Rheann L Vera

Professor Garcia

CS 300-ON

October 29, 2023

Non-Relational Data Storage and Retrieval Systems

In this era, as technology advances and more users contribute to the growth of information on the internet, it becomes a challenge for relational database management systems (RDBMS). Overcoming these challenges of unstructured growth, developers have switched to NoSQL databases. With businesses and companies requiring large volumes of data to be processed and analyzed in real-time, it's not suited to relational databases. NoSQL is a type of database management system (DBMS) widely used in applications to process large volumes of data because of its flexible data models. The term means non-SQL referring to non-relational which is a term interchangeably used or not only SQL. NoSQL "databases can be queried using idiomatic language APIs, declarative structured query languages, and query-by-example languages, which is why they are also referred to as "not only SQL" databases." ("What is NoSQL?", n.d.)

With an increase in digital interactions and data consumption, this has affected how businesses approach managing and accessing their data. Moving towards databases to handle high capacities of unstructured and semi-structured data, having more flexible schema models that can adapt and grow as fast as it's being received and can horizontally scale. Relational database management systems (RDBMS) have a structured approach by normalizing the data storing them into columns creating tables and defining them by keys. NoSQL can consume unstructured or semi-structured data and store the data in a model that can accommodate it, keeping all its elements within that element without separation. It offers a quicker way of processing and analyzing the different types of data in real-time, in turn allowing a higher capacity.

Unlike RDBMS, NoSQL is not limited to one type of database. RDBMS are organized only using tables. The data is structured into rows which hold individual elements, and columns representing the categories of data arranged in multiple tables that are combined and assigned a key. The key is used as a unique identifier and displays relationships among linked information across various tables. This level of organization requires a fixed schema, so each table has the columns and datatypes known beforehand and then written into the tables.

NoSQL has four main classifications, document databases, key-value stores, column-family stores, and graph databases. Each type is architecturally different allowing large volumes of data with flexible organization. With four ways to organize data compared to the one with RDBMS, this allows developers the choice to choose what schema design and relationship will better suit their needs.

Document databases or document stores store semi-structured documents like JavaScript Object Notation (JSON) and are commonly used for content management. MongoDB is a program and a tool that operates on a document database and is made up of collections and documents. The documents contain the data the user wants to store in their database. Their unit of data is made up of key-value pairs, using a primary key as a unique identifier, and supports many languages such as C, C++, C#, Go, Java, Ruby, Python, and Swift. Their documents differ from JSON and use a variant called Binary JSON (BSON) for its accommodation to more data types. Changing the structure of the document is performed by adding or deleting new or existing fields.

Key-value stores are a simple model using key-value pairs, where the key identifies an associated value. These stores are commonly used in gaming or finance applications. Aerospike database adopts this form of NoSQL and is considered a row-oriented database. Their unit of data for storage is a record, an object like a row in RDBMS. Each record contains multiple components, a key, metadata, and a bin. The record is uniquely identified by a key created by the client hashing the key. The metadata contains information like time-to-live (TTL) expiration date and last-update-time (LUT) of when the record was last updated. A bin is where the record data is stored and has a name and value, bins are not defined by data types and each record can have multiple bins and data types. The data type is determined by the value within the bin, this allows for flexibility as the data grows and progresses. Changing its structure can be done by adding and removing bins from the record.

Column stores utilize a table-like structure storing data in numerous columns and can be used with internet searching. Google Cloud Bigtable implements a column-oriented data store to handle their volumes of data, which are stored in scalable tables with a sorted key and value map. The rows are comprised of single elements and columns contain individual values for each row. Each row is arranged by a row key and columns that share connections in data are combined into a column family. Each row and column crossing can contain multiple cells, within a cell there is a timestamped version of the data for that row and column. When multiple cells are being stored in a column it showcases how the data has changed over time for that row and column. Although numerous cells can be stored within a row/column, if a column isn't used in a row it doesn't take up space and can be left unused.

Each NoSQL database has marketed similar factors to its program, and it's been made clear that having a dynamic or schema-less model, the ability to horizontally scale, flexibility, and high performance are key advantages. With sources of data coming from areas such as social media, the difficulty of arranging the data in predefined schemas lacks application speed in development. Having dynamic or no schema allows the data to be processed faster with or without structure and semi-structure. With changes being made to the stored data, this makes it easier to adjust. The four categories as previously discussed allow the data to be stored with multiple formats and data types. NoSQL databases are designed to scale out or horizontally rather than scaling in, by adding more of the same hardware that is distributed. This benefit can continue to scale by adding more rather than purchasing more expensive hardware with more storage or upgrading. Examples of horizontal scaling databases are MongoDB and Cassandra. The more clusters added allow the data to be distributed amongst the machines, lightening the load with each addition increasing performance. The combination of key features in non-relational DBMS, handling traffic, and sizable amounts of data has improved compared to relational DBMS which has led to an overall higher and faster performance.

