Bollywood Movies Analysis Using Neo4j

Bollywood Movies Analysis Using Neo4j

Abstract

Graph databases are different from regular databases in that they focus more on relationships between different pieces of information. This means that the database can remember the relationships between different pieces of data, which makes it much faster when you want to look up information about a particular subject. There are many different graph databases out there, but Neo4j is the leader. It's easy to use and scales well, and its Cypher language is simple to understand. The Graph Data Science Library also has a lot of powerful graph manipulation and machine learning capabilities. Our team is going to analyze Bollywood movies using Neo4j to figure out which relationships exist between different pieces of information. This will be a difficult task because of the large amount of data involved, but Neo4j is a good choice because it can handle complex relationships easily. The data has been inserted into the Neo4j database and subsequently, relationships between Actors, Movies, and Director nodes have been analyzed.

Keywords- Neo4j, Graph Databases

1. Introduction

Graph databases are specially designed to store and manage relationships between different pieces of data. This makes them extremely powerful for storing and exploring data, as relationships are the focus of these databases. There is a way of thinking about things that are very flexible. You can use it to think about things in different ways or to solve problems. A new type of database called "NoSQL" is gaining popularity, and it's different from traditional databases like SQL. NoSQL databases mostly fall into two categories, "Documental" and "Key-Value." Documental databases are designed to store data in a hierarchical format, while Key-Value databases use a simple, flat index to store data. Graph databases are a new type of NoSQL database that is growing in popularity. They are different from other databases in that they can be used to store data in graph form. This can be used for things like tracking customer relationships or tracking the flow of information. Overall, it seems to be a powerful and versatile database that is gaining in popularity.

A graph database stores data in nodes (like people, places, things, etc.) and the relationships between them (like parents and children, actions, and results, etc.). Each node has information like its name, type (like person, place, thing, etc.), and direction (like north, east, south, etc.). Graph databases also allow for a lot of associations between nodes, like parent-child relationships, actions, ownership, etc. A database can be used to explore a graph, which is a collection of nodes and edges between them. This is useful for things like social networking or finding recommendations. An engine helps you find patterns in data, and fraud detection uses these patterns to help you know if something is wrong.

Graph databases are different from other databases in that they focus on relationships between items. This means that relationships between items (such as those in a table in an RDBMS) are not required to be calculated during execution. This can cause problems in cases where relationships need to be computed, as this can take longer than usual. When you use an RDBMS, the computer gets slower as the size of the data increases. Graph databases, on the other hand, use connections or pathways between data points, which means they are more efficient even for large data sets. Graph databases can help prevent fraud by detecting patterns in relationships between people and things. This can help you do things like buying things online in near-real time because you can use quick graph queries to find out who is likely to be fraudulent. If you have an email address and you use the same one on different websites or on different devices, that might mean there is a connection between you and that person. For example, if many people have the same IP address but live in different places, that might be a sign that they are related.

Neo4j is a database that is built around simple but effective optimization. All the data that Neo4j is connected to is contained in each record or node. Relationships are what we term these connections. The node itself contains all the data necessary to identify the next node in the sequence. A connected graph serves as the native storage layer. Native means that Neo4j doesn't have to calculate relationships between your data at query time because of this approach. Data engineers and scientists who try to adjust product requirements in relational database systems often run into trouble because most of the operations in relational databases are based on set operations. As the size of data grows, these operations become slower and more difficult. Unexpected delays or memory usage can occur when executing queries on a database. This is because the data is stored in disconnected aggregates, which makes it difficult to represent relationships between the data. This can lead to expensive join operations being required, which can be time-consuming. Neo4j is a much faster way to search for relationships between data than other databases. For example, if you wanted to find all the orders that were made by someone who also works at a company, Neo4j would be much faster than a database that just stores direct pointers between records. Instead, Neo4j would have to look through an index to find all the links between the records, which would be much faster than looking through direct pointers.

Cypher is a type of language that allows you to query the data in Neo4j. It is like SQL, which is a common language used to query data from databases. Because Cypher is so easy to learn, it is the most easily accessible graph language. Cypher is a special kind of software that lets you see relationships between items in a data set by drawing a graph. Cypher is a language used to create queries for working with Neo4j data. These queries can be used for any CRUD operation on the graph, such as creating, reading, updating, or deleting nodes. Cypher's syntax is made up of round brackets and arrow symbols, which are used to indicate relationships between nodes. Cypher uses an ASCII-art type of syntax.

2. Related Work and Background

According to previous literature, there is a trend of people wanting a more flexible system of schemas when it comes to big data applications. This is because when many tables are joined together, the efficiency of a relational database decreases. Graph databases, such as Neo4j, help to overcome this problem. There are differences in the way queries are structured and the data models used when talking about Neo4j and relational databases.

Neo4j is an open-source program with a fast graphics engine at its core, recovery capabilities, two-phase submission, support for XA transactions, and other database product characteristics, which is presently the industry standard. It is an embedded, disk-based, fully transactional Java persistence engine that stores data structured in networks rather than tables. It is a network-oriented database. Neo4j's use of the so-called "network-oriented database" is what makes it intriguing. As opposed to the relational model's tables, rows, and columns, this paradigm expresses domain data as a "node space," which is a network of nodes, relationships, and attributes (key-value pairs). The context in which nodes interact is shown via the properties that can be annotated on relationships, which are first-class objects. Natural hierarchically arranged problem areas are well suited for the network model.

Graph databases are becoming more popular than relational databases. They're used in a wide range of fields, including recommendation engines, the semantic web, and social networking. RDBMSs and graph databases, like Neo4J, have different features, making them better suited for certain tasks. Graph databases let you store information in a way that lets you easily see how it relates to other information. This can be useful for creating applications that don't need to use a traditional database system’s limitation.

Recent studies have shown that the set of pre-defined queries is executed faster than relational databases, and that graph databases are more flexible when it comes to restructuring schemas. This is useful when implementing commercial systems in different domains.

Throughout the COVID outbreak, we've seen doctors prescribing antibiotics to patients indiscriminately. This could lead to antibiotic resistance and possibly superbugs. We need to be careful when prescribing antibiotics, considering the patient's allergies and the infection itself.

Doctors can see more information about an antibiotic by connecting data about patients, diagnoses, dosages, and durations into a Neo4j graph. This knowledge graph can help clinicians provide the best care for their patients while minimizing antibiotic resistance and aggravating the medical environment issue.

In the early years, before graph databases were invented people had to work hard to get their businesses off the ground. Many used computer systems that were specifically designed for data analysis and developed a product that couldn't sell, and this is how graph databases came into existence. They were used to finding a way to make the product successful. Having technology that was 1000 times faster on certain operations, it had huge potential for customer value. This is what we call a "graph database," and it's good at solving many different problems.

3. Background

Bollywood is the Indian film industry that produces movies in the Hindi language. It is the world's largest film industry, with over 1000 movies produced each year. It is a billion-dollar industry and an important contributor to India's economy.

