Volume 4, Issue 5, May 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Review: Graph Databases

Pallavi Madan
M.Tech,Dept of CSE,
Graphic Era University, Dehradun, India

Anuj Saxena
CEO, Institute De Informatica
Dehradun, India

Abstract: This paper covers NOSQL databases. With the increase in net users and applications it is important to understand graph databases, the future of tomorrow's data storage and management. Basic architecture of some graph databases has been discussed to know their working and how data is stored in the form of nodes and relationships.

Keywords- NOSQL databases, Neo4j, JENA, DEX, InfoGrid, FlockDB.

I. INTRODUCTION

The use of relational database leads to problem because of deficits and problem in modeling of data, constraints over several servers and big amount of data. There were trends that brought the attention of software community to these problems:

1. The growth of the volume of data generated by users, systems and sensors further accelerated by big distributed systems like Amazon, Google and other cloud services.
2. The increasing complexity of data accelerated by the internet, social networks.

In order to overcome these and other more problems, NOSQL databases were introduced. NOSQL databases can be categorized on the basis of data model in to four categories:

1. Key value stores
2. Big table implementations
3. Document stores
4. Graph databases

Here, we will go with graph databases. Some applications of graph databases are: social graph, recommender system, software design, fraud detection.

II. METHODOLOGY

Graph is a set of vertices and edges where vertices denote some entity and edges represent relationship between these entities. Graphs can be directed or undirected. Undirected graphs can be traversed in both directions while directed graphs can be traversed only in one direction. Properties are key value pairs attached to nodes and edges. A vertex and an edge can have one or more property based on the entity and relationship respectively represented by them. Nodes or vertices can be visited following the edges based on the properties. All this forms a node space.

Graph databases are helpful in solving complex and dynamic relationships in highly connected data to generate advantageous results. They are generally used with transactional systems. In graph databases relationships can either be pre defined or created by the user. It is the best way to query connected data.

Data can be easily casted into the form of graphs with the help of graph database e.g. a person on a social networking site can be represented through a node having properties like name, age, hobbies, etc. and its relationships with other people like friends with can be represented through edges.

Queries can be applied using languages CYPHER or GREMLIN. For this, we have to give names to connections between nodes.

III. GRAPH DATABASES

There are many graph databases like Neo4j, AllegroGraph, Hypergraph DB, DEX, FlockDB, Infogrid, Open Graph, Knowledge Graph etc. Chad Vicknai, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen and Dawn Wilkins explained that security plays an important role in the selection of graph databases for any data system [6].

3.1 Neo4j

Neo4j is a full ACID-transactional database. Transactions in Neo4j are similar to traditional database transactions. D. Dominguez-Sal, P. Urbon-Bayes, A.Gimenez-Vano, S.Gomez-Villamor, N.Martinez-Bazan, and J.L. Larriba-Pey mentioned that Neo4j is one of the most popular alternatives of graph databases due to its dual free software/commercial license model (AGPL license) [2]. It is an open source. Neo4j is fully written in Java and can be deployed on multiple systems. It allows the users to develop fraud detection systems based on connected intelligence which provides enhanced degree of insights compared to algorithms that uses statistical analysis and pattern recognition.

Shalini Batra, Charu Tyagi mentioned that Neo4j 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[8].