Although these advantages have helped overcome challenges with big data, disadvantages to NoSQL should help in choosing between NoSQL and relational databases. NoSQL as a newer DBMS, lacks the maturity relational databases have, this can mean less support making it less reliable and less secure. With different data models within NoSQL, it isn't standardized. Each database has its own syntax for querying and managing data while traditional databases use SQL which has become a common language and well known for decades of use. NoSQL databases are open-source and because there isn't a standard for this database at this time, no databases created are alike or made equally. Although this can benefit an organization to create their databases with no restrictions there's a learning curve making it more difficult to apply NoSQL databases. Shifting from a rigid schema can mean a loss of data integrity. Especially with transaction data, relational databases maintain their consistency and integrity because they follow ACID properties. The ACID acronym stands for atomicity, consistency, isolation, and durability. It means that each transaction before and after it will be fully complete or nothing at all, it will be executed with no interference and if it fails the transaction will be placed back to its original state, and once completed it will remain completed regardless of system failures. The focus of details as described is something non-relational databases can lack. The focus of NoSQL is to store and manage large volumes of data but with little functionality and it can be a challenge to manage than relational databases.

Graph databases organize data with nodes and edges, the nodes store the elements and are connected by edges showing the relationship between elements. Its ability to show relationships this way displays a better representation of data relationships compared to other databases. “It can describe parent-child relationships, actions, ownership, and the like” (“What is NoSQL?”, Amazon, n.d.). The edge functions with a start node, end node, type, and direction and there is no limitation on size or the kind of relationships a node can have, allowing a stable environment for more complex queries. This form of database can be used in recommendation engines. There are two types of graph databases, property graphs and RDF graphs. Property graphs have relationships that are not only declared connections by carry properties such as a name or type. Its focus is to display connections with data that are dispersed within data architectures and data schemas. Showing also how different types of metadata relate. Use cases for property graphs can include tracking networks of people and their connections on social networks and for financial services such as simplifying fraud detection by graphing transactions between entities and their relationships. RDF graphs which stand for resource description framework. It was created to model metadata and showcase data integration. RDF graphs are expressed in a triple store of a three structure, subject-predicate-object. The subject and object are represented by two nodes, the starting and ending nodes, they hold the data entities while the edge is the predicate connecting the subject and object nodes, representing the relationship between the object and subject.

Amazon Neptune is an example that utilizes a graph database. Neptune use cases for recommendation engines, fraud detection, knowledge graphs, drug discovery, and network security (“What is Amazon Neptune?”, Amazon, n.d.). Amazon markets Neptune to be fully manageable meaning data management tasks will be covered from hardware provisioning, software patching, setup, configuration, or backups. The Neptune graph has a unit of data described as a quad element or four positions, consisting of subject (S), predicate (P), object (O), and graph (G). The four positions is a statement that creates a declaration about one or more resources. The statement declares an existing relationship between sources, or the resource can be a key-value pair. Predicate describes the type of connection being declared. The object shows the value of the resource, and the graph position has a graph identifier but dependent on how it’s used the identifier can change. An example Amazon describes how its graph model works, “a relationship between two vertices can be represented by storing the source vertex identifier in the S position, the target vertex identifier in the O position, and the edge label in the P position” (“Neptune Graph Data Model”, n.d.). A set of quad statements with relationship identifiers establishes the graph.

Another example that works differently is the IBM Db2 Graph. Db2 Graph interprets Db2 data to graph queries without needing third-party software, without exporting the data or transforming it. It was made possible by using a graph overlay file that defines each row in a table as an edge (“IBM Db2 Graph”, 2023). The data model defines all edges that exist in the database and their relationships. “The simplest way to map relational tables to a graph involves using the graph modeler functionality in the Db2 Graph user interface (UI). The Graph modeler builds a JSON document that is stored in the Db2 Graph metadata table in the Db2 instance associated with a connection, and the JSON describes the mapping of Db2 tables in vertices and edges” (“IBM Db2 Graph”, 2023). The most popular example of a graph database is Neo4j. Neo4j Graph database is its primary product and can handle transactional and analytic data. It’s created for traversing paths through the data by utilizing relationships in the graph to find connections between elements. It challenges one of the disadvantages of using NoSQL databases by being an ACID-compatible transactional database. Using Cypher Query Language (CQL) to create, modify, interlink, and delete nodes to change its structure without using more complex queries which is something graph databases tend to lean towards.