Bollywood is a movie industry from India that is very popular and has a lot of cultural significance. It reflects the culture of India and is heavily influenced by Indian society. Throughout its history, Bollywood films have often reflected the changing morals and values of Indian culture. Many of the storylines in Bollywood films are about traditional Indian customs and practices. Bollywood films are also often released on important festivals, such as Diwali, Holi, Ramadan, or Eid. Indian society is also heavily influenced by Bollywood films, songs, and stars. For example, people in India often look to Bollywood for fashion inspiration. If an actress wears a new style of dress in a Bollywood movie, it becomes a popular trend the following year among brides in India. Bollywood music also has a big impact on Indian culture, as songs from Bollywood movies are often played at weddings, parties, and festivals. Bollywood has also helped India’s economy by creating jobs and shaping India’s international image.

The way movies have changed over the years is a result of the way things are in India today. Movies used to be entertainment for the lower classes, but now they have a more important purpose of helping people learn and be aware of important issues in India. Back in the day, movies were made mainly for Hindus and men, but now they have evolved to include more people and reflect the concerns of the modern, educated Indian. Movies used to be just for fun, but now they have a more important role in educating people.

Bollywood movies are growing in popularity all over the world, as they showcase the unique aspects of Indian culture in a way that is interesting and entertaining to people from all over the world. For example, Western influences are seen in some of the films, while others focus on the love for India and the country’s history.

The word "Bollywood" is a portmanteau of the words "Bombay" and "Hollywood." This means that the Bollywood film industry is based in the city of Bombay. These movies cover a wide range of topics, and some of them even touch on sensitive political and religious subjects. Many of the movies contain cheerful dance scenes which help to lighten the mood. Bollywood movies are very popular all over the world, and one of the reasons for this is that Dangal, a movie from India, was such a huge success in China. Hollywood has been extensively written about, but Bollywood seldom receives the same level of attention. This project will focus on the film industry to learn more about it. The tabular data is compiled and contains 1,698 Bollywood movies.

4. Proposed Work

It is important to choose the right technology for your project before starting. This can be a difficult process, involving a lot of study and consideration of the solution and its technical capabilities. We are using Neo4j, Cypher SQL, Python, and Machine Learning for our project.

Our team will be conducting an analysis of Bollywood movies and understand them in a better way using Neo4j. Neo4j will be used to analyze the associations among the key objects in the Bollywood movie data which include but are not limited to directors, actors, writers, etc. The Neo4j database is a convenient tool to use because of its ability to deal with complex and multi-connection data. With the development of the Internet, data has grown immensely large to be handled by a single relational database with connections among different tables using foreign keys, and it's hard to query complex relationships among different entities. The volume and complexity of Bollywood movie data represent a big challenge in terms of data. Also, other NoSQL databases like MongoDB and Cassandra do not handle relationships very easily and features like foreign keys must be used explicitly to implement them. Hence, Neo4j is an appropriate choice for this project.

Below are the main components that are required as prerequisites to move ahead with the project.

(i) Neo4j

The Neo4j Graph Data Platform provides tools and applications to help you easily access and understand data. It is based on the world's leading graph database, which makes it a powerful tool for analysis and navigation.

The Neo4j Graph Data Science Library is a great place to search for information about graphs. Bloom is a way to ask questions about graphs and can operate across Neo4j. There are many products that use open-source toolkits. For example, Cypher on Apache Spark (CApS) and Cypher for Gremlin. These toolkits make it easy to build graph-based solutions. Additionally, the community around these toolkits is incredibly large, so you can always find help if you need it.

Neo4j is a database that is very scalable and able to handle lots of data quickly. It is also ACID compliant, which means that it is reliable and efficient. Neo4j is used by businesses to understand how data is related and to make better decisions based on that information. Customers are important to businesses. Business culture is important to customers.

It is a database that is designed around relationships between data. Instead of having fixed relationships between tables, Neo4j stores data based on the connections between data points. This makes Neo4j a type of "NoSQL" database, which are databases that are different from traditional relational databases. Neo4j is open source and is backed by Neo Technology. In a graph database, data points (called nodes) are connected to each other in any way that makes sense, and the relationships between the nodes and the direction of the relationships also have properties that describe them. This is different from traditional databases, where tables of data all have the same form, have fixed relationships to other tables, and the relationships between the tables themselves do not have any properties.

If you have a lot of data, or it's in a structure that makes it difficult to search, you can use graph search to find what you're looking for. Some big uses for it are things like finding people or things in a large database. Social networks are a great way to store information about people and the relationships between them. Queries that would be difficult or impossible to do with a traditional database, like finding out the second-degree friends of a person, are easy in a social network. Cypher is a simple language that makes these queries easy to understand. With a graph database, you can use your connected data to make recommendations to your customers. Neo4j offers the power and speed to keep up with the growth of your data.

Geo-tracking is like trying to visit each city on a map exactly once, but with the shortest possible route. This is done with a data model that is a map of cities and the routes between them. This is useful for things like routing inventory, scheduling tours, and geospatial analysis. Neo4j is a better choice for doing this than a relational database because it is faster and doesn't require extra lookups to indices during retrieval.

This software is reliable and can be used by billions of people. It also has operational reliability, meaning it can be backed up and used in a clustered setting. Neo4j supports a variety of programming languages and has a built-in API that allows any program to access the data it stores.

The dual license model means that you can use Neo4j for both open-source projects and commercial projects. This means that there is a big community of support available, as well as extra production support and scale for large enterprises.

Graph databases are great for modeling data that is highly associative, like relationships between people or products. Neo4j keeps your data safe by making sure it is always accurate and consistent. With a lot of built-in support, it's easy to integrate with other applications and systems.

(ii) Cypher SQL

Cypher is like a kind of language that you can use to ask questions about graphs. It's declarative language, which means that you write it out exactly as you want it to be interpreted, rather than relying on code to do the job for you. Function (clause, keyword, expression) means that something can do something. Predicate (clause, keyword, expression) means that something is a characteristic of a thing.

Cypher is a computer language that is used to store and process data. This allows you to easily see and understand the relationships between different pieces of data. We are using round parentheses to show the circles around the node entities for example (m:movies) .

Cypher is a program that helps you understand how things work in the world. It has a lot of features that make it easy to work with lists of data, and it also has some special abilities that make it easier to get results than some other programs. Cypher is a tool that lets you change the structure of a graph and the data it contains. You can also use Cypher to import large amounts of data into your graph. With user-defined procedures, you can add features to the language that are not currently available.

(iii) Python

Python is a programming language that is easy to learn and use. It can be interpreted or run on a computer. It is also object-oriented and dynamic, which makes it a good choice for Rapid Application Development and for connecting different pieces of software. Additionally, Python's simple syntax makes it easy to read and maintain programs.

Python is popular due to its fast edit-test-debug cycle. There is no compilation step, so the cycle is very fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source-level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and more. The debugger is written in Python itself, testifying to Python's introspective power.

Python was created in the late 1980s and early 1990s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is a language that is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and Unix shell scripts.

Python is copyrighted, but you can use it freely so long as you follow the rules of the GNU General Public License. Python is now a part of the institute's core development team, and while Guido van Rossum still plays a major role in its direction, others also contribute.

(iv) Machine Learning

Machine learning is a kind of technology that helps machines learn and understand things on their own. This technology is used in a lot of different places, like chatbots and predictive text services, the shows that Netflix recommends to you, and how your social media feeds are displayed. It's also used in self-driving cars and to help machines diagnose medical conditions. Artificial intelligence (AI) is a type of technology that allows machines to do things that would normally be done by people. Some companies use machine learning, which is a type of AI that lets computers learn on their own.

