Carl Archemètre

CS 300-ON

Assignment 1

Non-Relational Data Storage and Retrieval Systems

Although Relational Database Management System (RDMS) is the dominant data-processing software in use, relational databases don't solve every problem involved in data-processing. Other database models have been implemented to address some of these issues. One such model is NoSQL (originally referring to "non-SQL", "not only SQL" or "no-relational"). This modeling do away with using relations to represent data, they are non-tabular and these databases represent their data in many ways that are different than relational tables. There isn't a single modeling for this database model but rather this modeling composes of a variety of modeling techniques that fall under the NoSQL database types. The different types of modeling includes document, key-value, wide-column and graph.

In document-oriented databases, the data is stored in documents that contains pair of fields and values. Key-value databases contains keys and values similar to a key-value pair of a map, it's a collection of records that are retrieved using a key that uniquely identify each record of the database. Wide-column databases stores its data in tables, rows similar to the relational model but its columns are dynamic. The last of the NoSQL database are graph databases that stores its data in nodes and edges. The nodes contain the data while the edges stores information about the relationships between the data. All of these modeling helps with large amounts of data and as well as a large number of end-users. Broadly speaking NoSQL refers to any non-relational databases. Since file-based system have mostly disappear from use, if you're not using relational database, the only other option is NoSQL.

As databases got larger, the prices increased dramatically, so one of the primary goals of NoSQL is to lessen the impact of that trend. One of the many benefits of NoSQL is that it allows software developers to store unstructured data allowing them a lot more flexibility. A large part of database design is the conceptual database design which requires the development of schemas to model the data and defining such schemas in advance can be difficult because data comes in a variety of ways such as structed, semi-structure and polymorphic. NoSQL's ability to stored large amounts of unstructured data helps in this problem. Schemas in the relation model are rigid when

compared to how SQL models its data. Some of the major advantages of NoSQL databases include flexible schemas, horizontal scaling, fast queries and ease of use for developers.

Oracle NoSQL Database (ONBD) is a NoSQL key-value database developed by Oracle. This NoSQL modeling is similar to a relational table but with additional properties. These additional properties include provisions to write units, read units and storage capacity. It is dynamic in that the database driver partitions the data in real time and distributes it across the storage nodes, it routinely read and write operations to the appropriate storage to optimize load performance and load distribution ("Oracle NoSQL Database," 2021). Oracle describes this database as a shared (shared-nothing) system meaning the data is shred uniformly across multiple shards in a cluster. In each shard, the storage nodes are replicated in order to ensure availability and rapid failover if a node fails. Oracle NoSQL Database provides drivers for procedural 3rd generation languages like Python, Node.js, Java, C , C# ("Oracle NoSQL Database," 2022).

Azure Cosmos BD is a NoSQL database that is schemas-agnostic, horizontally scalable modeled by storing its items in containers. The containers are grouped in the databases, these containers are schema-agnostic meaning no schema is imposed when adding data. Items can be thought of as tuples being inserted into a relation. By default, every attribute is indexed automatically proving great execution when it comes to queries. ("Azure Cosmos DB resource model," 2022).

Google Cloud Bigtable is another NoSQL database that is a compressed, data storage system built on Google technologies. This database is of type wide-column store that is implemented by mapping two string values (a row key and column key) as well as a timestamp into an associative array. For example, a webpage can be inserted into Bigtable database which the row key containing the URL, having columns describing the various properties of the webpage with one column containing the webpage itself and several timestamped versions describing copies of the web timestamped at the time they were accessed. It is designed to scale across thousands of machines and many more machines can be added without any need for reconfiguration ("Bigtable," 2022).

Although NoSQL answered some of the problems that relational modeling couldn't solve, it still has some disadvantages of its own. Some of these disadvantages include a lack of standardization similar to the one provided by the SQL standard. There are no standards that defines NoSQL, subsequently the design and the query languages vary between the NoSQL products. NoSQL databases prioritize scalability and performance over data consistency.

There are no duplicates in relational modeling but in NoSQL the same set of data can be entered in the same database. (“Advantages and Disadvantages of NoSQL Databases”).

