OUTCOME ANALYSIS IN ACADEMIC INSTITUTIONS USING NEO4J

Reshma K.R, Mary Femy P.F and Surekha Mariam Varghese

Department of Computer Science and Engineering, Mar Athanasius college of engineering, Kothamangalam, Kerala, India

ABSTRACT

Databases are an integral part of a computing system and users heavily rely on the services they provide. When interact with a computing system, we expect that data be stored for future use, that the data is able to be looked up fastly, and we can perform complex queries against the data stored in the database. Many different emerging database types available for use such as relational databases, object databases, key-value databases, graph databases, and RDF databases. Each type of database provides unique qualities that have applications in certain domains. Our work aims to investigate and compare the performance and scalability of relational databases to graph databases in terms of handling multilevel queries such as finding the impact of a particular subject with the working area of pass out students. MySQL was chosen as the relational database, Neo4j as the graph database.

KEYWORDS

Neo4j, NOSQL, Graph database

1. INTRODUCTION

The relational model has dominated the computer industry since the 1980s mainly for storing and retrieving data. Lately, however, relational database has been losing its importance due to its reliance on a strict schema which makes it difficult to add new relationships between the objects. Another important reason of its failure is that as the available data is growing manifolds, it is becoming complicated to work with relational model as joining a large number of tables is not working efficiently. One of the proposed solutions is to transfer to the Graph databases as they aspire to overcome such type of problems. This paper provides a comparative analysis of a graph database Neo4j with the most widespread relational database MySQL.

Relational databases store information in relations, which are a set of tuples that have the same attributes. Relations are then stored together in a table, organized by rows and columns.

- Features of relational database
    - Easy query language
    - Schema based
    - Data stored in rows & columns
    - Support ACID –Atomicity, Consistency, Isolation, Durability

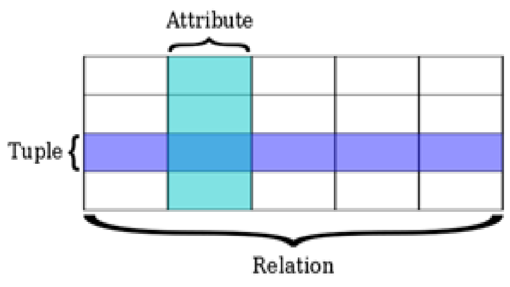

Figure 1: Relational database

Relational databases such as Oracle and MySql excel when it comes to capturing repetitive, tabular data. Despite the word “relational” in their name, relational database are much less effective at storing or expressing relationships between stored data elements. Unlike a relational database, a graph database is structured entirely around data relationships. Graph databases treat relationships not as a schema structure but as data, like other values.

A graph database contains a number of nodes, which have properties, and relationships exist between nodes in the graph, which can also have attributes.

- Features
    - Easy Query language-Cypher query language
    - Easy to represent connected data
    - Faster to retrieve/traversal/navigate connected data
    - Can represent semi structured data very easily
    - Cypher commands are human readable and easy to learn
    - Simple and powerful datamodel
    - Not require complex joins to retrieve connected/related data

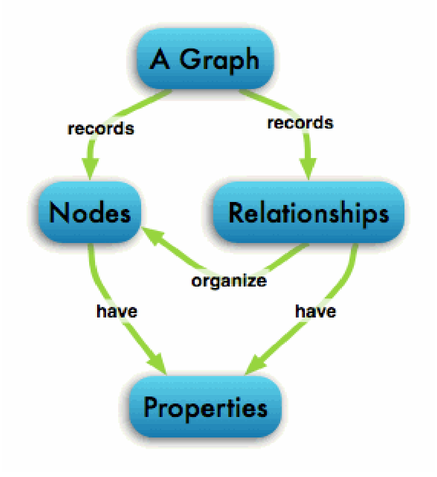

Figure 2:Graph database

The graph database queries are domain-specific user-friendly and can be considered as "SQL for graphs". The similarity to SQL is intentional and makes transition much easier for developers. When SQL query on the RDBMS is as long as half a novel, the Cypher Query equivalent is much shorter and intuitive. The traverser API in RDBMS is highly resource intensive, since each step to neighboring node has to be depicted with JOIN. In contrast, the graph database hypergraph property allows direct access to neighboring nodes by eliminating the edge attribute.