Machine learning is a type of technology that can be used to make things easier or faster for businesses. It's becoming more common, so it's important to be knowledgeable about it if you work in business. Some tasks, like recognizing pictures of people, are easy for humans, but it's hard for machines to learn how to do them on their own. Machine learning is a way of helping machines learn from experience, which can make the task easier for them. Machine learning starts with information like numbers, photos, or text. This information is used to train a machine-learning model. The more data is used, the better the model.

5. Data Collection

We will use the IMDB and Bollywood Movies datasets for our project. These datasets can be easily obtained from Kaggle. Both datasets have information about Bollywood movies from the 1990s to 2021. We are also using a Bollywood movie dataset compiled by P. Premkumar includes roughly 1700 Bollywood films from a two-year period for our reference. Apart from using the Neo4j database for our project, we will also use Cypher SQL, a declarative, textual query language, like SQL used for graphs. It consists of expressions and keywords for execution purposes, some of which we are already aware of like WHERE, ORDER BY, SKIP, AND, etc. It’s the query language for graphs. We can get data from the graph with Neo4j's Cypher. Because it is like SQL for graphs and was inspired by SQL, we can directly focus on collecting the desired data from the graph. It is by far the simplest graph language to learn due to its similarity to other languages and its intuitiveness. Unlike SQL, Cypher is all about expressing graph patterns. There is a special clause MATCH for matching those patterns in your data. One would usually draw these patterns on a whiteboard, just converted into text using ASCII-art symbols.

6. Exploratory Data Analysis

Our team will use Neo4j to study Bollywood movies and understand them better. Neo4j can help us analyze the relationships among different objects in the Bollywood movie data, such as directors, actors, writers, etc. This is a helpful tool because it can handle complex and multi-connection data. With the growth of the Internet, data has become much larger and more complex to store in a single database. This makes it difficult to query relationships among different entities. The volume and complexity of Bollywood movie data is a big challenge.

The python script was created to process the data. The script is presented as a Python notebook. We first preprocessed the data, run the python notebook and the file was created and stored as CSV files which were then imported into Neo4j Desktop. There are three CSV files- actor_movie.csv, director_movie.csv, and movie.csv respectively. We used these files interchangeably throughout the implementation of this project.

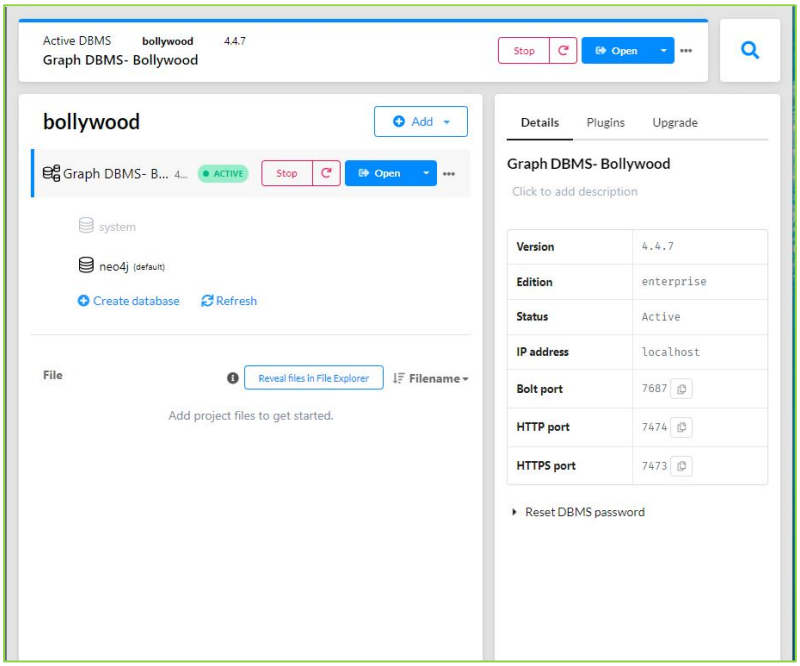
7. Data Insertion

The data was inserted into Neo4j Desktop using Cypher queries. Three different nodes were created, and labeled as "director", "movie", and "actor". These nodes are interconnected with each other. We use different queries to find if there are any connections. Two different kinds of relationships were established, which were "act_in" and "direct". The “act_in” relationship is between the actor and movie nodes and the ‘direct” relationship is between the director and movie nodes respectively.

8. Results

(i) Overview

On Neo4j Desktop, create a project called "bollywood" and put the three CSV files into its "import" folder. After importing the data, open the Neo4j Browser and enter the following commands. The figure below shows what Neo4j Desktop Application looks like once the data is imported with the required plugins.



After importing the data, open the Neo4j Browser and enter the following commands. This command imports the data into the database.

LOAD CSV WITH HEADERS FROM 'file:///movie.csv'

AS row MERGE (m:movie {name: row.name, release_period: row.release_period, remake: row.remake, franchise: row.franchise, genre:row.genre, screens: toInteger(row.screens), revenue: toInteger(row.revenue), budget:toInteger(row.budget)});

CREATE CONSTRAINT ON (m:movie) ASSERT m.name IS UNIQUE;

LOAD CSV WITH HEADERS FROM 'file:///director_movie.csv'

AS row MERGE (d:director {name: row.director}) MERGE (m:movie {name: row.movie}) MERGE (d)-[r:direct {new_director: row.new_director}]->(m);

LOAD CSV WITH HEADERS FROM 'file:///actor_movie.csv'

AS row MERGE (a:actor {name: row.actor}) MERGE (m:movie {name: row.movie}) MERGE (a)-[r:act_in {new_actor: row.new_actor}]->(m);

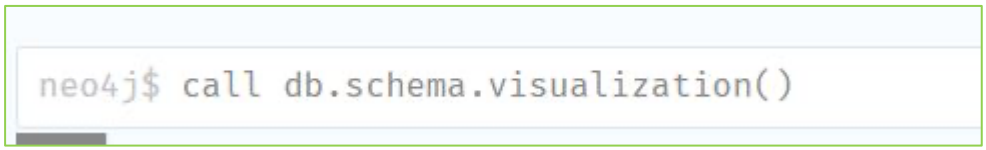
CREATE CONSTRAINT ON (a:actor) ASSERT a.name IS UNIQUE;

