A Hybrid Database Approach Using Graph and Relational Database

H.R.Vyawahare
Department of CSE
Sipna COET
Amravati, India
harsha.vyawahare@gmail.com

Dr.P.P.Karde
Department of Information Technology
Government Polytechnic
Amravati, India
p_karde@rediffmail.com

Dr.V.M.Thakare
Department of MCA
SGBAU
Amravati, India
vilthakare@yahoo.co.in

***Abstract*—Database industry is recently facing a major challenge in gaining scalability and handling multiple databases in one application. Relational database is clearly a well known standard for storing, managing and querying information. It is used in variety of contexts including analytical querying and business intelligence. However, due to enormous growth in distributed databases and massive use of social networks, new requirements have emerged and graph-structured data is becoming more and more important. Graph database can store graph data more naturally. Relational databases and graph databases both have strengths and weaknesses depending on the characteristics of data and the types of queries to be evaluated. However benefits can be taken from mix of both by moderating some of the constraints. Thus propose a design of hybrid model such that the two models are integrated in one system to eliminate the limitations of individual systems. The hybrid system unifies the strengths of relational databases and graph databases by analyzing their strengths and weaknesses. This paper attempts to provide an overview of past investigation on the hybrid database model.**

Keywords—distributed database, graph database, hybrid database, relational database

I. INTRODUCTION

Relational databases have been the universal industry standard for almost as long as databases existed. It is undoubtedly useful for storing tabular data that fits in a per-defined schema which in fact poorly accommodates the interconnections within the data set. Thus forcing highly connected data set in relational database results in severe performance issues in query return time. SQL too becomes complex when handling large database. Larger the database more is the number of joins which ultimately leads to increase in query retrieval time.

With the recent rise of social networks and modern technological advancements, data is quickly becoming more and more connected and less suitable for relational databases. As a result the database industry started looking for other efficient alternatives. The Nosql movement has brought many new database models in database industry where each model had some important features that relational model doesn't have. Motivations for this database include schema agnostic, simplicity of design, use of commodity hardware, easy to add more servers (horizontal scalability), are highly distributable, high availability and are open source. Nosql databases are classified into key-value store, document store, column store, and graph store. Key-value databases show good performance when handling interim unstructured data; graph databases handle relationships as first-class citizens and column databases are better for storing historical data for business analysis. Even though relational databases have gained competition, they are still dominant database.

Deciding which database is more appropriate for a task at hand is not always trivial since different databases are designed for different task. Many times using Nosql database generates the problem of storing ideal relational data. Also polyglot persistence is a need of today. Taking into consideration, the need of today's customer, more and more companies are using multiple database systems where databases are chosen such that advantages are exploited by one db and the weaknesses are covered by another. The interest in graph databases is continuously growing because of its ability to analyze the data in non-relational format (e.g. social networking data). A graph database basically focuses on connections between data. But still there is enough room for relational databases. A relational model is worth for well-structured and tabular data. So we focus on attempting to bridge graph databases with relational databases to acquire the properties and advantages of both. Motivating factor is the current importance to use non-relational data. In designing hybrid model, the data retrieval query need to search intended data in both databases. Data insertion query runs data classifier algorithm to find appropriate database that correctly models the data. Therefore, the task of finding the features which describe the data lays the foundation for classification.

II. DATA MODELS

A. Relational Model

Introduced in 1970s by E. F. Codd, the relational model offers a very mathematically-adapt way of structuring, keeping, and using the data. Figure 1 shows relational model along with its few terminology.

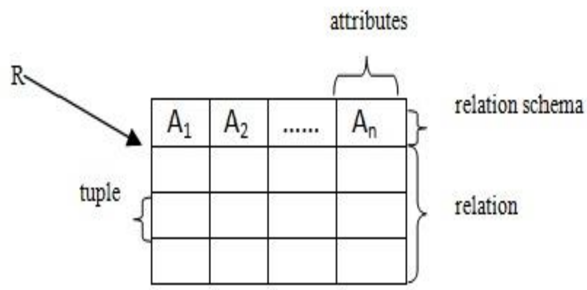

Fig. 1. The Relational Model

A relational database consists of a collection of tables, each of which has a unique name. A row (or tuple) in a table represents relationships among set of values, thus table is collection of such relationship. Relational model has got its name from the close correspondence between table and mathematical concept of relation Figure1 shows A_1, A_2,--,A_n which are column headers representing table attributes. A database schema is logical design of database whereas

978-1-5386-2599-6/18/$31.00 ©2018 IEEE

database instance is snapshot of data in database at a given instant in time.

