International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 216-219
Published Online June 2023 in IJEAST (http://www.ijeast.com)

Learning Graph Databases: Neo4j an overview

Dr. Dipali Meher and Dr. Pallawi Bulakh and Prof. Meenal Jabde
Modern College of Arts, Science and Commerce,
Ganeshkhind, Pune 411016, India

Abstract: **As the internet is growing day by day, the amount of data being generated is huge. This data includes structured data and unstructured data. The data along with its relationship with other data makes the most powerful and meaningful information. Maximum data exists in the form of the relationship between different or same objects and the noticeable thing is the relationship between the data is more important than the data itself. These relationships are handled efficiently by Relational databases that store data having structures and which have several records. The important point to be noted here is that these Relational database management Systems use tables with normalization concept. If the amount of data in such tables is huge, then handling such a large amount of data with its relationships is a tedious task. Here, Graph Databases come into picture. Entities and their relationships in relational databases will be reflected with nodes and relationships in graph databases. Graph databases provide very simple data model than databases with Online Transaction Processing systems. Graph databases provide features such as transactional integrity and operational availability. This paper introduces the idea of graph database systems in conjunction with Neo4j encompassing its query features, consistency, transactions, availability, and scaling.**

Keywords: **Graph databases, Relational databases, Neo4j, NoSQL, Features of Neo4j, Cypher query Language**

I. INTRODUCTION

1.1 Introduction to Graph Databases

Graphical databases replace relational databases in upcoming days. Graphical databases fall under the category of NOSQL. NOSQL stands for Not Only (Non) Structured Query Language[1]. Before 1990 relational databases were dominant soon after web development started, and the world of unstructured data started increasing. It has been found that relational databases do not fulfill the future requirement of web estate databases[2]. As this is the Web era of everything and hereafter drastic changes in databases will come due to customization of open-source databases for its use. Agility is the useful property of graph databases [3]. NoSQL databases are non-tabular databases and stores data indifferent format that relational databases. They come in variety of types which is based on their data model [4]. The main types of NOSQL databases are document, key-value, wide-column, and graph databases. Due to provision in database design and schema flexibility they store large amount of data using clustering concepts. As relational databases fail to provide clustering concept, hence does not suitable for distributed database concepts. After 1995, tremendous use of polymorphic data in web development has been started and business managers were started thinking to make this data for monitory gain[5]. Unfortunately, relational databases do not provide facility for storage of polymorphic data in clustered design. Carlo Strozzicomes with idea of NOSQL databases in 1995. Soon after Google and Amazon has developed their research papers on NOSQL databases and their use [6][7]. This is the turning point for relational databases and became a stepping stone for NoSQL databases.

Instead of tables and documents, graph database stores nodes and relationships. Data storing based on sketching of diagram. Data in graph database is stored without predefined model i.e., it allows flexibility of its storage, use and it can be rebuilt also[8][9].

Table 1 Difference between relational databases and graph databases

Key Point	Graph database	Relational database
Format	Nodes and edges	Tables with rows and columns
Relationships	Considered data, represented by edges between nodes	Related across tables, established using foreign key between tables
Complex queries	Run quickly and do not require JOINS	Always require complex joins between tables
Top use cases	Relationship focused use cases Fraud detection and recommendation engines	Transaction focused use cases Online transaction and accounting

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 216-219
Published Online June 2023 in IJEAST (http://www.ijeast.com)

Example		

1.2 Overview of Neo4j Graph Database

The world is now a connected enterprise. Everything is rich set of connections on what is happening. Many times, it has been found that connections between items are more important that the item itself. For example, WWW is a directed network contains nodes and edges. These nodes represent Web pages. Edges represents the hyperlinks between web pages. There exists an edge from one page to another page if one page contains at least one hyperlink pointing to another page. For example, social networks like Facebook – It links people according to various social relationships, like friendship, collaboration and colleague relation. Relational databases use tables to store these relationships may be in same table or in different table. For navigation of relationships JOIN operations and cross-lookup concept is necessary. All to do this requires very rigid database schema. Which means relations databases handles relationships poorly. In case of graph databases there will be no concept of JOIN or lookups. In graph databases relationships are stored natively alongside of data elements (data elements are known as nodes) in elastic format. In graph databases [10], everything about the graph is optimized for traversing through the data (nodes and relationships) very quickly, as millions of connections per second, per crore speed. In Graph databases there are many questions on which queries can be written are relationships rather than elements. Neo4j is very popular graph database that uses Cypher Query Language (CQL). Currently Neo4j is the world's principal, most widely used open-source graph database [11]. This database extremely scalable and schema less. Following table gives the difference between Graph databases and relational databases [12]. Neo4j is an open-source database with online backup and high availability. Neo4j is developed using JAVA language byNeo4j, Inc.