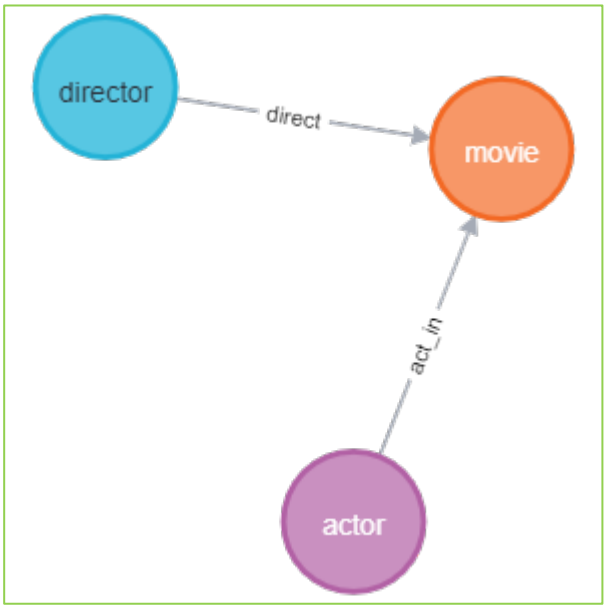
CREATE CONSTRAINT ON (d:director) ASSERT d.name IS UNIQUE;

```
LOAD CSV WITH HEADERS FROM 'file:///movie.csv'
AS row MERGE (m:movie {name: row.name, release_period: row.release_period, remake: row.remake, franchise: row.franchise, genre:row.genre, screens: toInteger(row.screens), revenue: toInteger(row.revenue), budget:toInteger(row.budget)});

CREATE CONSTRAINT ON (m:movie) ASSERT m.name IS UNIQUE;

LOAD CSV WITH HEADERS FROM 'file:///director_movie.csv'
AS row MERGE (d:director {name: row.director}) MERGE (m:movie {name: row.movie}) MERGE (d)-[r:direct {new_director: row.new_director}]->(m);

LOAD CSV WITH HEADERS FROM 'file:///actor_movie.csv'
AS row MERGE (a:actor {name: row.actor}) MERGE (m:movie {name: row.movie}) MERGE (a)-[r:act_in {new_actor: row.new_actor}]->(m);

CREATE CONSTRAINT ON (a:actor) ASSERT a.name IS UNIQUE;
CREATE CONSTRAINT ON (d:director) ASSERT d.name IS UNIQUE;
```

(ii) We can see how the three types of nodes (actors, directors, and movies) are connected by looking at the connections between them. The figure given below illustrates this.

call db.schema.visualization()





(iii) Open Neo4j Bloom on Neo4j Desktop. Set the query limit to 3600 and execute match all with the following query.

```
MATCH p=(a:actor) -->(m:movie) <--(d:director) RETURN p;
```

Going back to the main window when we run it, Bloom generates a topological view of the dataset. Figure 1. shows the original version of the data with all the nodes and Figure 2. shows the zoomed-out version for proper viewing of connections.

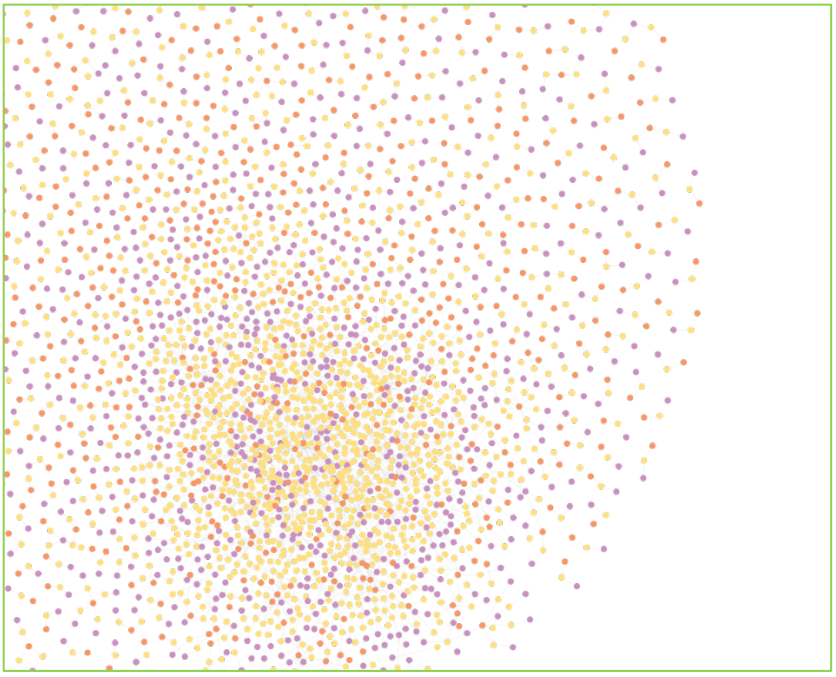
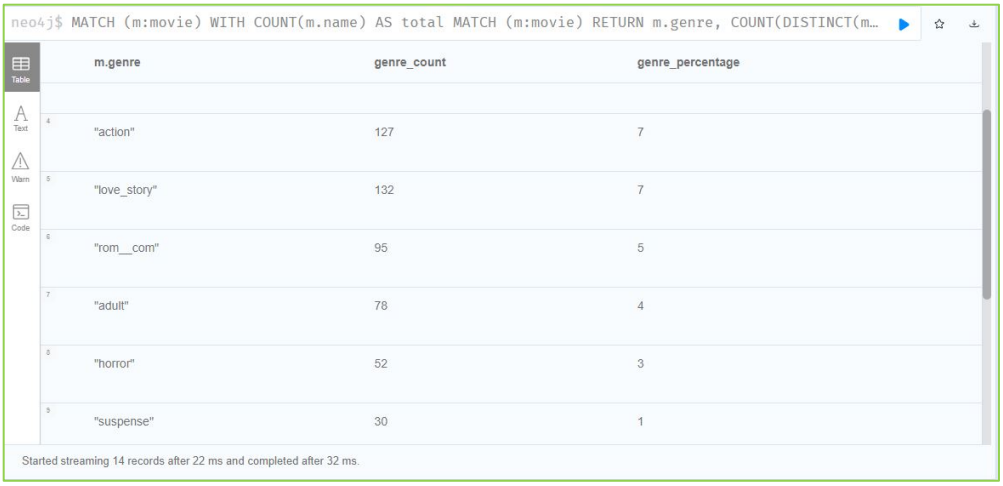


Figure 1.(Normal graph view)

Figure 2. shows us a lot of different actor-director pairs in the Bollywood movie world. But a big group of them is clustered together.

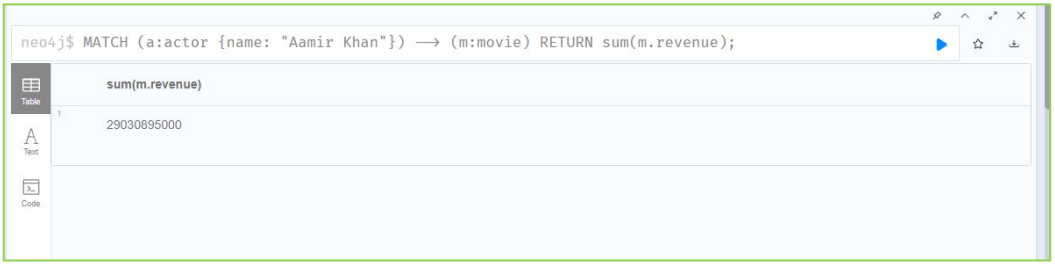
(iv) We execute the following query to get the total no of movies with different genre distributions.

```
MATCH (m:movie)
WITH COUNT(m.name) AS total
MATCH (m:movie)
RETURN m.genre, COUNT(DISTINCT(m.name)) as genre_count,
100*COUNT(DISTINCT(m.name))/total as genre_percentage
ORDER BY genre_percentage DESC
```