A tuple must be distinguished from other tuples in terms of its attributes i.e. every tuple must be uniquely identified. This is achieved by defining keys. Further relationship constraints can be added by defining foreign keys. Additional other integrity constraints can also be specified. Also a query language is required for users to request information from database.

B. *Graph Model*

In mathematical terms, graph is composed of collection two elements: a node (also called vertex) and an edge. Each node represents an piece of information entity (like a person, place, thing, category or other piece of data), and each edge represents a connection or relationship between two nodes. Relationships are first-class citizens of the graph data model. It is any storage solution where connected elements are linked together without using an index. The neighbors of an entity are accessible by dereferencing a physical pointer. It is a database with Create, Read, Update, and Delete (CRUD) methods exposing a graph data model, such as property graphs (containing nodes and relationships), hypergraphs (a relationship can connect any number of nodes), RDF triples (subject-predicate- object).

Figure 2 shows structure of property graph model and Figure 3 shows an example. Property graph model is most widely used and we use the same in our proposed model. There are various popular graph database models available like neo4j, titan, hypergraphdb, flockdb, dex, infinite graph and so on. Depending upon user needs a particular database can be selected.

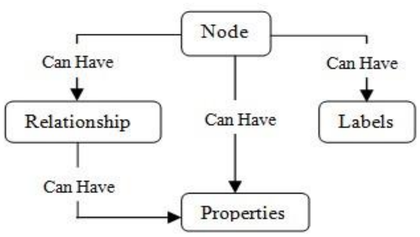

Fig. 2. Building blocks for property graph model

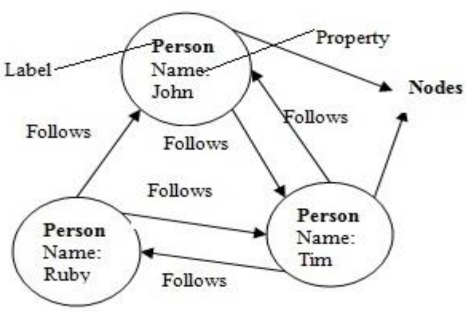

Fig. 3. A property graph model

C. *Hybrid Model*

One of the main advantages of graph databases over relational databases and other NoSQL stores is performance. Graph databases are typically thousands of times more powerful than conventional databases in terms of indexing, computing power, storage, and querying. In contrast to relational databases, where the query performance on data relations decreases as the dataset grows, the performance of graph databases remains relatively constant. So considering the pros and cons of both the models, today it makes pragmatic sense to use both databases. So the proposed focuses on building a hybrid database system for the storage and management of big data. Hybrid database approach or multi database system is defined as integrated data system composed of collections two or more autonomous datasets and/or databases. Figure 4 shows a hybrid approach. There are multiple issues that must be addressed like developer needs to learn multiple technologies, multiple query languages and many more. Literature review enlists work done by number of researchers in the field of hybrid databases using graph database and relational database.

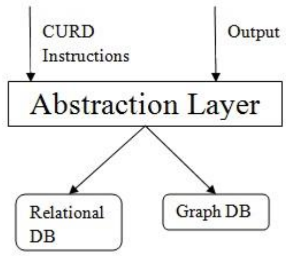

Fig. 4. A Hybrid Approach

III. LITERATURE REVIEW

Current research can be divided into three steps. Step1, the relational database side, step2 concentrating on relational database limitations and graph database emergence as alternative to relational db. The last step is hybrid merger of both. Several researchers enlists about the superiority, matured nature, market hold, popularity, robustness and many more success stories of relational databases. The relational database model first came to the force in the 1970s with Codd's Relational Model of Data [1]. A Millions of developers use a mature querying technology that is SQL. Johannes Zollmann wrote in his chapter about the ACID properties of the RDBMS. He also notes that ACID properties give strong guarantees on consistency [2].

