Summary Survey, Fall 2021

On the Topic of Storage and Visualization Options for Unstructured Data Warehousing

An exploration of RDBMS based in SQL and NoSQL based DWs reveals that a lot of work has already been done in regards to identifying differences between the two methods of implementing data. Primarily, several articles have worked towards finding a way to transition between the two platforms.

Venkatraman, S., Fahd, K., Kaspi, S., & Venkatraman, R. (2016). SQL versus NoSQL movement with big data analytics. *International Journal of Information Technology and Computer Science*, *8*(12), 59-66.

Summary:

The article focuses on the two major impacted fields in data management, namely analytics in Big Data and NoSQL databases respectively. It explores the contexts and components of both and aims to demonstrate the strengths of NoSQL while highlighting its specialization and coexistence with standard SQL databases. Main points revolve around the shortcomings of standard SQL, particularly when it comes to large database size and easily accessible analytics in the business sphere, whether it be representation or visualization. Big emphasis placed on flexibility as well as horizontal scaling, with many alternatives available to address limitations and possibilities. Particulars include the needs of Big Data (high velocity, data complexity, volume, variety) and how through key-value storage, scaling, and modeling, whether through graph or document representation.

Critique:

We focus mainly on the contrasts in this article to highlight the differences between what we’ve done/know and what we will be doing, as well as to single out particular aspects of NoSQL that seem promising and could provide alternatives. Luckily for us, this article is all about alternatives, and more importantly all about feasible alternatives. While a graph relies on graph theory and the translatability of the topic, document, wide-column, and key-value models all vary and can be considered for topic modeling. And while wide-column is the recommendation the article makes for solutions in big data that may be useful to us. It also points out that several of the weaknesses and shortcomings of NoSQL lie not in the technology itself, but the novelty of the technology and the limited expertise that would come with it. On top of that, due to the lack of a standard that is agreed upon across all fields, there is no real complication that would hinder our own progress and we can use this as a frame of reference.

Bicevska, Z., & Oditis, I. (2017). Towards NoSQL-based data warehouse solutions. *Procedia Computer Science*, *104*, 104-111.

Summary:

The paper discusses the many possibilities and delves into the NoSQL approach for data warehouse, both in terms of usage and application. It aims to strike a balance between the old and the new, as NoSQL data marts arguably cannot directly translate over all of the meaning and functionalities that lie within the original SQL mart. Many of these steps have already been explored in our own Data Warehouse course, in particular the denormalization of data structures, maintenance of hierarchies, and measuring of the data, whether it be through OLAP or not. It also places a good amount of emphasis on the reporting and representation aspect and how data can be displayed that doesn’t entirely rely on graphics.

Critique:

While we’ve visited this article prior, this time we will be looking at if solely from the angle of NoSQL feasibility, as opposed to prior when we were only seeking to create a visual or graphical representation that would work with our standard SQL procedures. And while this article does not entirely address the ability to “transition” from an established standard SQL data mart to an equivalent NoSQL version and at the same time utilize all of its abilities to display or represent the data via NoSQL’s unique features, it does provide a good feasibility check on our intended product and confirms the process is certainly doable. We must, however, determine the scale in the end.

Chung, W. C., Lin, H. P., Chen, S. C., Jiang, M. F., & Chung, Y. C. (2014). JackHare: a framework for SQL to NoSQL translation using MapReduce. *Automated Software Engineering*, *21*(4), 489-508.

Summary:

Chung et al discusses Hadoop and MapReduce, more specifically the idea of developing SQL/SQL-like queries and translation to Apache HBase, which they use as the software for NoSQL. Their own framework of JackHare is meant to develop a “systematic” means of conversion, which in this case is the carrying over of SQL via GUI and ease-of-access systems that will help “ease the learning curve”. Primarily, this allows for our data model to conform to the style that matches our necessary input data without needing to make major adjustments, and the best part is all the queries will be able to match and run without extra hassle.

Critique:

Again another article that we visited in the past, with a twist. This time, we want to focus more on the aspects of translating and scaling to a NoSQL database, particularly with large-scale enterprise data that is very common in big business like the Amazon example we will likely be using. We plan on using MongoDB/Neo4j but can take a great deal of inspiration from the equivalence process and visualize more clearly the separate steps we will be undertaking (user submit SQL query, compiler scans, lookup equivalent in HBase, generate MapReduce, access HBase and execute MapReduce, wrap results, display results).

Vathy-Fogarassy, Á., & Hugyák, T. (2017). Uniform data access platform for SQL and NoSQL database systems. *Information Systems*, *69*, 93-105.

Summary:

The article takes a look at the integration of data and database systems along with the usage of NoSQL interpretation among standard SQL mechanisms. The solution listed does not involve queries such as aggregations or joins (a major aspect of data warehousing), but instead focuses on semantic, structural, and syntactical heterogeneous aspects of the source data systems. The authors also propose a web-based solution and application meant to be deployable and return the results in a normal NoSQL JSON format. While the solution mostly deals with the integration of data, it mostly does so by shielding specific details and methods of acquisition. It also tests upon querying and runtime via both MySQL and MongoDB databases, which performs primarily dependent on the size and transfer times of the data itself.

Critique:

The system’s novelty lies in the means by which two different and contrasting systems are combined together. The usage of the HybridDB “web client” is a very common means of cloud-based remote server hosting and can be linked to our usage of A3 or Azure, which could be translated over. Our issue in making the connection, or linkage, from the different schema and the actual source database are what cause issue, and with this methodology we could try to establish a similar kind of pattern or process to match up with what’s listed in the article. Since it can also handle both the MySQL and MongoDB DBMSs, we can also use this to function as a means for us to utilize a similar method in our own project.