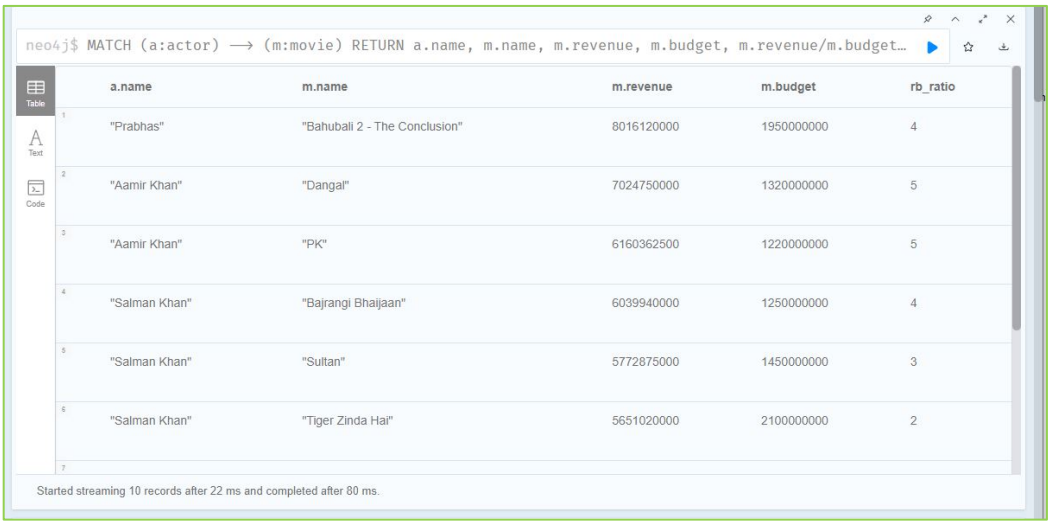
(v) The Cypher matched all films with Aamir Khan as the lead actor and then calculated their total revenue.

```
MATCH (a:actor {name: "Aamir Khan"}) --> (m:movie)
RETURN sum(m.revenue);
```



(vi) The figure below shows the query execution for top earners. This query removed the constraint on the actor nodes. It calculated the ratios of revenue to budget to show how profitable the films were. Finally, the query sorted the results based on how much money the films made.

```
MATCH (a:actor) --> (m:movie)
RETURN a.name, m.name, m.revenue, m.budget, m.revenue/m.budget as rb_ratio
ORDER BY m.revenue DESC LIMIT 10;
```



(vii) This query tells us about the biggest box office flops of time.

```
MATCH (m:movie)
RETURN m.name, m.revenue, m.budget, m.revenue - m.budget AS profit
ORDER BY profit LIMIT 10;
```

neo4j$ MATCH (m:movie) RETURN m.name, m.revenue, m.budget, m.revenue - m.budget AS profit ORDER BY …

Table | Text | Code

	m.name	m.revenue	m.budget	profit
1	"Bombay Velvet"	431365000	1180000000	-748635000
2	"Broken Horses"	10130000	600000000	-589870000
3	"Mirzya"	134665000	630000000	-495335000
4	"Jagga Jasoos"	868572500	1310000000	-441427500
5	"Zanjeer"	221475000	600000000	-378525000
6	"Mohenjo Daro"	1040750000	1380000000	-339250000
7				

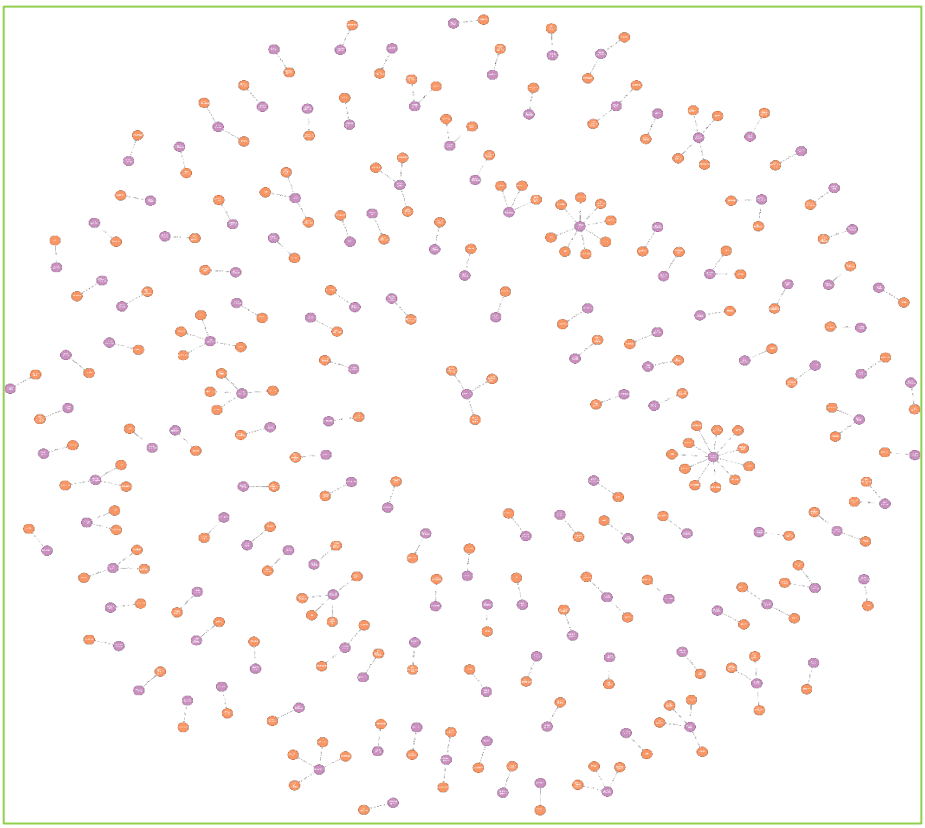
Started streaming 10 records after 18 ms and completed after 29 ms.

(viii) This query shows some of the actors who have starred in a lot of thrillers or horror movies

```
MATCH (a:actor) --> (m:movie {genre: "thriller"})
RETURN a.name, COUNT(DISTINCT(m.name)) AS Thriller
ORDER BY Thriller DESC LIMIT 10.
```

Its corresponding graph is shown by the following query.

```
MATCH path= (a:actor) --> (m:movie {genre: "thriller"})
WHERE m.genre = "thriller"
RETURN path;
```