The second step talks about its inability to stand in today's web 3.0 scenario. In spite of advances in computing, faster processors, and high-speed networks, the performance of relational database applications is becoming slower and slower. The striking constrains on relational db include lengthening of query times, numerous joins and self joins, rigid schema, structural limits, expensive to set up and maintain. Advances in the complexity of information cause another drawback to relational databases. This all is happening because of an overall evolution not only in the volume and velocity of data, but in its variety, complexity, and interconnectedness. Today's data can be characterized as densely connected, semi-structured, and with a high degree of data model volatility. And as the volume, velocity and variety of data are increasing, the data relationships are growing even faster. Relational databases were designed for tabular data, with a consistent structure and a fixed schema. Despite their name, relational databases do not store relationships between data elements; they are not well suited for today's highly connected data [3] [4].

Many researchers have suggested alternative for relational databases. Literature survey suggests graph db as appropriate alternative [3] [4]. Adrian Silvescu et al [5] introduce a concept of graph database. In [6] Jaroslav Pokorny has provided many important concepts of graph databases. He provided dual view focusing on bright as well as dark side of graph db. The major drawbacks focused include lack of maturity, functionality restrictions and big analytics requirements. Also it talks about the restrictions faced during designing, need for proper benchmarking, limitations in pattern matching queries and many more.

Thus considering the strengths and weakness of both database approaches researchers started ways to migrate their existing relational database into graph database. However that too posed challenges [7] [8]. So the optimal way was to find a solution that supported both databases. Also polyglot persistence is a need of today seeing towards the customer needs more and more companies use multiple database systems where databases are chosen such that advantages are exploited by one db and the weaknesses are covered by another.

Jeff Shute et al, have worked on F1 which is a hybrid database that combines high availability and scalability of NoSQL systems like Bigtable, and the consistency and usability of traditional SQL databases. F1 is built on Spanner [9].

Another Hybrid Database System approach is designed by Blessing E. James and P. O. Asagba for the storage and management of big data. The hybrid system is made up of MySQL database and MongoDB which are the most popular relational and NoSQL (non-relational) database servers. The author has loaded the data in all three modes i.e. SQL mode, MongoDB mode and hybrid mode. Their hybrid model is able to enhance storage and management of big data [10].

The approach taken by Christopher J. O. Little in his Grapht, provides an intermediate query processing layer between the traditional RDBMS and in-memory graph store. It describes a graph in relational environment and permits queries similar to SQL but with power of graph handler. The model design is a build in-memory store and works for hybrid query. When user hits query, query processor separates them into row-centric sub queries for relational database directly, and graph-centric sub queries for the graph handler. A hybrid query language gSQL is also presented [11].

John Roijackers has addressed another hybrid approach. According to author splitting the sql and nosql data leads to management of multiple databases. Instead we can create an abstraction layer over both databases through which it behaves as a single database. A virtual relation is created in sql database and is loaded by nosql data which is in advanced transformed into triple format. Also an extension to SQL query is created that includes a NoSQL query pattern [12].

Luis Ferreira tried to bridge the gap between SQL and NoSQL by building a layer between the SQL code and interpreter, and the actual database underneath it. This provides a way to run SQL queries on top of a NoSQL system at the cost of possible reduction on performance [13].

Another similar approach taken by [14] tried to bridge SQL and NoSQL with relational and Graph database. The work is implemented over FishBase(a relational database) which is a fish knowledge base used by research scientists, fishery managers, biologists and enthusiasts. To meet the current needs the relational FishBase was migrated into graphical FishBase. However it was not possible to update or add new data straightly into graph model. The implemented version was called as Regraph that maps data from a relational to a graph database, providing a hybrid architecture that bridges both databases, keeping them connected, synchronized and in their native representations.

As like [12] [13] [14] Rune Ettrup and Lisbeth Nielsen presented Bridge-DB which is distributed database concept targeting multiple data source. It supports all CRUD operations, however it uses it own query language BQL. It is connected to PostgreSQL and Neo4J. The multi database system created by Bridge-DB uses middle-ware layer between heterogeneous databases. For determining which database should be queried or whether the query should be enumerated to run on multiple databases, the author has implemented a cost-based optimizer that integrates dynamic and black box cost model. Finally the optimizer does the post processing of results to fulfill the query [15]. Table1 is showing results of executing set of queries on individual relational db, graph db and the optimizer.

