions that includes serial and non-serial schedule. We studied serializability and conflict in serializability.

# Queries

## 1. Show all the EPL teams involved in the season.

db.assignment2.distinct("HomeTeam");

## 2. How many matches were played on Mondays?

db.assignment2.find().forEach(function(day) {

day.Date = new Date(day.Date);

db.match.save(day);

});

db.assignment2.aggregate(

[

{

$project:

{

Date: "$Date",

day: { $dayOfWeek: "$Date" },

}

},

{

$match: {day: 2}

},

]

);

## 3. Display the total number of goals “Liverpool” had scored and conceded in the season.

db.assignment2.aggregate(

{ $match: { $or: [ { "HomeTeam": "Liverpool" }, { "AwayTeam": "Liverpool" } ] } },

{ "$project":

{

Club: "Liverpool", \_id: 0, "HomeTeam": 1, "AwayTeam": 1,

Total\_Goal\_Scored: { $cond: { if: { $eq: [ '$HomeTeam', 'Liverpool' ] }, then: '$FTHG', else: '$FTAG' } },

Total\_Goal\_Conceded: { $cond: { if: { $eq: [ '$AwayTeam', 'Liverpool' ] }, then: '$FTHG', else: '$FTAG' } }

}

},

{ $group:

{

\_id: "$Club",

"Total-Number-of-Goal-Scored": { $sum: '$Total\_Goal\_Scored' },

"Total-Number-of-Goal-Conceded": { $sum: '$Total\_Goal\_Conceded'}

}

});

## 4. Who refereed the most matches?

db.assignment2.aggregate(

{ $group: { "\_id" : "$Referee", "Total-Matches-Refereed-2018/19" : { $sum: 1} } },

{ $sort: { "Total-Matches-Refereed-2018/19" : -1 } },

{ $limit: 1}

);

## 5. Display all the matches that “Man United” lost.

db.assignment2.find({

$or: [

{ "AwayTeam": "Man United", "FTR": "H"},

{ "HomeTeam": "Man United", "FTR": "A" }

],

});

# REFERENCES

* Docs.mongodb.com. 2020. *Comparison Query Operators — Mongodb Manual*. [online] Available at: <https://docs.mongodb.com/manual/reference/operator/query-comparison/> [Accessed 19 April 2020].
* Docs.mongodb.com. 2020. *Aggregation Pipeline — Mongodb Manual*. [online] Available at: <https://docs.mongodb.com/manual/core/aggregation-pipeline/> [Accessed 19 April 2020].
* Docs.mongodb.com. 2020. *Logical Query Operators — Mongodb Manual*. [online] Available at: <https://docs.mongodb.com/manual/reference/operator/query-logical/> [Accessed 19 April 2020].
* Docs.mongodb.com. 2020. *$Cond (Aggregation) — Mongodb Manual*. [online] Available at: <https://docs.mongodb.com/manual/reference/operator/aggregation/cond/> [Accessed 19 April 2020].
* Docs.mongodb.com. 2020. *$Dayofweek (Aggregation) — Mongodb Manual*. [online] Available at: <https://docs.mongodb.com/manual/reference/operator/aggregation/dayOfWeek/#definition> [Accessed 20 April 2020].

.