Comparative Analysis of Graph Database and a Relational Database

Neha Tyagi¹, Neelam Singh²
M.Tech Scholar¹, Assistant Professor²
Dept of Computer Science & Engineering
GEU Dehradun, India

Abstract:
Database can accommodate a very large number of users on an on-demand basis. The field of database technologies has undergone drastic changes. The advent of complex connected data has given rise to database technologies that diverge from the traditional relational databases (RDBMS). Graph-based database, which abstracts data in the form of nodes, edges and properties, is one such innovative idea for highly associative data. In this paper, we have described a graph database model based on the graph theory and a prototype implementation of the same. The prototype implementation has a complete set of insertion, selection and deletion graph queries and a cache for fast data processing and retrieval. RDBMSs are a major technology that stores large portions of structured data which rely on providing ad-hoc querying facilities by using Structured Query Language (SQL). Oracle, MS SQL Server, MS Access, MySQL as well as Sybase are examples of relational database management systems which are utilized to store, manipulate and retrieve data. In this paper we will do theoretical analysis of Graph base database and relational base database.

Keywords: DBMS, RDBMS, Graph database etc.

I. INTRODUCTION

A database management system is the software that allows a computer to perform database functions of storing, retrieving, adding, deleting and modifying data. It provides facilities for controlling data access, enforcing data integrity, managing concurrency control, recovering the database after failures and restoring it from backup files, as well as maintaining database security. Databases have been in use since many decades. The vast majority of older systems were tightly linked to the custom databases in order to gain speed at the expense of flexibility.

Originally DBMSs were found only in large organizations with the computer hardware needed to support large data sets. A database is an integrated collection of data records, files, and other database objects [1].

A database engine may comply with a combination of any of the following:

- The database is a collection of tables, files or datasets.
- Each table is a collection of fields, columns or data items.
- One or more columns in each table may be selected as the primary key.
- There may be additional unique keys or non-unique indexes to assist in data retrieval.
- Columns may be fixed length or variable length.
- Records may be fixed length or variable length.
- Table and column names may be restricted in length (8, 16 or 32 characters).
- Table and column names may be case-sensitive.

A. Databases:

Databases are defined as organized collections of data. Although when using the term database we refer to the entire database system, the term actually refers only to the collection and the data. The system which handles the data, transactions, problems or any other aspect of the database is the Database Management System (DBMS).

Early designs and implementations were based on the use of linked lists to create relations between data and to find specific data. These models were not standardized and required extensive training in order to make efficient use of them. These models and other important types are explained briefly as well.

II. RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)

RDBMS is a database developed by E.F. Codd [1]. RDBMS having tabular schemas which contain arity (rows) and cardinality (columns), before relational databases, the file based approaches are used which had various problems like as data dependencies which were overcome by RDBMS. Some categories of data integrity exist with each relational database management system like as entity integrity, domain integrity, referential integrity and user-defined integrity. Structured query language (SQL) is used as query language for most common relational databases. RDBMS has 3-layer architecture depicted in Figure 1.1 which shows the layers name and their corresponding work. Primary key is one which defines the table uniquely.
A. Design of RDBMS: Relational databases use the notion of databases separated into tables where each column represents a field and each row represents a record. Tables can be related or linked with each other with the use of foreign keys or common columns. On an abstract level tables represent entities, such as users, customers or suppliers. This abstraction is helpful when designing the database schema as real-world objects need to be mapped to the database in addition with the relations between them. The design of a database schema can be visualized using diagrams such as the one in Figure 2.

Figure 1: Three-Layers Of Relational Database

- **First Normal Form (1NF):** Eliminate groups of repeating data by creating a new table for each group of related data which is identified by a primary key.
- **Second Normal Form (2NF):** If a set of values are the same for multiple records move them to a new table and link the two tables with a foreign key.
- **Third Normal Form (3NF):** Fields which do not depend on the primary key of a table must be removed and if necessary be put into another table. It is also necessary for a database schema to satisfy the 2NF to first satisfy 1NF and the same applies for 3NF correspondingly. While there are other forms a schema is considered normalized if it satisfies the above 3 conditions.

B. Architecture of RDBMS:
From Figure 1 it is clear that internal layer includes the storage schema of database. It is responsible for data compression, encryption and decryption of data. Conceptual layer has the schema of table. It defines the fields and column attributes. It hides the details of internal layer. External layer is the user view. RDMS is popular due to ACID properties [4]. Figure 2 explains about the ACID properties. The developers of the components that comprise the transaction are assured that these characteristics are in place.