Graph databases support a graph model which allows a direct persistent storing of objects in the database together with the relations between them. A graph database should provide access to query methods that not only deal with the stored objects, but also with graph structure. The best known example of such an operation is traversal, which in its most simple form can be used to obtain the neighbors of an object, that is, the objects that are directly related to. In this paper we investigate the outcome analysis of pass out students in a particular institution using Neo4j and compare it with MySQL.

2. BACKGROUND

The relational database management system was created in the 1970s. Its popularity has skyrocketed, and it has become a primary data storage structure in academic and commercial pursuits. Relational databases range from small, personal databases like Microsoft Access to large database servers like MySQL, Microsoft SQL Server, and Oracle. This paper focuses on MySQL. Graph database researches was popular in the early 1990s, but died out for a series of reasons including the surge of XML research and hypertext. With the rise of the Internet as a tool for the public, data began to increase both in interconnectedness and in volume. The graph model was used to represent huge amounts of data more than it had in the past. Traditional data stores were capable of handling graph data. Yet, they were neither designed to do so nor efficient at it. There was clear desire for data store tailored to the needs of graph data.

In recent years, software developers have been investigating storage alternatives to relational databases. NoSQL is a term for some of those new systems. BigTable, CouchDB, Cassandra, Project Voldemort, and Dynamo are all NoSQL projects, as they are all high-volume data stores that reject the object-relational and relational models.

Atomicity, consistency, isolation, and durability (ACID) are a set of governing principles of the relational model. They guarantee database reliability. NoSQL rejects ACID. The term"NoSQL" as a term for modern web data stores, first began to gain popularity in early 2009. It is a topic that has gained recognition from IT community but has yet to garner large scale academic study. The NoSQL movement has its own discussion groups, conferences and blogs. As the typical database administrator attempts to question whether to move from the relational model to NoSQL model, the NoSQL community presents him or her with potential flags that data might be more suitable for a NoSQL system.

1. Having tables with lots of columns
2. Having attribute tables.
3. Having lots of many-to-many relationships.
4. Having tree-like characteristics.
5. Requiring frequent schema changes.

Data provenance meets several of these criteria, so it would be fitting to investigate NoSQL solutions to the provenance storage problem. An experiment was conducted to test the viability of NoSQL data store, Neo4j, for data provenance needs.

3. MOTIVATION

This paper is a comparison of the relative usefulness of the relational database MySQL and the graph database Neo4j to store graph data. A directed acyclic graph (DAG) is a common data structure to store data provenance information relationships. The goal of this study was to determine whether a traditional relational database system like MySQL, or a graph database, such as Neo4j, would be more effective as the underlying technology for the development of a data provenance system.

Graph database models can be characterized as data structures for the schema. The instances are modeled as graphs or generalizations. Data manipulation is expressed by type constructors and graph oriented operations. One of the motivations towards this paper is to provide a benchmarking mechanism to measure the effectiveness of graph traversal operations. It also motivates us to measure the capabilities of graph databases to perform query like traversal where one searches for topologically related vertices for a given vertex. It also searches the graph analysis/mining operations that require the traversal of the whole graph.

4. APPLICATIONS OF GRAPH DATABASE

Several areas have witnessed the emergence of huge data networks called complex networks. So graph databases are the best database to implement such complex network of relationships having millions of nodes and relationships. The main application areas of graph databases are:

1) Social networks: In social networks, nodes are people or groups, while links show relationships or flows among nodes. Some examples are friendships, business relationships, research networks, communication records (mail, telephone calls and email), computer networks, and national security. There is growing activity in the area of social network analysis and also in visualization and data processing techniques for these networks.

2) Information networks: Information networks model relations representing information flow, such as citations among academic papers, World Wide Web, peer-to-peer networks, relations among word classes in a thesaurus, and preference networks.

5. ADVANTAGES

The benefits of using a graph data model are given by: the introduction of a level of abstraction which allows a more natural modeling of graph data; query languages and operators for querying directly the graph structure; and ad-hoc structures and algorithms for storing and querying graphs. Graph databases are also somewhat similar to object databases in case where objects and relationships between them are all represented as objects with their own respective sets of attributes. Graph database consists of several advantages:

- It enables very fast queries when the value of the data is the relationships between people/items.
- Use Graph Databases to identify relationships between people/items, even when there are many degrees of separation.
- Where the relationships represent costs, identify the optimal combination of groups of people/items.

6. PROPOSED SYSTEM

Neo4j, like many other graph databases, builds upon the property graph model; labeled nodes (for informational entities) are connected via directed, typed relationships. Both nodes and relationships hold arbitrary properties (key-value pairs). There is no rigid schema, but with node-labels and relationship-types we can have as much meta-information as we like. When importing data into a graph database, the relationships are treated with as much value as the database records themselves. This allows the engine to navigate your connections between nodes in constant time. That compares favourably to the exponential slowdown of many-JOIN SQL-queries in a relational database.