A graph database ues the mathematical definition of graph to model its data as well as a graph data structure to implement the database. The graph data structures are used for queries with nodes, edges and properties used to store the data. Edges between the nodes containing data are used to model relationships between the data. Those individual data representation are intrinsically liked through the edges between them. That direct link allows data queries to be retrieved with minimal operation. No relations need joining because the relationships are stored permanently through the linkage by the edges. These types of databases are similar to network model of database that was modeled as general graphs. Some efforts were initially taken to standardize this modeling leading to query languages Gremlin, SPARQL and Cypher. One of the many problems that graph database was design to solve included the relational model’s inability to be malleable, so a software designer has to predict future use of the database to increase its longevity. A graph database model doesn’t require the software designer to plan details for the database’s future use. A graph database lends itself to easily model data that are themselves graphs such as family tree, hierarchical data like evolutionary trees etc.… Complex modeling of relationships aren’t easily model with the relational model so one of the reason for designing database graph databases was to make it easier to model complex data and make queries on such complex modeling easier to implement.

There are two main types of graph databases include property graphs and RDF graphs. The property graph databases are focuses on the modeling of relationship among the data and the query and analytics are based on the relationships amongst the data. The vertices of the graph contains the data and edges signify the relationships amongst the data as describe previously and both the vertices and edges have properties that are associated with them. The other main type of graph databases is the RDF graph which stands for Resource Description framework. It was designed to implement a set of W3C (Worldwide Web Consortium) standards. A statement in this model is represented as an RDF triple in which two vertices connected by an edge represents the subject, predicate and object. A URI, or Unique Resource Identifier associates every vertex and edge of the graph. This type of database modeling is very popular amongst healthcare organizations, pharmaceutical companies and government agencies (“Graph Database Defined”). Graph database modeling is a powerful tool in analytics. Traversals perform on the data provides different information about the data revealing relationships on massive amount s of data. What’s very

interesting about this type of modeling is that the structure of the data provides a lot of information about the data itself. Metadata is intrinsically built in its data structure used to implement this database model. For example, the different paths, path length between vertices and the clustering of vertices provide important insight and information about the data. This type of modeling also lends itself naturally to machine learning and AI due to how its metadata can be used to extract more information about the data.

AllegroGraph is an RDF graph database currently being used both commercially and by the US government, for example its being used to store data for Twitter's TwitLogic project. It was developed to meet the WC3 standards and uses SPARQL as its query language. The host languages for this database management system includes Java, Phyton, Ruby, Perl, C#, Closure, and Common Lisp. AllegroGraph is available for the three most popular operating systems Mac OS X, Windows and Linux. It is a horizontally scalable technology enabling its customers to extract sophisticated decision insights as well as predictive analytics from the data being modeled ("Innovation & Technology"). This database more specifically implements OWL, a semantic extension that allows SPARQL queries to sensitive to OWL ontologies. It also implements document technologies such as JSON, JSON-LD. The combination of these two implementations allows the database to process data with conceptual and contextual intelligence ("Innovation & Technology").

Amazon Neptune is a managed graph database management system developed by Amazon as part of the Amazon Web Services (AWS). It is optimized to store billions of relationships and for querying those relationships with minimal latency ("What is Amazon Neptune?", 2020). Its query language includes Apache TinkerPop Gremlin and SPARQL allowing its users to construct queries to traverse highly connected data, this significantly reduces the complexity of your code and less time is required to build applications that process those intricate relationships. The scope of the many uses for this database is very large. This includes drug discovery, network security, knowledge graphs, fraud detection. Amazon Neptune is also self-managed, so clients doesn't have to worry about common database management task such as software patching, hardware provisioning, backups, configuration, or setup. It is also fault-tolerant and self-healing, meaning when disk fails, they are repaired in the background without the losing the ability to use the database while it is being repaired, it also detects crashes and restarts automatically without the need for recovery or rebuilding cache and if this process fails, it'll fails over to one of its read replicas automatically ("What is Amazon Neptune?", 2020).

Graph databases advantages includes its ability to be malleable to frequent schema changes, managing large amounts of data, real-time query response time. When it comes to the language, there are no hidden assumptions that are found in SQL, meaning the queries written are clear and explicit. Each vertex contains its adjacent vertices information only, so this type of database performance is consistent even as the graph/database size grows extremely large. Graphs solve problems that can be both practical and impractical to the relational database modeling, so in other words this type of modeling has the ability to answer queries that are not possible from the relational modeling of data. The type of data modeling can produce metadata that is beyond what relational modeling can provide.

Researchers have proven that some of the query languages for graph database management systems to be Turing complete. With some of these query languages being Turing complete, any kind of high complexity algorithm can be written on them. Relational model can efficiently adapt to the ever-evolving nature of data once its schema is set in place, with graph database clients can continually add and drop vertex and or edges to shrink or expend the modeling. The graph data structure that used to implement a graph database is a recursive data structure so naturally it lends itself to make recursive queries easy to handle. These types of queries are extremely difficult to do in relational databases because there are no fixed number of joins ("What are the Major Advantages of Using a Graph Database?", 2019). Graph data are also more flexible to grouping and aggregating data. The tabular representation of data in the relational model greatly restricts how its data can be aggregated and grouped.