1.3 Nodes and relationship creation using Cypher Query Language(CQL)

Graph databases consists of nodes and edges. Nodes represent detailed entities and edges (relationships) are nothing but connections between two entities (i.e., two nodes) [13]. Every edge in graph databases must have directional significance which clearly identifies source and destination. Nodes are related with each other with some relationships they allow users to find fascinating patterns or paths among same or different types of nodes. In graph databases data is organized at one time then interpreted in various ways based on relationships created. A node can have unlimited number of relationships [14]. Relationships are known as first class citizens in graph databases. Relationships i.e. edges defines the value of graph. Relationships may also have properties for them. Using these properties users can add importance to the edges i.e. to the relationships. For example, since when did two peoples (means nodes in graph) know each other (may be friendship relation), the distance between the nodes, or information sharing aspects between the nodes. The graph is queried by using these relationships. Neo4j Online console is used to fire the queries. The link for using console is https://console.neo4j.org/. User has to first clear the database to start creation of own database structure.

Example, Node creation Syntax:

Create (alis: node type{attributed name1: attribute value1,attribute name 2: attribute value 2……}) return alias

Example: Create (: Person{name: "Dipali",age:39})

Create (: Person{name:"Pallawi",age:41}) Create (:area{name: "Pune"})

Relationship Creation Syntax: match(alias1:node type1),(alias: node type 2),…. Where alias1.attribute namex=alais2.attribute namex

Create (alias1)-[: relationship name{attributes of relationship name: values of attribute}]->(alias2) return alias,alias2…..

Example1 There are two nodes of person types. One person is a friend of another person. Here, friend of is the relationship.

match(p:Person),(pp:Person) where p.name="Dipali" and pp.name="Pallawi" create(p)-[:Friend_of]->(pp)return p,pp

Example 2 Another relationship: person lives in the city. Here, lives in a relationship.

match(p1:Person),(a1:area) where p1.name="Dipali" and a1.name="Pune" create(p1)-[:lives_in]->(a1)return p1,a1

1.4 Querying with Cypher Query Language to Neo4j database

Syntax for firing queries

Match (aliasx: nodetypex), (alaisy:node type y) Where (aliasx)-[:relationship name {relationship attributes checking if any}]->(aliasy)…. return aliasx, aliasy

Query1: List the names of people who are friend of each other

MATCH (p: Person), (pp:Person) Where(p)-[:Friend_of]->(pp) RETURN p.name,pp.name

Query 2: Display the names of people living in Pune.

Match (p1:Person),(a1:area) where a1.name="Pune" and (p1)-[:live_in]->(a1) return p1.name,a1.name

the output of both queries is as follows:

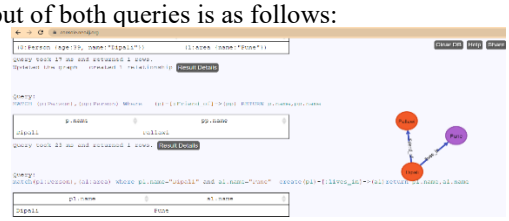

Fig. 1. Firing queries in Neo4j

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 216-219
Published Online June 2023 in IJEAST (http://www.ijeast.com)

1.5 Features of Neo4j

a) Consistency: Graph databases are operating on nodes which are connected with each other. Usually graph databases do not support node distributions of same graph on dissimilar servers at different locations. Infinite graph supports node distribution across a cluster of servers. Consistency will be clearly reflected on single server. Neo4jalways supports ACID properties. When Neo4J is run on a cluster, a write operation to the master is eventually synchronized to the its slaves. The slaves are always available for read operation. Immediate synchronization of write operations with master will be done. Other slaves have to wait for synchronization as they have to wait for the data to propagate from the master. Transaction concept will be used to ensure consistency in graph databases. They do not allow dangling relationships: The start node and end node always have to exist, and nodes can only be deleted if they don't have any relationships attached to them [15].

b) Transactions: As Neo4j is ACID-compliant. Transaction will be started before changing any nodes or adding any relationships to existing nodes. If operations are not wrapped in transactions then user will get a Not in Transaction Exception. Transaction initiation is not required for read operations. i.e. Neo4j supports Atomicity, Consistency, Isolation and Durability properties like relational databases.
example (db: database)
Transaction t1 = db.beginTx();
try { Node n1 = db.createNode(); node.setProperty("name", "Research Paper"); n1.setProperty("published", "2022"); t1.success();
} finally
{t1.finish();}
Explanation: A transaction on the database is started, A node is created and set properties on it, Marked the transaction as a success and finally completed it. If the transaction is not marked as success, then Neo4j assumes that transaction was a failure and rolls it back when finish is issued. When success is set without finish does not mean that transaction is committed. This is the different that how RDBMS manages the transaction and has to be keep in mind when development of an application.

c) Availability: High availability is achieved in Neo4j version 1.8 as it provides replication through slaves. These slaves can handle write operation. Write operation is first committed at the master slave secondly at the remaining slaves with synchronization manner. Apache Zoo Keeper will be used in Neo4j to keep track of the last transaction IDs persisted on each slave node and the current master node. To find out which server is master it communicates with Zoo Keeper. Server will become master if it is the first one to join the cluster. When a master slave goes down, the cluster elects a master from the available nodes, thus data is highly available.