Martin Grund and et al;, have proposed a new architecture of enterprise application specific database system that allows including semantic and graph data directly in the same storage engine combining the advantages of relational and graph data processing in a single in-memory database engine. In-Memory technology is used since it provides the required performance and flexibility to combine different storage types in a single storage engine without losing performance. The author uses HYRISE - a Main Memory Hybrid Storage Engine as architectural foundation which also uses compression and partitioning algorithms. The second aspect is execution of query which covers the integration of two different kinds of storage engines [16].

TABLE I. RESPONSE TIMES IN MS FOR MERGED DATASET

Query	Neo4j Cached	Neo4j Non cached	Postg-reSQL cached	Postg-reSQL Non cached	Optim-izer Single MQ	Optimi-zer Multiple MQ
1	19	1654	247	324	263	24
2	29	1557	213	287	244	36
3	26	1685	212	290	144	34
4	15	1908	701	1150	346	25
5	32	1668	474	681	289	41

IV. PROPOSED SYSTEM: THE HYBRID DATABASE APPROACH

A database design model which is able to work with multiple different databases like, traditional relation tables stored in rows, columns and graphical database, as being able to store graphs natively for storage and management of data mainly categorized as big data. This approach of integration of graph and relational data was ignored because both data types store data differently yielding different access patterns to disk which were not compatible thus the resultant system developed will have power of two systems and will overcome the limitations of the individual systems.

It aims to enhance storage and management of big data for easy retrieval without migration of database. The resultant system basically uses a classifier that performs data bifurcation while storing. The classifier checks for nature of data. Highly structured data is dumped into relational while structure less data goes to graph database. Figure 5 shows probable proposed approach. Thus instead of giving up the benefits of relational database systems for NoSQL database, proposed approach offers the benefits of both systems in a single database system.

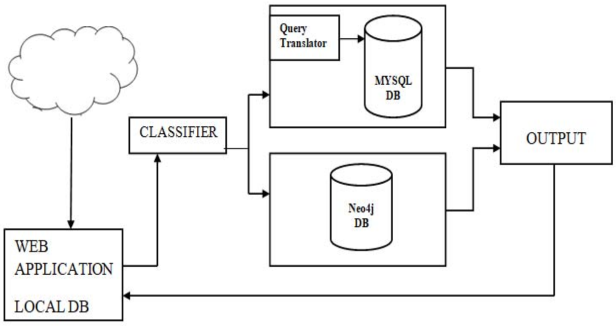

Fig. 5. Proposed Model (Functional Block Diagram)

Algorithm like B-tree or DFS works on graph dealing with its node and edges and proportional fill algorithm work for entities and tuples of relational database.

A. *Basic Algorithm*

Step 1: Load data
Step 2: Define class of data
Step 3: Initialize DB hybrid Interface
Step 4: Test Data: if
Data is structured then
Store in MySQL database
Data is unstructured then
Store in Neo4j database
Step 5: Update DB hybrid interface
Step 6: View, Delete, Update, Exit

V. Conclusion

Thus a method for implementation of hybrid database system between a client and two heterogeneous databases, which would be capable of decomposing and executing queries on multiple databases with the purpose of speeding up response times by allowing the databases to work in union, is proposed. The next phase would attempt to implement the said concept to obtain the results. In this paper, an extensive survey on various hybrid database approaches is done.

References

[1] E.F Codd, “A Relational Model of Data for. Large Shared Data Banks” Communications of the ACM, Vol. 13, No.6, pp. 377-387, June 1970 DOI=http://dx.doi.org/10.1145/362384.362685.

[2] Johannes Zollmann, “NoSQL Databases”, [Online] Available at: http://sewiki.iai.unibonn.de/_media/teaching/labs/xp/2012b/seminar/10-nosql.pdf, August 2012

[3] Chad Vicknair, Michael Macias,Zhendong Zhao, Xiaofei Nan, Yixin Chen and Dawn Wilkins, “A comparison of a graph database and a relational database: a data provenance perspective” , In Proceedings of the 48th Annual Southeast Regional Conference ACM SE,10, pp.42:1-42:6, April 2010, doi:10.1145/1900008.1900067.

[4] “Overcoming SQL strain and SQL pain”, Neo4j white paper, April 2015

[5] Adrian Silvescu, Doina Caragea and Anna Atramentov, “Graph Database”, Artificial Intelligence Research Laboratory, Department of Computer Science, Iowa State University, 2012, [Online] Available at: http://people.cs.ksu.edu/~dcaragea/papers/report. pdf.