What make complex query of graph database possible is the use of special algorithms that are intrinsically defined on its data structure to accomplish their essential functions which in return simplify and speeding up the complex queries of the data. Two of those special algorithms are the traversals that can be performed on the data. The depth-first search traversal algorithm searches the data in the adjacent node to a data node in question, while the breadth-first traversal algorithm moves to different layers of the data being modeled in the graph. These traversals make it extremely efficient to find patterns in the data. The previously mentioned algorithm also makes it possible to find data that directly and indirectly linked. For example, data that are linked are adjacent to one another meaning that there is an edge between them, while indirectly linked data are linked thought their adjacent data (their neighbors) there isn't an edge between these two sets of data, but they are linked through the connections that its adjacent data forms with other data. Other algorithms intrinsically define by the data structure itself are path length

between the vertices containing data and the shortest path between two vertices containing data. Hotspots of data, meaning data that are highly connected with a lot of edges conveys a lot of information on the data set ("Graph database explained", 2019). For example, if a vertex represents a branch office, a particular branch having a lot of edges could correspond to it having a lot of staff members. Querying these relationships are extremely efficient and fast since they don't need to be calculated or require any types of joins since the relationships are stored within the data itself.

Another important advantage of graph database is the ability to model multiple dimensions. Time series, geo-dimensions, demographic can be combined to manage big data of multiple dimensions. This ability provides data scientist and business analyst to perform any analytical query on big data.

We've established the many advantages of the graph database model, but it too contains some inefficiencies. There isn't a standardized language for querying data in a graph database. The language being used depends on the on the platform of the database. Graphs are excellent for complex queries but are not appropriate for transactional-based systems. The relational modeling data is ubiquitous, so the usership of other types of database models is small making it harder to find technical support when a client runs into problems ("What is a Graph Database*?*", 2021). As its popularity increases and the need to model complex data becomes the norm, this will no longer be a disadvantage. It is the most modern of database modeling, so it is expected not have a well-defined community of experts on its use and full potential.

In conclusion NoSQL and graph database have their advantages and disadvantages, the proper choice of database management system depends on what the client wants the data to model. Less complex data can easily be modeled with a relational database management system, data of intermediate complexity can be modeled with NoSQL more efficiently and data of the highest complexity should be left to be modeled using a graph database management system. Therefore, the selection of the correct database management system is critical for the efficiency of its use. For example, you wouldn't want to have implemented data into the database and not have it be able to answer the queries that are important to the function of the applications that it supports. The reverse of this scenario should also be taken into consideration, for example you wouldn't want to use a graph database management system that provide the ability to formulate complex queries that are not necessary for a simple application that neve requires such queries therefore waste memory and dollar.

Oracle NoSQL Database (2021, November 21). In *Wikipedia*. https://en.wikipedia.org/wiki/Oracle_NoSQL_Database

Oracle NoSQL Database (2022). In *Oracle*. https://www.oracle.com/database/nosql/technologies/nosql/

(2022, February 02) Azure Cosmos DB resource model. Retrieved from https://docs.microsoft.com/en-us/azure/cosmos-db/account-databases-containers-items

(2006). 7th USENIX Symposium on Operating Systems Design and Implementation.

Bigtable (2022, March 3). In *Wikipedia*. https://en.wikipedia.org/wiki/Bigtable

Advantages and Disadvantages of NoSQL Databases. Retrieved from https://technologypoint.in/advantages-and-disadvantages-of-nosql-databases/

Graph Database Defined. In *Oracle*. https://www.oracle.com/autonomous-database/what-is-graph-database/

(2020, December 18). *What is Amazon Neptune?* Retrieved from https://docs.aws.amazon.com/neptune/latest/userguide/intro.html

Innovation & Technology In *AllegroGraph*. https://allegrograph.com/innovation-technology/

Wu, M. (2019, November 4). *What are the Major Advantages of Using a Graph Database?* Retrieved from https://www.tigergraph.com/blog/what-are-the-major-advantages-of-using-a-graph-database/

Dancuk, M. (2021, April 22). *What is a Graph Database?* Retrieved from https://phoenixnap.com/kb/graph-database

(2019, October 30) *Graph database explained*. Retrieved from https://www.ionos.com/digitalguide/hosting/technical-matters/graph-database/