The benefits of using graph databases lie mainly in their structure, finding complex relationships and patterns in data is made easier when using nodes and edges. The connections between elements are not overshadowed by the elements themselves and this is not easily seen in relational databases. Relationships are prioritized in this database and are one of the main reasons to utilize this data model over others. Graphs provide flexibility that won’t impact existing data or its structure when changing or adding data, especially with rapid and complex data requirements. Adding nodes does not have a limitation and this extends to its scalability, it provides horizontal scaling to continue to add data without a lack in performance. There is an increased efficiency in graphs in comparison to RDBMS, graph queries are shorter with the same reports by its use of linked nodes. The physical linking of nodes is what makes relationships between nodes easier to find, this is a simpler model compared to relational databases making it easier to use as well. Traversing the relationships in a graph is a quicker procedure because in relational databases the connections made between tables require an index to be looked up by its key. With graph databases having more specific functionality the data model is not ideal for all types of data or applications. For simple queries or data that doesn't require a complex form to represent its data and relationships, it isn't necessary to use graph databases and because it's geared for more complex queries this can become a slower process in searching. Although it's a part of the NoSQL family, there isn't a standardized language like SQL in relational databases so its syntax is independent of its data model which can be more difficult to learn in comparison to SQL databases. Not only is its syntax independent but so is its incorporation with other tools and systems that RDBMS are equipped to work with. Graph databases are a newer data model and are currently still in a developmental phase that cannot completely replace relational databases but is becoming an important tool for many organizations.

In conclusion, NoSQL databases provide many advantages over traditional relational databases. To name a few, its ability to manage and analyze high volumes of data in real-time, its ability to horizontally scale, partitioning data into clusters of machines allowing a shared load of data rather than overworking one machine making it more cost effective. Purchasing the same hardware in multiples makes expenses more predictable and easier to accumulate than upgrading or purchasing more storage. This can limit options making it more expensive. The variety in data models over four different types of databases document-based, key-value pairs, column-oriented store, and graph databases, provides organizations the ability to implement which model can better suit their needs. The flexibility of organizing unstructured or semi-structured data with its dynamic schemas provides capabilities for their data to grow and change over time. Even with the constant change and growth in data, it remains higher in performance over RDBMS. There are still things NoSQL lacks that keep relational databases in use. Such as its lack of standardization there are no two NoSQL databases created the same and each has its own syntax. The lack of support for complex queries is because its focus is the quantity of data, meaning less functionality. Both non-relational and relational database management systems have their place and based on their use case and type of data should determine the best fit.

Works Cited

17 use cases for graph databases and graph analytics - oracle. (n.d.).

https://www.oracle.com/a/ocom/docs/graph-database-use-cases-ebook.pdf

Data model. Aerospike Documentation. (2023, March 30).

https://docs.aerospike.com/server/architecture/data-model

GeeksforGeeks. (2023a, February 22). *Introduction to graph database on NoSQL*. GeeksforGeeks.

https://www.geeksforgeeks.org/introduction-to-graph-database-on-nosql/

GeeksforGeeks. (2023, October 13). *Introduction to NoSQL*. GeeksforGeeks.

https://www.geeksforgeeks.org/introduction-to-nosql/

Gillis, A. S., & Botelho, B. (2023, March 7). *What is MongoDB? features and how it works – TechTarget definition*.

Data Management.https://www.techtarget.com/searchdatamanagement/definition/MongoDB

Google. (n.d.). *Bigtable overview | Cloud Bigtable documentation | google cloud*. Google.

https://cloud.google.com/bigtable/docs/overview

IBM DB2 graph. Graph and network analysis with IBM Db2 Graph. (n.d.).

https://www.ibm.com/docs/en/db2/11.5?topic=deployments-db2-graph

Kovačević, A. (2023, February 21). *What is a relational database? {examples, Advantages & disadvantages}*.

Knowledge Base by phoenixNAP. https://phoenixnap.com/kb/what-is-a-relational-database

MENEGASSO, A. EDUARDO. (2018). *NOSQL*. Amazon. https://aws.amazon.com/nosql/graph/

Mullins, C. S., Vaughan, J., & Beal, B. (2021, April 8). *What is NoSQL and how do NoSQL databases work?*. Data

Management. https://www.techtarget.com/searchdatamanagement/definition/NoSQL-Not-Only-SQL

Vogt, G. (2000). *Neptune*. Amazon.

https://docs.aws.amazon.com/neptune/latest/userguide/feature-overview-data-model.html

What is a relational database?. IBM. (n.d.). https://www.ibm.com/topics/relational-databases

Why do developers prefer nosql databases?. Oracle. (n.d.).https://www.oracle.com/database/nosql/what-is-nosql/