Figure 3 represents the core content of the ACID properties. From the figure it is clear that the atomicity asks for transaction complete to be either 100% or 0%, consistency says data should be consistent before and after each operation. Isolation means running of multiple transactions and durability means redo or undo transactions when system crashed.
- **Atomicity:** Either all parts of a transaction must be completed or none.
- **Consistency:** The integrity of the database is preserved by all transactions. The database is not left in an invalid state after a transaction.
- **Isolation:** A transaction must be run isolated in order to guarantee that any inconsistency in the database involved does not affect other transactions.
- **Durability:** The changes made by a completed transaction must be preserved or in other words be durable.

III. GRAPH DATABASE

In contrast to the other No-SQL implementations, in a graph database the relations between the objects are of primary importance. Graph databases support a graph model which allows for a direct persistent storing of the particular objects in the database together with the relations between them. In addition, a GDB should provide an access to query methods that not only deal with the stored objects, but also with the graph structure itself. The best known example of such an operation is traversal, which in its most simple form can be used to obtain the neighbors of a specified object, that is, the objects that the specified object is directly related to.

A. Graph model:
The topology of a graph G can be expressed as G = (V, E), where V is the set of vertices (also called nodes) and E is the set of edges (or relations). An edge connects two vertices which both have to exist, i.e. no dangling relations are allowed. Finally, a graph can be directed or undirected, which denotes whether the graph’s edges have a direction or not. In addition to the topology, there usually is more information contained within the graph, amount of which distinguishes basic graph models: Labeled graph. All the edges have a label which denotes the type of the edge. There also is a Vertex-labeled graph variant where even vertices can have labels.
- **Multi-graph:** Multi-graph is a labeled graph where multiple edges can exist between any two vertices, granted that these edges have different labels.
- **Attributed graph:** Graph elements can have attributes (also called properties) appended to them to carry additional

Figure 2. Database Schema Example

An important design aspect of relational databases is the normalization of the schema. This involves 3 steps which are described below:

- First Normal Form (1NF): Eliminate groups of repeating data by creating a new table for each group of related data which is identified by a primary key.
- Second Normal Form (2NF): If a set of values are the same for multiple records move them to a new table and link the two tables with a foreign key.
- Third Normal Form (3NF): Fields which do not depend on the primary key of a table must be removed and if necessary be put into another table. It is also necessary for a database schema to satisfy the 2NF to first satisfy 1NF and the same applies for 3NF correspondingly. While there are other forms a schema is considered normalized if it satisfies the above 3 conditions.
data. Properties are usually expressed as a map of keys and their associated values.

- Property graph is defined as an attributed directed multi-graph and it is the model which is implemented by the majority of graph databases [16].

B. Neo4j:
Neo4j is a high performance, robust and scalable graph database solving queries with multiple relationships storing data in the nodes and relationships [9]. Neo4j is fully written in Java and can be deployed on multiple systems [10]. It is comprised of two parts (1)- A client that sends commands to the server via RMI, (2)- A Server that processes these commands and sends back the processed result to the client. The database is queried through Cypher Query Language. It is a new query language that has been recently added to the Neo4j. Unlike imperative languages like Java and scripting language like Gremlin and the Ruby Neo4j bindings, Cypher is a declarative language. Using Cypher, efficient querying of the graph is possible, without having to write traversers in the code.

C. Challenges in graph databases:
Though studied previously, the search for efficient methods of answering graph queries remains open. Essential problems include: (1) how to store graph databases efficiently, (2) how to define similarity between graphs, and (3) how to create efficient index structure to accelerate pattern matching and graph similarity search. A primary challenge in pattern matching is that pairwise comparisons of graphs are usually hard problems. Sub-graph isomorphism is known to be NP-complete. As a result, most meaningful definitions of similarity will result in NP-hard problems. Even a relatively simple comparison, graph isomorphism, defies a polynomial bound for the general case. These costly pair-wise comparisons, when combined with the increasing size of modern graph databases, makes finding efficient search techniques difficult. Even more complications arise from the many classifications of graphs and the fact that some techniques apply only for a specific class.

IV. LITRATURE SURVEY

Dr. Mrs Pushpa Suri and Mrs Meenakshi Sharma [6] In this paper we have discussed the concept of object oriented data warehousing. For complex data the object oriented database management system is faster and better than relational database management system. For object oriented database management system, the goal is to direct the persistent objects manipulations by the application software. The inseparable practices are the operational burdens and the semantic burdens who attend the direct manipulation. Data warehouses contain consolidated data from many databases and other external data sources, spanning long time periods, and augmented with summary information.