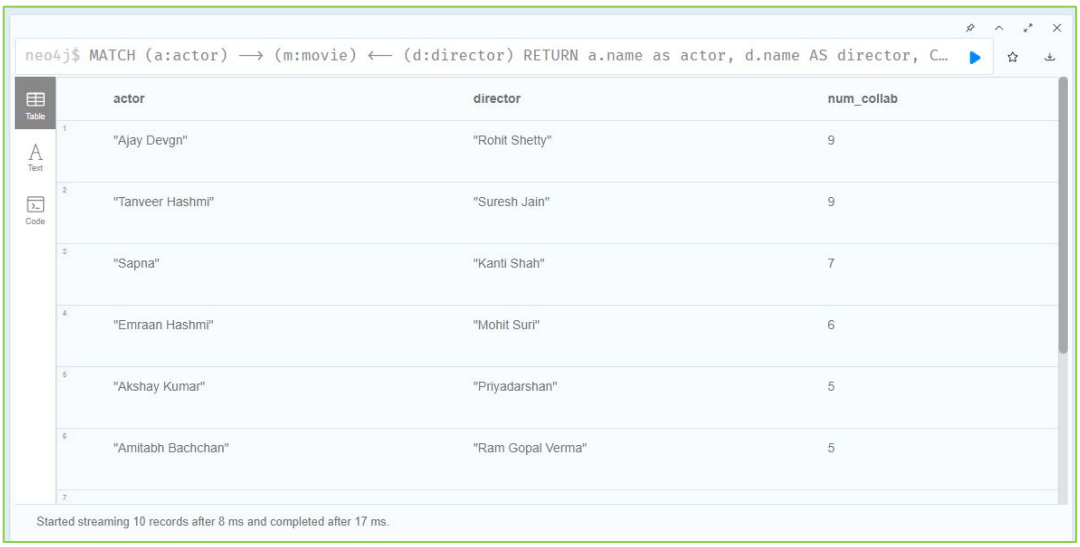


(ix) This query is the successor of the previous one. Actor Emraan Hashmi has been successful in the thriller genre, and his movies are all listed in the dataset.

```
MATCH p=(a:actor {name: "Emraan Hashmi"}) -->(m:movie)
RETURN m.name, m.genre
ORDER BY m.genre DESC;
```

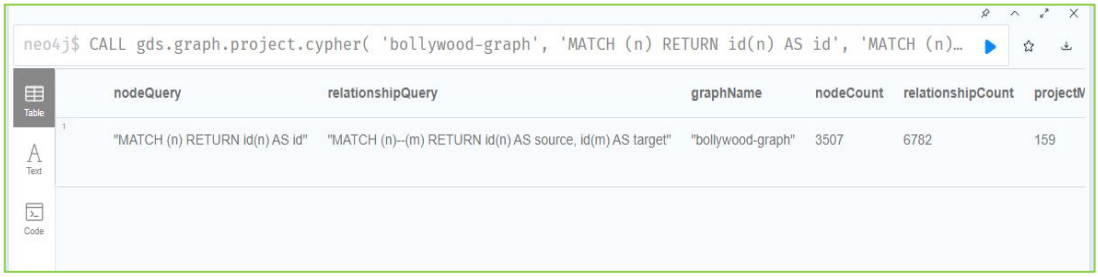
(x) Actor-Director work together

```
MATCH (a:actor) --> (m:movie) <-- (d:director)
RETURN a.name as actor, d.name AS director, COUNT(DISTINCT(m.name)) AS num_collab
ORDER BY num_collab DESC LIMIT 10;
```



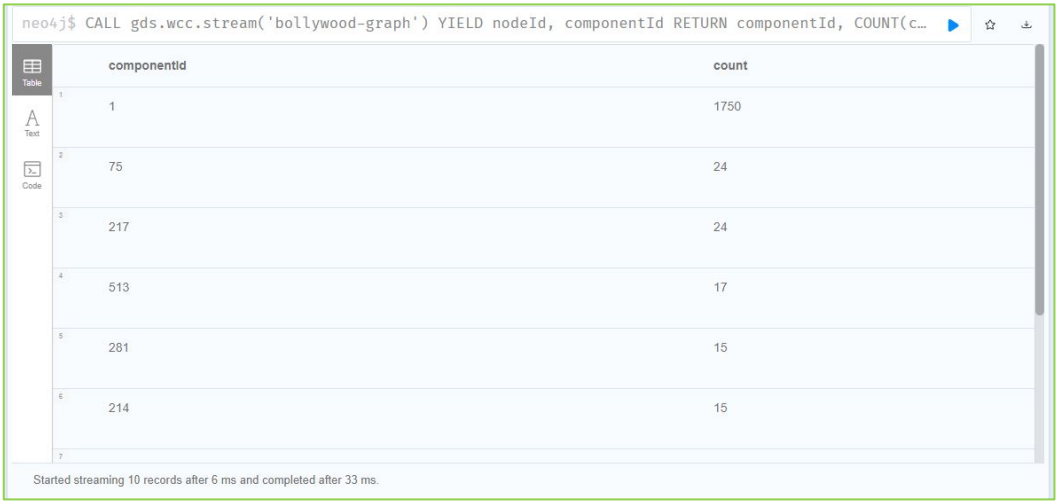
(xi) Enabling GDS Plugin for the project

```
CALL gds.graph.project.cypher(
   'bollywood-graph',
   'MATCH (n) RETURN id(n) AS id',
   'MATCH (n)--(m) RETURN id(n) AS source, id(m) AS target');
```



(xii) Running WCC and it returns the top 10 largest communities.

```
CALL gds.wcc.stream('bollywood-graph')
YIELD nodeId, componentId
RETURN  componentId, COUNT(componentId) as count
ORDER BY count DESC LIMIT 10;
```

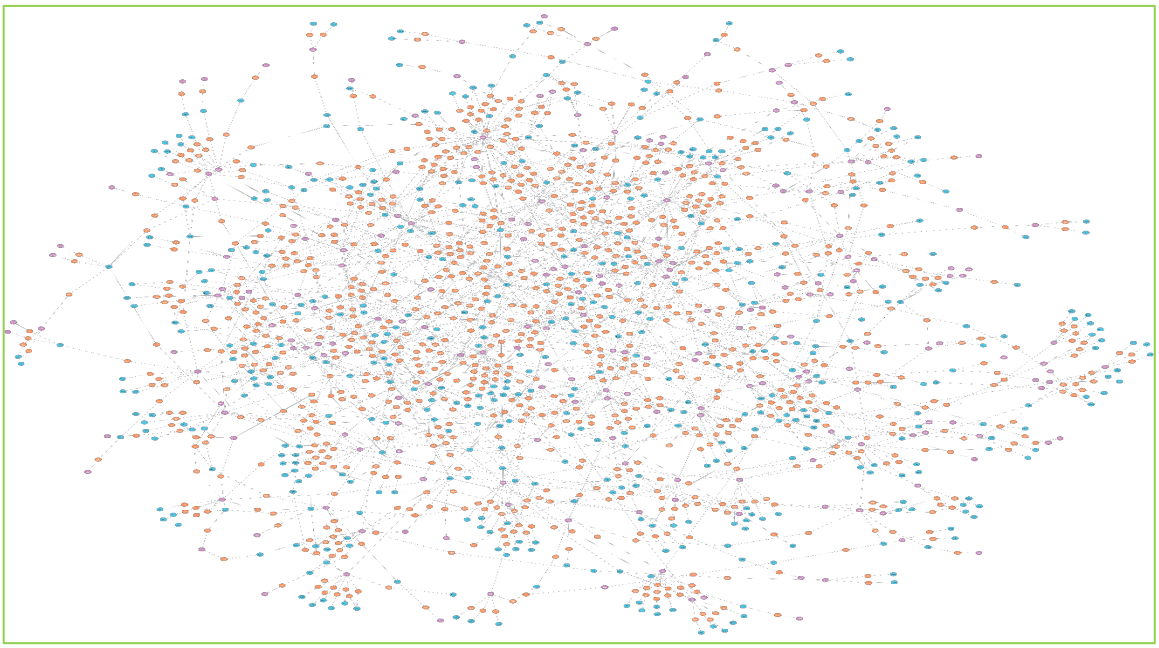


(xiii) Execution of nodes using DISTINCT function

```
CALL gds.wcc.stream('bollywood-graph')
YIELD nodeId, componentId
WHERE componentId = 1
RETURN  DISTINCT(gds.util.asNode(nodeId).name) AS name, componentId ORDER BY name
```