Proposed System consists of research and comparison of two databases such as Neo4j and MySQL databases. A graph database stores data in a graph, the most generic of data structures, capable of elegantly representing any kind of data in a highly accessible way.

MySQL has significant market penetration in the academic and scientific fields. Furthermore MySQL has significant support, both from the manufacturers and from the user community. It is a pure relational database, as opposed to an object-relational database like Oracle and SQL Server.

Neo4j is open source for all noncommercial uses. It has been in production for over five years. It is quickly becoming one of the foremost graph database systems. According to the Neo4j website, Neo4j is “an embedded, disk-based, fully transactional Java persistence engine that stores data structured in graphs rather than in tables”. The developers claim it is exceptionally scalable (several billion nodes on a single machine), has an API that is easy to use, and supports efficient traversals. Neo4j is built using Apache’s Lucene for indexing and search. Lucene is a text search engine, written in Java, geared toward high performance.

A. Types of Graph Database Models

1) Neo4j Graph Database: As a robust, scalable and high-performance database, Neo4j is suitable for full enterprise deployment or a subset of the full server can be used in lightweight projects.

It features:

- Intuitive using a graph model for data representation
- Reliable
- Durable and fast, using a tradition disk Based native storage engine.
- Extraordinarily scalable, up to several billion nodes/relationships/properties.
- Expressive with a powerful, human readable graph query language
- Fast with a powerful traversal framework for high speed graph queries.
- True ACID transactions
- High availability
- Scales to billions of nodes and relationships
- High speed querying through traversals

Proper ACID behavior is the foundation of data reliability. Neo4j enforces that all operations that modify data occur within a transaction, guaranteeing consistent data. This robustness extends from single instance embedded graphs to multi-server high availability installations. Neo4j is a commercially supported open-source graph database. It was designed and built from the ground-up to be a reliable database, optimized for graph structures instead of tables. Neo4j is based on the data model of a directed multigraph with edge labels and optional node and edge properties.

Node and links can be changed but have identity maintained by DBMS. Labels and property keys are strings, property values can be primitive java data types and strings or arrays of both. The fundamental units that form a graph are nodes and relationships. In Neo4j, both nodes and relationships can contain properties. Nodes are often used to represent entities, but depending on the domain relationships may be used for that purpose as well.

Neo4j's query language Cypher aims to be a user-friendly language that is designed to be read and understood easily. It allows you to declare patterns `(MATCH)` that you want to find in the graph and then apply filters `(WHERE)`, projection `(RETURN)` and paging `(LIMIT,SKIP,ORDER BY)` to your result data.

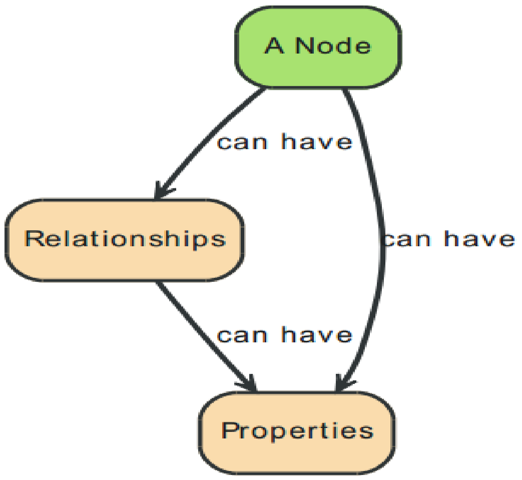

Fig.3. Neo4j Graph Database Nodes and relationships

B. Graph Databases and Their Support for Querying Graphs

1) *Adjacency Queries:* In this type of queries the primary notion is node/edge adjacency. Two nodes are adjacent when there is edge between them.

2) *Reachability Queries:* These queries are characterized by path or traversal problem. The problem causes in reachability test whenever two given nodes are connected to path.

3) *Pattern Matching Queries:* Pattern matching queries find all sub-graphs of data graph that are isomorphic to pattern graph.

4) *Summarization Queries:* Summarized queries are not related to consult the graph structure. They are based on special functions that allow summarizing or operating on the query results, normally returning a single value.

7. EXPERIMENTAL EVALUATION

A. Setup: Computer Configurations and Datasets

We used core i3 processor, 2 GB of RAM and 10 GB SATA for implementing Neo4j graph database. Here we used institution data for evaluating Neo4j.