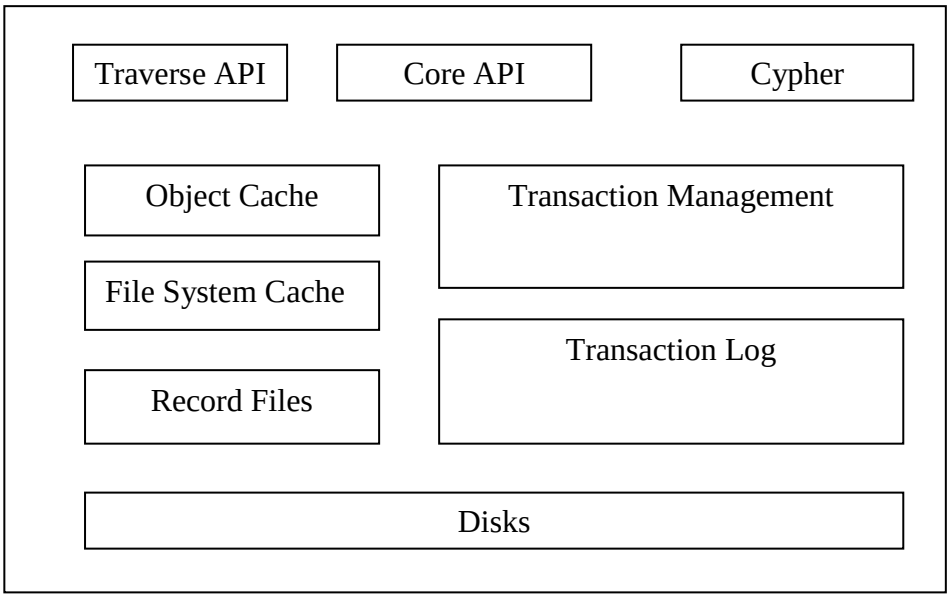

Fig 1: Neo4j Architecture

Neo4j stores graph data in different store files where each store file contains data for specific part of graph. The node store file stores node records, the physical file for which is neostore.nodestore.db. Similarly, relationships store in relationship store file neostore.relationship.db.

3.2 *Jena (RDF)*

Brian McBride explained that Jena is an API developed in the Java programming language, for the creation and manipulation of RDF graphs [9].It is an open source. It is a standard for describing relationships in the form of triplet: subject, object and predicate; where subject and object are the vertices and predicate is the edge combining these vertices. D. Dominguez-Sal, P. Urbon-Bayes, A.Gimenez-Vano, S.Gomez-Villamor, N.Martinez-Bazan, and J.L. Larriba-Pey describe the test to the TDB backend which stores the graph as a set of table and indexes: the node table and the triple indexes [2]. The node table stores the map of nodes in the graph to the node identifiers and vice versa. The triple indexes describe the structure of the graph.

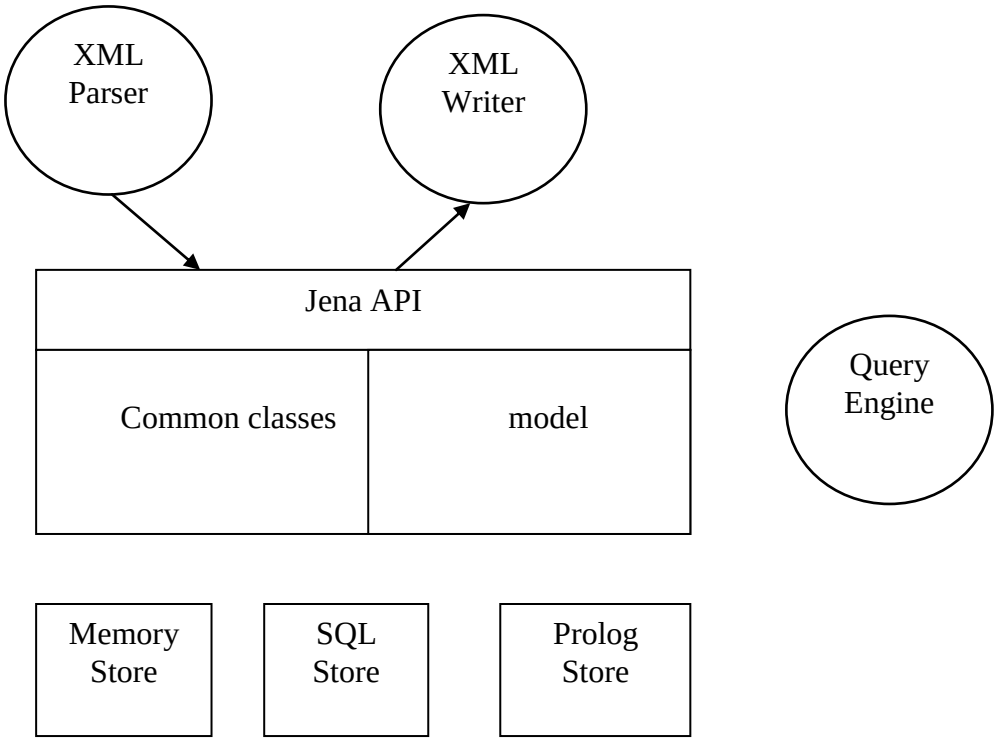

Fig 2: JENA Architecture

The Jena API itself consists of a collection of Java interfaces implemented by common set of classes; representing resources, properties, literals, containers, statements and models. The model class is a generic implementation of an RDF graph. An interface connects model to common classes that implement storage and querying of RDF statements.