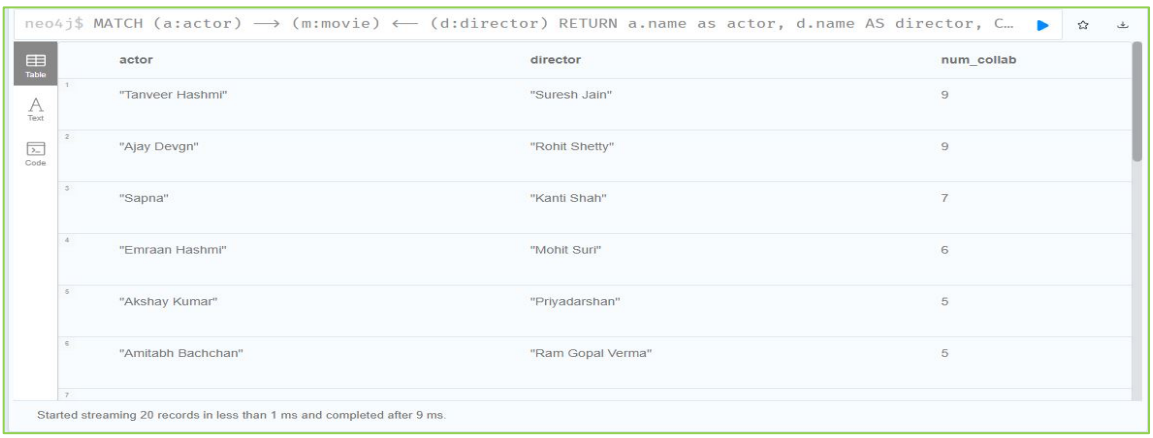
(xiv) The COLLECT function takes the names and turns them into a list. We then use a MATCH query to find only those names in the list, and filter the results.

```
CALL gds.wcc.stream('bollywood-graph')
YIELD nodeId, componentId
WHERE componentId = 1
WITH  COLLECT(DISTINCT(gds.util.asNode(nodeId).name)) AS name_list

MATCH path=(a:actor) --> (m:movie) <--(d:director)
WHERE (a.name IN name_list) AND (m.name IN name_list) AND (d.name IN name_list)
RETURN path LIMIT 2000;
```



(xv) This query returns the names of the actors and directors with the movies they did together

```
MATCH (a:actor) --> (m:movie) <-- (d:director)
RETURN a.name as actor, d.name AS director, COUNT(DISTINCT(m.name)) AS num_collab
ORDER BY num_collab DESC LIMIT 20;
```



neo4j$ MATCH (a:actor) ⟶ (m:movie) ⟵ (d:director) RETURN a.name as actor, d.name AS director, C…

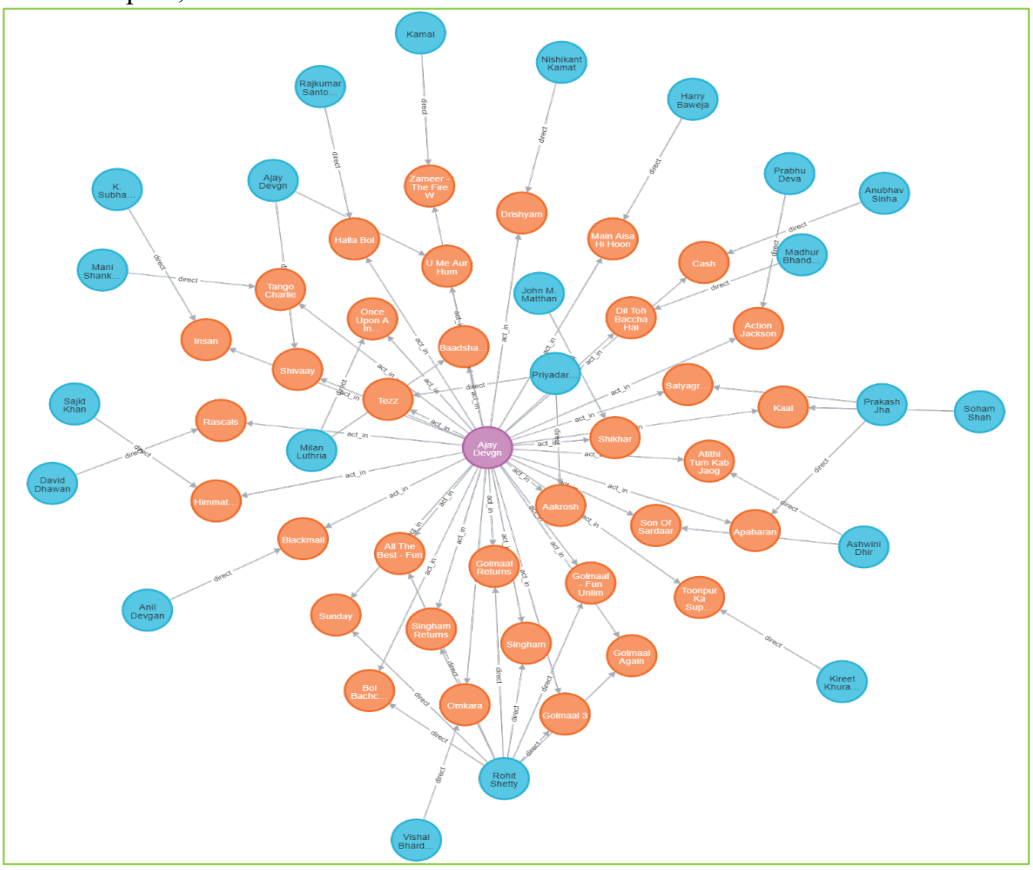
Table | Text | Code

	actor	director	num_collab
1	"Tanveer Hashmi"	"Suresh Jain"	9
2	"Ajay Devgn"	"Rohit Shetty"	9
3	"Sapna"	"Kanti Shah"	7
4	"Emraan Hashmi"	"Mohit Suri"	6
5	"Akshay Kumar"	"Priyadarshan"	5
6	"Amitabh Bachchan"	"Ram Gopal Verma"	5
7			

Started streaming 20 records in less than 1 ms and completed after 9 ms.