Konstantinos Barmpis, Dimitrios S. Kolovos [7] this paper has explored the use of graph-based No-SQL databases to support scalable persistence of large models by exploiting the index-free adjacency of nodes provided by these stores. Prototypes for integrations of both Neo4J and Orient DB with EMF as well as with Epsilon's EMC have been implemented and described in detail. Benchmarks using the Grabats 2009 query have been executed and have shown that:

- Native queries provide performance results which greatly surpass XMI text file based stores and can store models larger than CDO.
- Back-end independent queries (using Epsilon's EMC) provide a maintainable and performing alternative to native querying. These results can promote further research and development of large-scale model persistence solutions based on graph-based No-SQL databases, a feasible preformat alternative to XMI.

Azhi Faraj, Bilal Rashid [8] The purpose of this paper is to demonstrate the differences of response time in relational and non-relational databases by executing a number of queries on the target systems, namely Oracle and MongoDB. Our results show that for such a high number of records MongoDB does a better job while retrieving data however it fails to exceed Oracle's speed when it comes to aggregation functions. This proves that NoSQL databases are no replacement for relational databases, the two types can coexist and it is only the needs of the customer that specifies which one suits better. This research can be further enhanced by involving several NoSQL database systems and using a higher number of records.

Ali Salehnia [9] In this paper the characteristics of Relational Database Systems and Big Data were presented. Also, the types of data each approach has to deal which were discussed. The security issues in each model were presented. The operational issues such as scale, performance, and availability of data by utilizing these database systems were also compared.

Mohammad Beydoun and Ramzi A. Haraty [10] they proposed algorithms are a great start and a first step in identifying future trends of graph usage in relational databases. The world of relational databases and graphs are slowly merging into one. Hence, we see the growth of some graph engines that are used to store relational database information and vice-versa [9]. This growth is starting to materialize in everyday’s business problems. On one hand most data-mining systems rely heavily on crawling and traversal from one bit of information to the other. On the other hand data storage systems (DBMS) are here to stay since they provide a resilient and accessible system for fast data retrieval and safe data storage.

Ayush Dubey, Greg D. Hil and Robert Escriva [11] This paper proposed refinable timestamps, a novel, highly scalable mechanism for achieving strong consistency in a distributed database. The key idea behind refinable timestamps is to enable a coarse-grained ordering that is sufficient to resolve the majority of transactions and to fall back on a finger-grained timeline oracle for concurrent, connecting transactions. Weaver implements refinable timestamps to support strictly serializable and fast transactions as well as graph analyses on dynamic graph data. The power of refinable timestamps enables Weaver to implement high-performance applications such as Coin-Graph and RoboBrain which execute complicated analyses on online graphs.

Joseph V. Homan, Robert Morris [12] This paper compares the relational database model with the associative database model. This paper briefly summarizes the relational and other familiar data models. The remainder of this paper will introduce and describe the associative database model. The associative model is less known because it is relatively new and does not have a large software supplier base. While it seems to offer a number of benefits and advantages over other
database structures, it has yet to become a commercial success in the mainstream database market.

Cornelia Győrödi, Robert Győrödi and Roxana Sotoc [13]

In this paper, we showed the results of different operations that had been applied for MongoDB and MSSQL databases. MongoDB provided lower execution times than MSSQL in INSERT, UPDATE and DELETE operations, which is essential when an application should provide support to thousands of users simultaneously. The only time when the MSSQL obtained an advantage was with the SELECT operations, the other ones gave advantages to MongoDB. We can also notice that the difference between the results of each database was not noticeable until around 1,000 records. Thus, we can say that relational databases, namely MSSQL is suitable for small and medium applications. Relational databases are widely used in most of the applications and they have good performance when they handle a limited amount of data. Angela Bonifati Radu Ciucanu Aurélien Lemay [14] We have studied the problem of learning path queries defined by regular expressions from user examples. We have identified fundamental difficulties of the problem setting, formalized what means for a class of queries to be learnable, and shown that the above class enjoys learn ability. Additionally, we have investigated an interactive scenario, analyzed what means for a node to be informative, and proposed practical strategies of presenting examples to the user. Finally, we have shown the effectiveness of the algorithms and the improvements of using an interactive approach through an experimental study on both real and synthetic datasets. [15] We compared the performance of a relational database (implemented in MySQL) and a graph database (implemented in Neo4j), in the context of a health application for personalized cancer treatment in Costa Rica. A detailed description of the methodology was offered, including the experimental setup. The comparison encompassed twelve queries and three data size configurations: all tables or node types with 1,000 entries, 10,000 entries and 100,000 entries. The results of the experiment indicate that MySQL performs better than Neo4j in most cases, but Neo4j outperforms MySQL in two queries that require multiple join operations when data size is 100,000 entries per table or node type.