3.3 DEX

DEX is based on bitmap representation of entities. All node and edges are encoded as collection of objects. It is able to support attributes for either vertices or edges. Two main types of structures are implemented: bitmaps and maps. DEX encodes the adjacency list of each node into bitmaps. Maps are traversed depending upon the query.

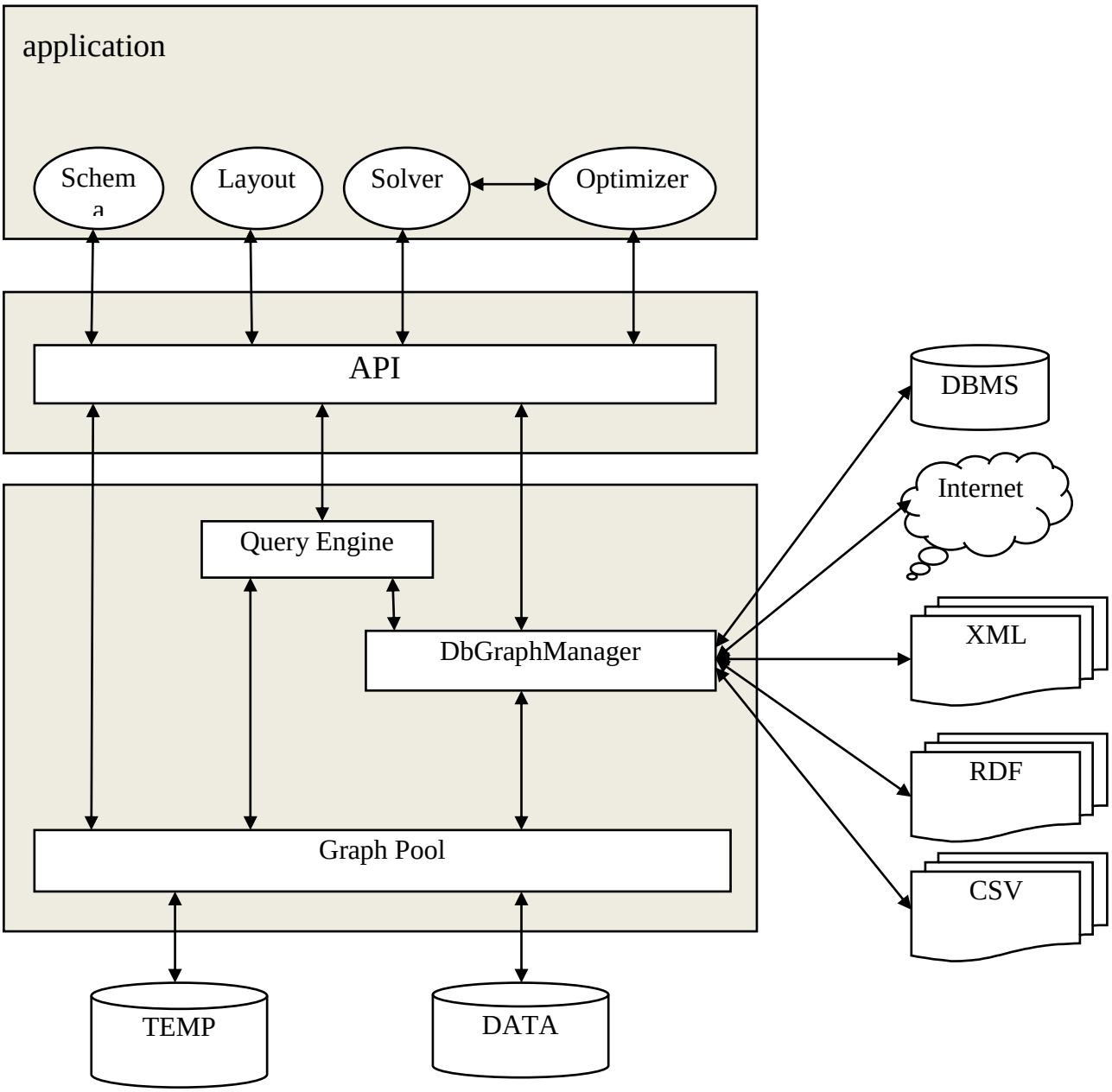

Fig 3: DEX Architecture

DEX contains several modules:

1. Schema module- provides the maintenance of the database graph.
2. Solver module- implements the queries.
3. Optimizer module- improves the access plans generated.
4. Layout module- display relational graphs.

API provides an interface to basic operations and functions like shutdown.

Core provides data management and query resolution. It consists of 3 modules:

1. Graph Pool- manages graphs and their data structures and provides them a persistent repository.
2. DbGraph Manager- provides the interface between external data modules and DbGraph.
3. Query Engine- provides methods for query optimization.

3.4 Infogrid

Nikolaos Giannadakis, Anthony Rowe, Moustafa Ghanem, Yi-ke Guo give introduction about infogrid that it is a data integration middleware engine, designed to operate under a Grid framework[7]. Grid refers to a global computing platform where users can connect to the services provided and access the data resources.

InfoGrid servers are connected to the middleware. This provides an interface to the wrappers. Resource wrappers are registered with the Wrapper Directory. The wrappers may wrap anything from executables to HTTP servers. Wrapper also provides the reference to document viewer directory which contains visualization services registered with middleware. The data returned by wrapper can be seen in this directory.

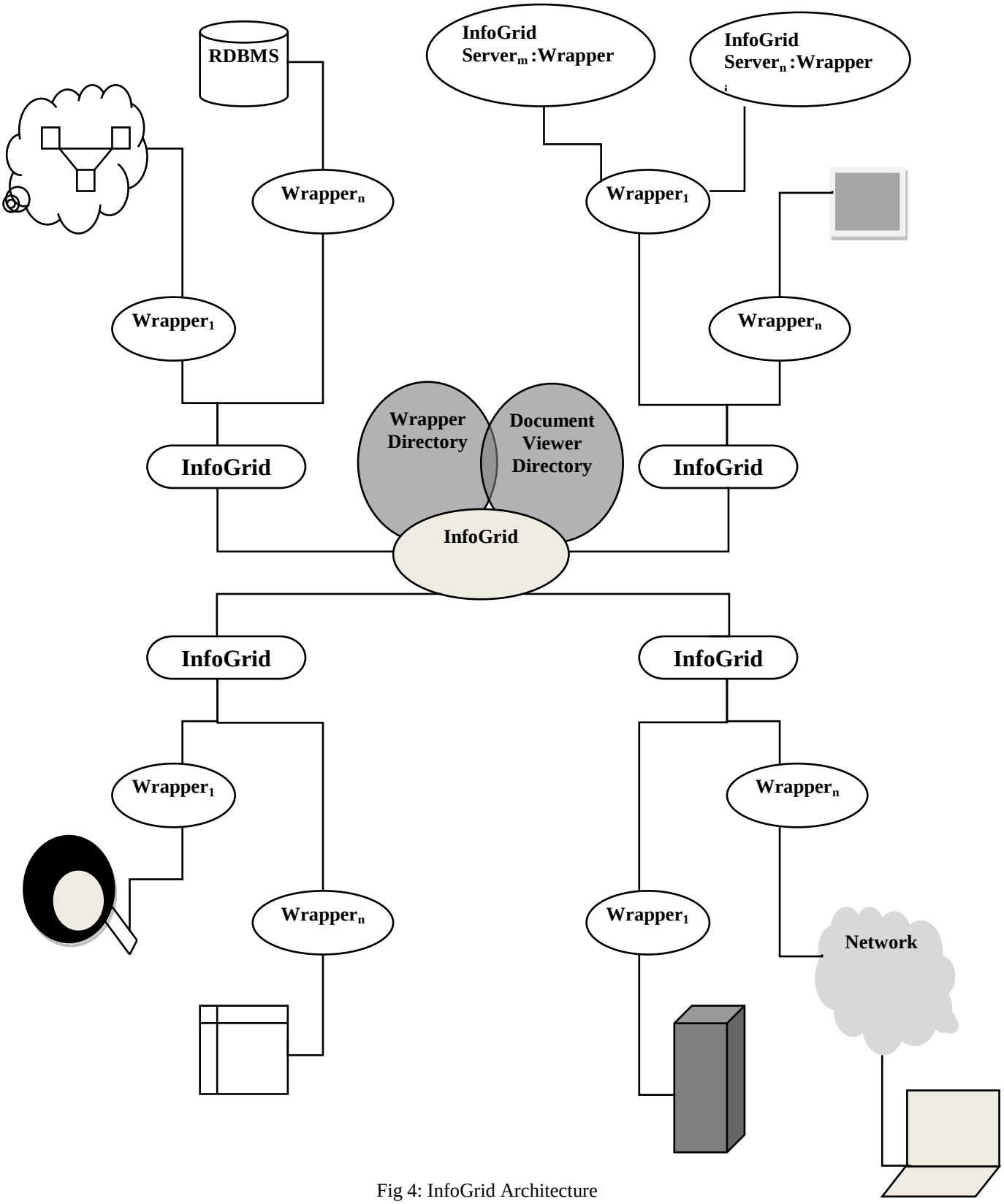

Fig 4: InfoGrid Architecture

3.5 FlockDB

Silvan Weber gives introduction of Twitter's FlockDB [10]. It is interesting because it is easy to understand that a vertex belongs to a person and an edge either has the property "following" or "followed". FlockDB uses MySQL as the basic database storage system. NoSQL databases sometimes use SQL database systems as their basic database system, and therefore the term not only SQL is justified.

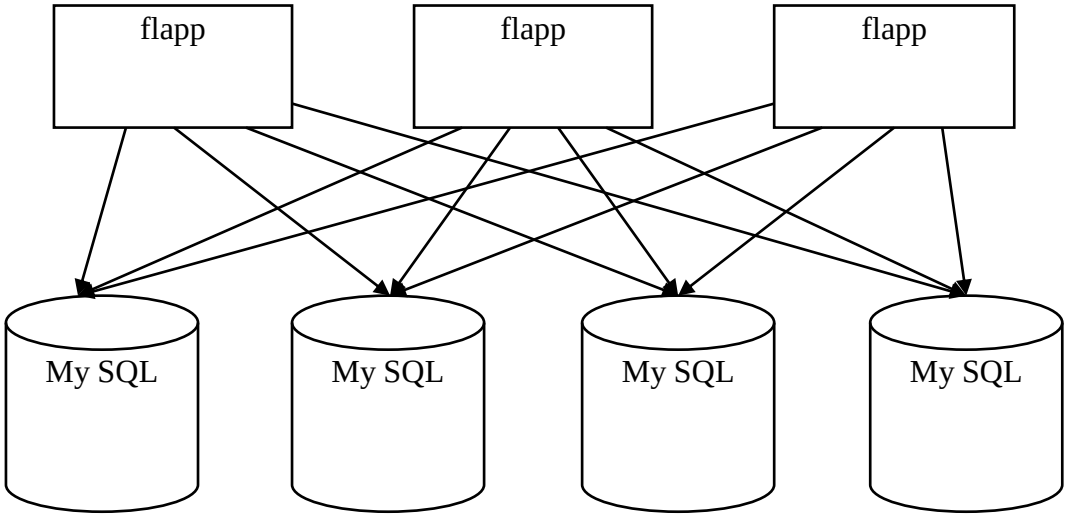

Fig 5: FlockDB Architecture

In flapps (FlockDB application servers), each request is independent from all other requests. FlockDB is not made for traversals. It is rather built for very quick queries.

IV. CONCLUSION

Graph databases have brought a new trend of modeling the data and solving problems. They perform better than relational database. Many big organizations like Google, twitter, facebook are using graph databases to handle their large amount of data and queries. The selection of the graph database should be on the basis of the data. In this paper, introduction to graphs is given to understand graph databases easily. The architectures of five graph databases have been discussed for the understanding of their internal functioning.