d) Flexible schema: Neo4j graph database does not allow users to follow fixed schema. Users can add and remove properties or relationships at any time as per requirement.

e) Query Features: Neo4j using Cypher Query Language which uses ASCII for depicting graphs. keywords like Match, return, order, Aggregate, Limit and skip can be used.

f) Scalability and relatability: At any given point of time any number of nodes can be added or removed depending on read/write operations in query processing. Data Safely and reliability is supported by replication feature of this database. Sharding concept is used in NoSQL databases, where data is split and distribution is done across different servers, but in sharding becomes difficult in graph databases.

g) This database provides a **built-in web browser web application**. This database supports ample drivers that can work for REST API, Node JS and supports cipher API and Java API

h) Neo4j supports indexing by **Apache Lucene**.

II. CONCLUSION

Graph databases supports ACID properties hence they are odd man out in fish in NOSQL pond. Due to Agility property of graph databases, they will be used everywhere instead of relational databases. This paper gives an idea about how user can start self-learning for graph databases. By understating simple syntax of node and relationship creation any database administrator can start learning Neo4j graphical database and ultimately started using it for database creation. This paper also explains various features of Neo4j graph database and their simplicity.

III. REFERENCES

[1]. Biswajeet Sethi, Samaresh Mishra, P. ku. P. A Study of NoSQL Database. International Journal of Engineering Research & Technology, 67(6), 14–21,(2014).

[2]. Harpreet, K., Jaspreet, K., & Kamaljit, K. A Review of Non-relational Databases, Their Types, Advantages and Disadvantages. International Journal of Engineering Research & Technology (IJERT), 2(2), 1–5. https://www.ijert.org/a-review-of-non-relational-databases-their-types-advantages-and-disadvantages,(2013)

[3]. Paul, Subrata & Mitra, Anirban &Koner, Chandan. A Review on Graph Database and its representation. 2019 International Conference on Recent Advances in Energy-efficient Computing and Communication (ICRAECC) 1-5. 10.1109/ICRAECC43874.2019.8995006,(2019)

[4]. Bathla, G., Rani, R., & Aggarwal, H. Comparative study of NoSQL databases for big data storage. International Journal of Engineering and Technolo-

gy(UAE), 7(2), 83–87. https://doi.org/10.14419/ijet.v7i2.6.10072, (2018)

[5]. Jacksi, Karwan&Abass, Shakir. Development History of The World Wide Web. International Journal of Scientific & Technology Research. 8. 75-79,(2019)

[6]. Chang, fay, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. et.al., "BigTable: A Distributed Storage System for Structured Data." OSDI 2006 - 7th USENIX Symposium on Operating Systems Design and Implementation, 205–18,(2006)

[7]. DeCandia, Giuseppe &Hastorun, Deniz &Jampani, Madan &Kakulapati, et.al., Dynamo: Amazon's highly available key-value store. Operating Systems Review - SIGOPS. 41. 205-220. 10.1145/ 1294261. 1294281,(2007)

[8]. Patil, S., Vaswani, G., & Bhatia, A., Graph Databases-An Overview. International Journal of Computer Science and Information Technologies, 5(1), 657–660. www.ijsit.com, (2014)

[9]. Larriba-Pey, Josep-Lluis& Martínez-Bazan, Norbert & Domínguez-Sal, David. (2014). Introduction to Graph Databases. 10.1007/978-3-319-10587-1_4,(2014)

[10]. Agarwal Smita, Patel Atul, A study on graph storage database of NOSQL, International journal on Soft Computing, Vol.5, No.1, DOI :10.5121/ijscai.2016,(2016)

[11]. López, Félix & Cruz, Eulogio., Literature review about Neo4j graph database as a feasible alternative for replacing RDBMS. Industrial Data. 18. 135. 10.15381/idata.v18i2.12106,(2015)

[12]. Sadalage, PJ, and Martin Fowler, NoSQL, Distilled: A Brief Guide to the Emerging World of Polyglot Persistence. Vasa., Pearson Education,(2012)

[13]. Francis, N., Green, A., Guagliardo, P., Libkin, L., Lindaaker, T., Marsault, V., Plantikow, S., Rydberg, M., Selmer, P., & Taylor, A., Cypher: An evolving query language for property graphs. Proceedings of the ACM SIGMOD International Conference on Management of Data, 1433–1445. https:// doi.org/10.1145/ 3183713.3190657 (2018)

[14]. K. Saeed and W. Homenda (Eds.): CISIM 2015, Graph Databases: Their Power and Limitations, LNCS 9339, pp. 58–69, 2015. DOI: 10.1007/978-3-319-24369-6_5

[15]. NoSQL Databases | Research Paper | AZPapers