Our work aims to investigate and compare the performance and scalability of relational databases to graph databases in terms of handling multilevel queries such as finding the impact of a particular subject with the working area of pass out students. MySQL was chosen as the relational database, Neo4j as the graph database. Figure 4 shows institutional database

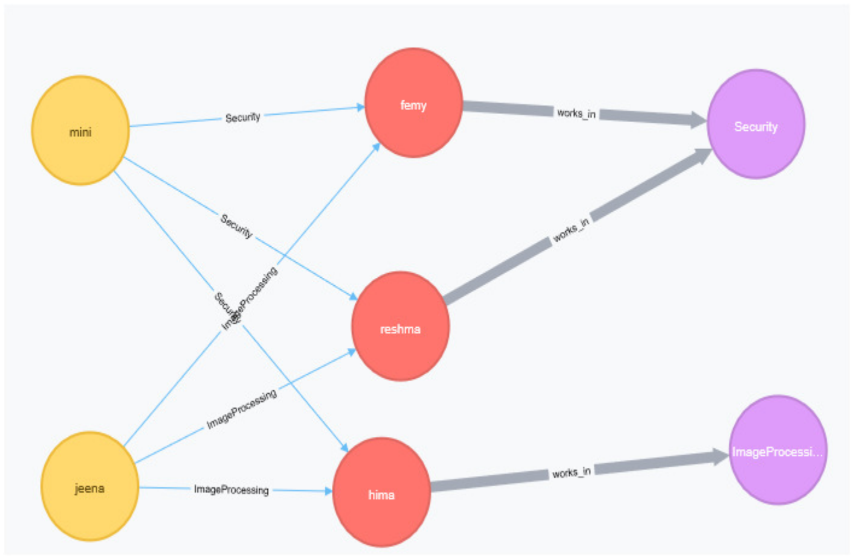

Figure 4: Institutional database

B. Global Query

The queries were designed to simulate some of the types of queries used in provenance systems. For example, traversals are necessary to determine data objects (nodes) derived from or affected by some starting object or node. That is, if a data object is determined to be incorrect, that information must be propagated to all "descendants" of the node. Searching for specific values within the payload is another common operation.

The performance of a global constraint based user lookup was constructed to measure the performance of queries typically issued on databases. The intent of the global query was to characterize the performance of queries requiring inspection of all users in the system.

For each of the queries with varying dataset sizes, 100 data points were collected. The resulting data points were then averaged to summarize the data collected for the particular test.

1). MySQL Global Query

The global query that was run against the MySQL database was intended to return all of the nodes in the systems that were between a provided age group. The query utilized is as follows (where the two question marks were the lower limit on the age and the upper limit):
SELECT count(*) FROM student_node WHERE student_node.age > ? AND student_node.age <?;

2). Neo4J Global Query

The global query that was run against the Neo4j database was aimed at attaining the same data as the MySQL global query. That is, users in the database that are within a given age range. The query utilized is as follows (where 'X' stands for the minimum age and 'Y' stands for the maximum age):

```
START x=node(*) WHERE (x.age? > X and x.age? < Y ) return count(*);
```

The following query returns the student nodes whose working area matches with the area of a particular teacher.

```
match (t:teacher)-[r:teaches {sub:"Security"}]->(s:student)-[w:works_in]->(a:area {name:"Security"})return s
```

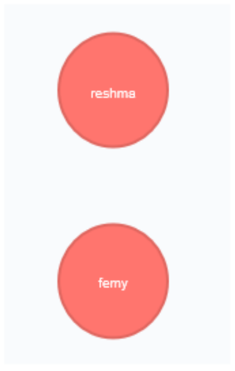

Figure 5: Result of Neo4j query

C. Query Performance

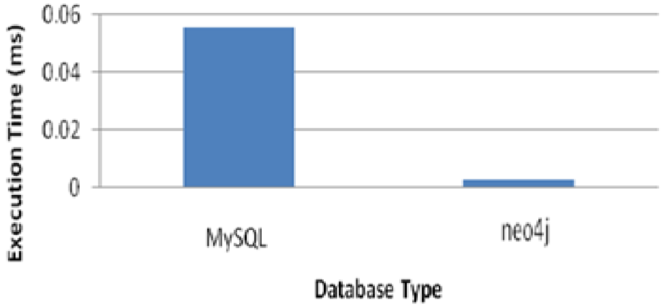

Fig.2.Global Query Performance

In Figure, the global query execution time is given for the neo4j database vs. the MySQL database. For this test, queries were run against the entire underlying database that looked for a range of random ages, which was run a total of 100 times with differing age ranges. The figure illustrates that the average execution time for the neo4j global query and the MySQL global query. The data shows that the neo4j execution time was an order of magnitude larger.