[6] Jaroslav Pokorny, “Graph Databases: Their Power and Limitations”, In IFIP International Conference on Computer Information Systems and Industrial Management, Springer Cham, pp. 58-69, Sept 2015.

[7] Roberto De Virgilio, Antonio Maccioni and Riccardo Torlone, “Converting Relational to Graph Databases”, Proc. ACM First International Workshop on Graph Data Management Experience and Systems ,Article No.: 1 pp. 1-6 June 2013 doi:http://doi.acm.org/10.1145/2484425.2484426.

[8] Subhrajyoti Bordoloi and Bichitra Kalita, “Designing Graph Database Models from Existing Relational Databases”, International Journal of Computer Applications, Vol. 74 No.1, pp .25-31 July 2013.

[9] Jeff Shute et al, “F1: A Distributed SQL Database That Scales”, In Proc. VLDB Endowment, Vol. 6, No. 11, pp 1068–1079, August 2013.

[10] Blessing E. James and P.O.Asagba, “Hybrid database system for big data storage and management”, International Journal of Computer Science, Engineering and Applications(IJCSEA), Vol. 7, No. 3/4, August 2017.

[11] Christopher J. O. Little, “Grapht: A Hybrid Database System for Flexible Retrieval of Graph-structured Data”, Master’s Thesis, University of Cambridge, Emmanuel College, Cambridge, United Kingdom, June 2016.

[12] John Roijackers, “Bridging sql and nosql”, Master’s thesis, Eindhoven University of Technology, Department of Mathematics and Computer Science, pp.25-32, May 2012.

[13] Luis Ferreira, “Bridging the gap between SQL and NoSQL”, A state of art report, Universidade do Minho, pp 187-197, May 2011.

[14] Patricia Raia Nogueira Cavoto, “ReGraph: Bridging Relational and Graph Databases”, Master’s Dissertation, State University of Campinas, Institute of Computing, Campinas, SP., [Online] Available at: http://repositorio.unicamp.br/handle/REPOSIP/3047203, April 2017.

[15] Rune Ettrup and Lisbeth Nielsen, “Bridge-DB- Query Optimization in a Multi-Database System”, Master’s Thesis, Software Mater, Aalborg University, June 2015.

[16] Martin Grund, Philippe Cudre-Mauroux, Jens Krueger and Hasso Plattner, “Hybrid graph and relational query processing in main memory”, IEEE 29th International Conference on Data Engineering Workshops (ICDEW), pp 23,24, April 2013.

[17] Cory Nance, Travis Losser, Reenu Iype and Gary Harmon, “Nosql vs Rdbms - Why There is Room for both”, In Proceedings of SAIS Savannah, GA, USA, March 2013.

[18] Shalini Batra and Tyagi Charu, “Comparative analysis of relational and graph databases”, International Journal of Soft Computing and Engineering issn: 2231-2307, Vol. 2, No.2, pp 509-512, May 2012 .

[19] Neal Leavitt, “Will NoSQL Databases Live Up to Their Promise?”, IEEE Transactions on Computer, Vol.43, No. 2, pp.12-14, Feb. 2010 doi: 10.1109/MC.2010.58.

[20] Harsha R.Vyawahare and Dr Pravin P. Karde “An Overview on Graph Database Model”, International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE) ISSN (Online): 2320-9801, Vol. 3, No. 8, pp 7455-7457, August 2015.

[21] Angira Amit Patel and Dr. Jyotindra N. Dharwa, “An Integrated Hybrid Recommendation Model Using Graph Database”, Published in IEEE International Conference on ICT in Business Industry & Government (ICTBIG), April 2017.

[22] Nahla Aburawi, Alexei Lisitsa and Frans Coenen, “Querying Encrypted Graph Databases”, In Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018), pages 447-451 ISBN: 978-989-758-282-0, 2018.

[23] Jaroslav Pokorny, Michal Valenta and Jiri Kovacic, “Integrity Constraints in Graph Databases”, Procedia Computer Science, Vol. 109, pp 975-981, 2017.

[24] Marco Vogt, Alexander Stiemer and Heiko Schuldt, “Icarus: Towards a multistore database system”, In IEEE International conference on Big Data, Jan 2018.