V. SYSTEM MODEL

A. RDBMS to MODEL:

Transformed into a graph databases using a very well-known example of RDBMS. We assume that we have three tables, Sailors, Boats and Reserves, whose schemas are given below:

- Sailors(sid:integer, sname:char(10), rating:integer, age:real)
- Boats(bid: integer, bname: char(10), color: char(10))
- Reserve(sid:integer, bid: integer, day: date)

The data available in the database is shown in the following tables

![Figure 4. some part of the corresponding part is shown in.](image)

Figure 5. Partial GDB For Sailors, Reserves, And Boats

The graph above can be described formally using several extensional definitions such as:

- ID1 : (sid=22, name="Dustin", rating=7, age=45.0).
- ID6 : (bid=101, bname="Interlake", color="red").

We show how to deal with foreign keys in Figure 4.2 Here we construct edges corresponding to Foreign keys between IDs from different related tables, and thus the original edges representing the Foreign Key attributes in the small subgraphs are deleted.

The formal definition for a sub-graph involving foreign keys is as follows:

- ID4: (sailor=ID1, day="10/10/96", boat=ID6).

The storage necessary for the GDB is at most three times bigger than the storage necessary for the corresponding relational database, as we store at most two more nodes for each instance value in a database. However, GDB come with some advantages in terms of dealing with the problems we are interested in.

![Figure 6. foreign keys representation](image)

Figure 6. foreign keys representation

B. ASSESSMENT PARAMETERS FOR GRAPH DATABASE AND RELATIONAL DATABASE:

The assessment between MySQL and Neo4j is based upon different criteria [17]. These criteria are the pillars to decide which database should be adopted for implementation.

- Level of Support/Maturity Maturity refers to how well the system is tested. If a system has been tested, number of times, it means it is more secure and more bugs have been found out. Maturity of a system is proportional to level of support. Relational Databases have been providing storage support for decades now. So they are more stable and mature. Relational databases have a unified language SQL. As SQL
does not differ much between implementations, support for one implementation is applicable to others as well. Since Neo4j version 1.1 was released in February 2010, it is less stable and less mature. Neo4j is still growing and maturing and has not undergone the same rigorous performance testing as relational databases. Most of the support comes from its parent company’s website www.neo4j.org and is limited from outside of Neo4j site.

- Security MySQL has extensive multi user support. However Neo4j does not have any built in mechanisms for managing security restrictions and multiple users [17]. It presumes a trusted environment. Although there are Access Control List security mechanisms but even Access Control List management is handled at the application layer. On the other hand, there is extensive support for ACL based security in MySQL.

- Flexibility Although relational databases are more mature and secure as compared to graph databases, but its schema is fixed, which makes it difficult to extend these databases [18] and less suitable to manage ad-hoc schemas that evolve over time

VI. DISCUSSION AND CONCLUSION

Literature about the various database models is described. Prior to the recent announcement by Ingres, only two software vendors have offered commercial products to support DBMS. It seems likely that the future needs of users of large database systems will drive them toward database systems based on the associative data model. Subsequent research and papers should focus on quantifiable advancements and specific benefits derived from the use of ADBMS database software. Both systems performed well on the objective and subjective benchmarks. In general, graph databases performed better when objective tests were performed. This implies that graph databases retrieve the results of the set of predefines query faster than relational databases. Not only this, graph databases are more flexible than relational databases as new relationships can be added to graph databases without the need to restructure the schema again. With such a difference in the query retrieval time of MySQL and Neo4j, Neo4j can be used for commercial purposes like website link structures and social networking.

VII. REFERENCES


[5]. Marko A. Rodriguez, April 8, 2009, —Graph Databases and the Future of Large-Scale Knowledge Managementl, Los Alamos National Laboratory.


[9]. Azhi Faraj, Bilal Rashid “COMPARATIVE STUDY OF RELATIONAL AND NON- RELATIONS DATABASE PERFORMANCES USING ORACLE AND MONGODB” All content following this page was uploaded by Bilal Rashid on 04 June 2016.

[10]. Ali Salehnia “Comparisons of Relational Databases with Big Data: a Teaching Approach” South Dakota State University


[15]. Angela Bonifati Radu Cucanu Aurélien Lemay “Learning Path Queries on Graph Databases” c 2015, Copyright is with the authors. Published in Proc. 10.5441/002/edbt.2015.11

[16]. Alexandra Martinez1, Rodrigo Mora2, Daniel Alvarado3, Gustavo L.opez3, and Steve Quir_os2