8.CONCLUSIONS

Graph databases are specifically designed to handle graph based data more efficiently than the traditionally adopted relational databases. One such specific type of data is social networking data, or any type of data that is highly dependent on relationships between collections of nodes. In particular, this study compared the performance of neo4j vs MySQL.

In general, the graph database did better at the structural type queries than the relational database. In full-text character searches, the graph databses performed significantly better than the relational database. The fact that the indexing mechanism used in the graph database was based on strings made the numeric queries less efficient. While a query on non-integer numeric data, such as doubles, was not included in the benchmark tests, the result would have likely been even worse for the graph database.

Graph databases exhibit the ability to scale exceptionally well with large numbers of nodes and/or relationships, whereas MySQL, or presumable any relational database begins to see a performance degradation with input data. The input data utilized in this experiment was comprised of only a few columns that represented friendship relationships between users. However, graph databases are easily transformed to contain many relationships amongst the nodes and also to have many attributes tied to any given node and/or relationship. In order to provide this in a relational database, many more relations (tables) need to be added to the underlying database, which has an impact on the system's performance.

In this paper we investigate and compare the performance and scalability of relational database to graph database by finding the impact of a particular subject with the working area of pass out students. In fact, Neo4j is best suitable to implement complex network of relationships having millions of nodes and relationships.

REFERENCES

[1] Renzo Angles And Claudio Gutierrez." Survey of Graph Database Models"ACM Computing Surveys, Vol. 40, No. 1, Article 1, Publication date: February 2008.

[2] Renzo Angles. "Comparison of Current Graph Database Models", Department of Computer Science, Talca.

[3] "Short overview on the emerging world of graph databases,"http://www.graph-database.org/overview.html.

[4] "Neo4j," http://neo4j.org/.

[5] P. Urb ´ on, "Nosql graph database matrix," http://nosql.mypopescu.com/post/619181345 /nosql-graph-databasematrix, May 2010.

[6] "NOSQL Databases", http://nosql-database .org/

[7] D. Dominguez-Sal, P. Urb on-Bayes, A. Gim enez-Va n o, S. G omezVillamor, N. Mart ınez-Baz an, and J. L. Larriba-Pey, "Survey of graph database performance on the hpc scalable graph analysis benchmark," in Proc. of the 2010 international conference on Web-age information management (WAIM). Springer-Verlag, 2010, pp. 37–48.

[8] Neo4j B log, Internet: http://blog. neo4j.org/2009/04/current-database-debate-andgraph.html ,201 0.

[9] Neo4jmanual, Internet: http://docs. neo4j.org/chunked/stabl e/graphdb- neo4jnodes. html ,2010

[10] J. Paredaens and B. Kuijpers, "Data Models and Query Languages for Spatial Databases," Data & Knowledge Engineering (DKE), vol. 25, no. 1-2, pp. 29–53, 1998.

AUTHORS

Reshma K R is currently pursuing M.Tec h in Computer Science and Engineering in Mar Athanasius College of Engineering. She completed her B.Tech from MEA Engineering College, Perinthalmanna. Her areas of research are Data Mining, Databases and Image Processing.

Mary Femy P F is currently pursuing M.Tech in Computer Science and E ngineering in Mar Athanasius College of Engineering. She completed her B.Tech from Matha College of Technology, Paravur. Her areas of research are Data Mining, Databases and Image Processing.

Surekha Mariam Varghese is currently heading the Department of Computer Science and Engineering, M.A. College of Engineering, Kothamangalam, Kerala, India. She received her B-Tech Degree in Computer Science and Engineering in 1990 from College of Engineering, Trivandrum affiliated to Kerala University and M-Tech in Computer and Information Sciences from Cochin University of Science and Technology, Kochi in 1996. She obtained Ph.D in Computer Security from Cochin University of Science and Technology, Kochi in 2009. She has around 25 years of teaching and research experience in various institutions in India. Her research interests include Network Security, Database Management, Data Structures and Algorithms, Operating Systems and Distributed Computing, Machine learning. She has published 17 papers in international j ournals and international conference proceedings. She has been in the chair for many international conferences and journals.