(xvi) The query returns the graph of movies and directors that had Ajay Devgan as a lead

```
MATCH path = (a:actor) --> (m:movie) <-- (d:director)
WHERE a.name = "Ajay Devgn"
RETURN path;
```



(xvii) Graph for movies done by Shah Rukh khan as a lead actor

```
MATCH path= (a:actor) --> (m:movie)
WHERE a.name= "Shahrukh Khan"
RETURN path;
```

(xviii) Graph execution for movies directed by Rohit Shetty in which lead actor was Akshay Kumar

```
MATCH path= (a:actor) --> (m:movie) <--(d:director)
WHERE a.name = "Akshay Kumar" OR d.name = "Rohit Shetty"
RETURN path;
```

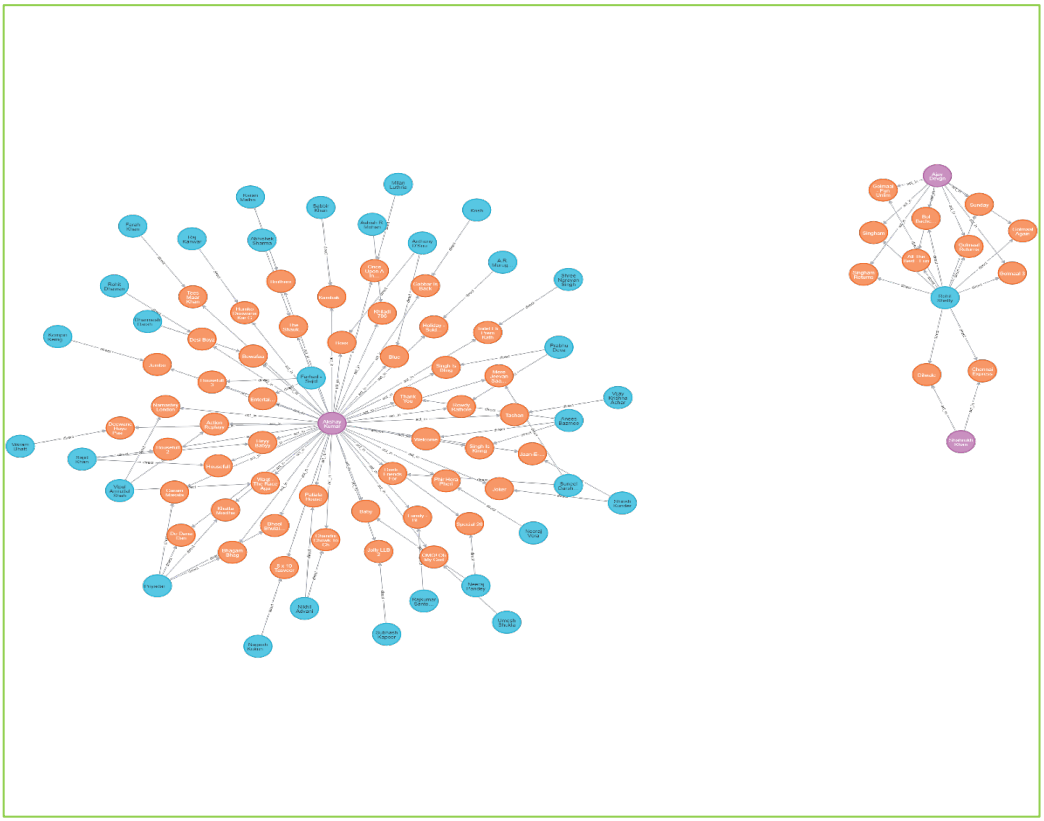


Figure 1

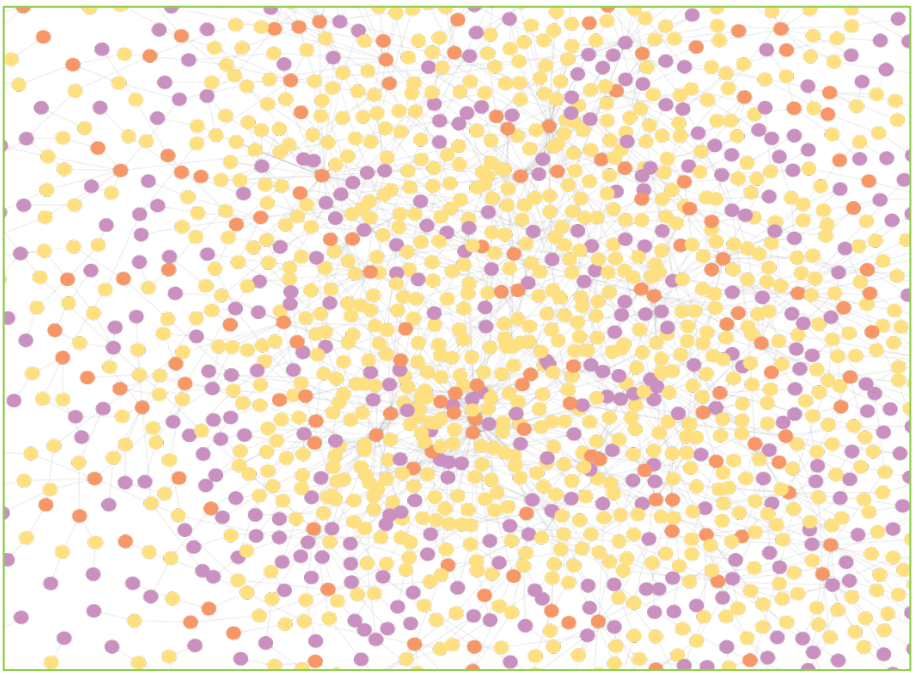


Figure 2 (Shows the bloom version of the graph above)

9. Conclusion

In SQL, we need to define relationships between tables explicitly. Cypher is a different kind of database than SQL. Cypher is more intuitive because we can formulate queries through the relationships without using foreign keys. In other words, Cypher makes us do things in networks instead of tables. And the graph visualizations in Neo4j Desktop and Bloom make it easy by letting us explore the relationships interactively.

This project shows how well Neo4j can handle relations-rich data and tabular data. It also shows how easy it is to use Cypher, a simpler language that is based on ASCII art. Together, these features make Neo4j a powerful tool for data analysis.

The Bollywood data also showed us some interesting facts about some of the biggest stars in Bollywood. For example, some of them have very impressive records, which could make watching their movies even more enjoyable in the future.

Neo4j 4.0 is a big improvement over previous versions of the graph database, and it will set the standard for all future graph database technology. This is because Neo4j 4.0 has been designed to be much faster and more reliable than previous versions, while also being secure and privacy-friendly. The modern application development process values speed and flexibility, so developers often rely on easy-to-use data abstractions to make it easy to translate between business needs and relational schemas. Graph databases are a great way to achieve this, and Neo4j is one of the most successful examples of this type of database.

10. References

1. https://www.researchgate.net/publication/307180380_Literature_review_about_Neo4j_graph_database_as_a_feasible_alternative_for_replacing_RDBMS
2. https://www.researchgate.net/publication/323720876_A_Survey_on_Graph_Database_Management_Techniques_for_Huge_Unstructured_Data
3. https://www.researchgate.net/publication/323720876_A_Survey_on_Graph_Database_Management_Techniques_for_Huge_Unstructured_Data
4. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7960078
5. https://www.computer.org/csdl/proceedings-article/icis/2017/07960078/12OmNzBOi3I
6. https://www.computer.org/csdl/proceedings-article/icis/2017/07960078/12OmNzBOi3I
7. https://neo4j.com/developer/get-started/
8. https://www.computer.org/csdl/proceedings-article/icis/2017/07960078/12OmNzBOi3I
9. https://www.semanticscholar.org/paper/Analysis-of-film-data-based-on-Neo4j-Lu-Hong/f307c76bc6399f91502f494b82eca1c5a498106f
10. https://www.kaggle.com/datasets/mitesh58/bollywood-movie-dataset
11. https://www.kaggle.com/general/231149
12. https://www.kaggle.com/code/adrianmcmahon/imdb-indian-movies-eda/notebook
13. https://academic.oup.com/database/article/doi/10.1093/database/baab026/6277712
14. https://www.nature.com/articles/s41599-020-0436-1