Bugiotti, F., Cabibbo, L., Atzeni, P., & Torlone, R. (2014, October). Database design for NoSQL systems. In *International Conference on Conceptual Modeling* (pp. 223-231). Springer, Cham.

Summary:

For NoSQL, one of the major aspects of our decision-making process is the design and formatting of the system. Because of all the choices and “heterogeneity” of the options of NoSQL (a part of what makes it so versatile), a particular methodology must be decided upon, and this article proposes a means of modeling which can be replicated well. With emphasis on aggregation (unlike the above article), conceptual modeling, high-end NoSQL DB design and structure, and above all implementation, Bugiotti et al propose a system of identifying aggregates not unlike the standards and conventions of data warehousing.

Critique:

While the specifics of the proposal lie in the process of conversion from SQL to NoSQL, the results we will obtain and the end-product are reliant on the procedural events that lead up to it. To put it simply, the abstract simplified model and the standards highlighted should be adhered to on both sides, which will make it easier to simply insert into when converting. This is somewhat more complicated by the fact that despite the common design scheme, it is still undetermined just what exactly the final form should take, despite the article’s claims of narrowing down the decision-making process. Some more research in this area will be needed, but we could try the means by which we crop out some “unneeded” info in order to represent the whole thing more smoothly.

**A more extensive subject-oriented dive, taken and modified from Sem 1:**

Subject: Topic Storage and Visualization

Exploration results included word cloud, topic size points, web vertex points, bar graph, serialization

In particular, Graph DB alternative to Relational for ease of visuals and user access

Title: Graph Data Warehouse

Authors: Yunkai Liu, Theresa Vitolo

Paper Contents (Significance): When looking at ways of storing and visualizing the unstructured data in our data mart, GraphDB through GDBMS like Neo4j lacked the ability to easily construct linked queries as a result of user necessity. It proposes a system comprised of the combination of standard SQL operations in noSQL and allowing these statements to be executed. With a JAVA-based API, the article demonstrates the possibility for effective combination of the two in the form of GVIEW, a copy of graph data in the local library that allows for quick and relatively painless access to data and stores previous graph searches.

Opinions/Applications/Future Direction: For storage and visualization, noSQL and GraphDB options are always something that should be considered. Whether as alternatives or supplements, these fields can certainly provide a lot of utility and flexibility to database processes, and this article, while short, is very clear in what it means to accomplish: an alternative to siding with one method or the other and instead using the strengths of both. It’s a potential direction to go into if not sticking exclusively to previous works.

Title: Deep Data Warehouse

Authors: Christoph Gröger, Holger Schwarz, Bernhard Mitschang

Paper Contents (Significance): Proposes a link-based integration of warehouse data coupled with unstructured data and gathered contents. Holistic analysis requires the combination of structured warehouse data and unstructured information to make insights in the rapidly-changing modern world. The Deep DWH (in name only) based around enrichment of unstructured content and links in graph structure. Graph representation of warehouse items and content elements (aka dimensions and hierarchies) are reminiscent of the standard Neo4j type of node and edge architecture. The distinguishing feature lies in the property schema for each of the links, where the information for each node also relates to the information that is present in the edges.

Text

Description automatically generated

Opinions/Applications/Future Direction: This article was a bit of an error on my part as well as a bit of misdirection on the authors’ parts. Based on the title I believed Deep DW was to be the concept, and not nomenclature, of the findings they had presented. Instead, the “Deep” aspect of the article only came in its naming scheme. The rest of the material is very straightforward and similar to other graph database systems like the mostly used Neo4j or alternatives like Amazon Neptune/OracleGraph. It’s interesting to note that while there are still few widely used GraphDB programs aside from the aforementioned Neo4j, the fundamental nature of Graph Databases only differs in execution and representation, and not the essential basics.

Graph-based Data Integration and Business Intelligence with BIIIG, Andre Petermann et al; focus on usage in business and display

Chart, radar chart

Description automatically generated

TOWARDS NoSQL, Zane Bicevska, Ivo Oditis

Title: Implementing multidimensional data warehouses into NoSQL

Authors: Chevalier, M., El Malki, M., Kopliku, A., Teste, O., & Tournier, R.

Paper Contents (Significance): Discusses SQL and noSQL implementations of OLAP technology, and explores the usage of noSQL and instantiating aggregation lattice. The fact table for noSQL has to be “equivalent” and store in column-oriented databases, with hierarchies being represented in the vertical partitioning of the columns.

Diagram

Description automatically generated

The fact table representation:

Diagram, engineering drawing

Description automatically generated

Graphical user interface

Description automatically generated

Column-oriented database. Tables are defined row by row, but physical storage in columns so a “vertical partitioning” by column. This represents the entries of columns as dimensions.

The combined representation:

Diagram

Description automatically generated

While it’s not what we want, there’s also a document, text-based representation thru noSQL that looks a lot like the Neo4j one. Shown below for reference and comparison

Diagram

Description automatically generated

ARTICLE Title: A Framework for Building OLAP Cubes on Graphs

Author: Ghrab et al

Paper Contents (Significance): Proposes a novel framework for building OLAP cubes through graph data. Because current warehousing and graph frameworks not equipped to handle multidimensional modeling, this model will address these issues and contain designs meant to support extraction and design of candidate multidimensional spaces in property graphs. This basically means that for several nodes that all have individual attributes describing it, a merger can be made between these separate nodes (aka aggregation) that displays OLAP operations easily.