REFERENCES

[1] Justin J.Miller, *"Graph Database Applications and Concepts with Neo4j",* Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA March 23rd-24th, 2013.

[2] D. Dominguez-Sal, P. Urb´on-Bayes, A. Gim´enez-Van´o, S. G´omez-Villamor, N. Mart´ınez-Baz´an, and J.L. Larriba-Pey "*Survey of Graph Database Performance on the HPC Scalable Graph Analysis Benchmark",* H.T. Shen et al. (Eds.): WAIM 2010 Workshops, LNCS 6185, pp. 37–48, 2010. Copyright Springer-Verlag Berlin Heidelberg 2010.

[3] Deepayan Chakrabarti and Christos Faloutsos, *"Graph Mining: Laws, Generators, and Algorithms"*, ACM Computing Surveys, Vol. 38, March 2006.

[4] Ming-Syan Chen, Senior Member, IEEE, Jiawei Han, Senior Member, IEEE and Philip S. Yu , Fellow, IEEE, "*Data Mining: an Overview from a Database Perspective"*, IEEE Transactions on knowledge and data engineering, vol. 8, no. 6, December 1996.

[5] Daniela Florescu, "*The Neo Database – A Technology Introduction (20061123)"*, Oracle, in the "ACM Queue", Vol. 3, No.8, 2005, Copyright 2006 Neo Database AB (info@neodatabase.net).

[6] Chad Vicknai, Michael Macias, Zhendong Zhao, Xiaofei Nan, Yixin Chen and Dawn Wilkins, "*A Comparison of a Graph Database and a Relational Database",* ACMSE '10, April 15-17, 2010, Oxford, MS, USA.

[7] Nikolaos Giannadakis, Anthony Rowe, Moustafa Ghanem, Yi-ke Guo, *"InfoGrid: providing information Integration for knowledge discovery",* Information Sciences 155 (2003) 199–226 www.elsevier.com.

[8] Shalini Batra, Charu Tyagi, *"Comparative Analysis of Relational And Graph Databases",* International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-2, May 2012.

[9] Brian McBride, *"Jena: Implementing the RDF Model and Syntax Specification",* brian_mcbride@hp.com.

[10] Silvan Weber, *"NoSQL Databases",* silvan.weber@msc.htwchur.ch.

[11] "B*asic Graph Properties",* Algorithms Lecture 11, http://creativecommons.org/licenses/by-nc-sa/3.0/, Copyright 2009 Jeff Erickson.

[12] Norbert Martínez-Bazan, Victor Muntés-Mulero, Sergio Gómez-Villamor, Jordi Nin, Mario-A. Sánchez-Martínez, Josep-L. Larriba-Pey, *"DEX: High-Performance Exploration on Large Graphs for Information Retrieval", CIKM'07,* November 6–8, 2007, Lisboa, Portugal. Copyright 2007 ACM 978-1-59593-803-9/07/0011.

[13] Anton Dries Siegfried Nijssen , "*Analyzing Graph Databases by Aggregate Queries",* MLG'10, July 24-25, 2010, Washington DC, U.S.A. Copyright 2010 ACM I978-1-4503-0214-2/10/07.

[14] Bin Shao, Haixun Wang, Yanghua Xiao*, "Managing and Mining Large Graphs: Systems and Implementations",* SIGMOD '12, May 20–24, 2012, Scottsdale, Arizona, USA. Copyright 2012 ACM 978-1-4503-1247-9/12/05.

[15] Chen-Chung Chi, Chin-Hwa Kuo, and Wen-Jui Yu*, "Using Web 2.0 To Design An English Article Recommendation System",* International Conference on Educational Research and Sports Education (ERSE 2013).

[16] Robin Hecht, Stefan Jablonski*, "NoSQL Evaluation",* 2011 International Conference on Cloud and Service Computing 978-1-4577-1637-9/11.

[17] Ciro Cattuto, Andre Panisson, Marco Quaggiotto, Alex Averbuch*, "Time-varying Social Networks in a Graph Database",* Proc. of the GRADES2013 workshop on Graph Data-management Experiences and Systems at SIGMOD2013.

