Christopher Burkhead
CS300-ON
Assignment 1

NoSQL is a terdeveloped to describe non-relational databases that store vast sums of unstructured data differently than SQL relational databases. (Microsoft) It essentially means “Not only SQL” or “No SQL” as a rebuttal to SQL databases being the industry standard since the 1970s. (Wikipedia) The term NoSQL came about in the early 21st century as web 2.0 companies were looking for new ways to store the vast amount of real time data that they were gathering from their users. (Wikipedia)

The problem these companies were running into was that they were gathering data that was non-relational, meaning that there was not a direct link between two objects, such as a person and their address, that relational databases rely on. Not wanting to toss out this data as it could prove to be useful, they needed a way to store it in a database that did not have to be structured in restricted tables, nor did they want to have a limit on the type of variables their databases could store. This led to the proliferation of NoSQL databases which were trying to solve those two main problems of relational databases: their lack of scalability and the limits that relational table designs place on a database. (Team Zuar)

Regarding scalability most SQL databases require that you scale up vertically, which means migrating the database over to a larger, more expensive server. (MongoDB) Creating a SQL database and its DBMS is already a huge investment, which might be unattainable for small startup and development companies. The thought of having to continue to invest in its maintenance might not be an option for these web companies. The data they were collecting was growing exponentially, this would result in them needing to scale up vertically at an economically unfeasible rate. NoSQL databases on the other hand can be scaled horizontally, meaning they can grow outwards by adding cheaper commodity servers when they needed to expand their storage capabilities. (MongoDB)

Additionally, NoSQL databases allow for greater schema flexibility, SQL’s schemas are very rigid and are labor intensive to make changes. Whereas NoSQL databases have very flexible schemas so database requirements can change easily as the needs of the database change. (MongoDB)

Lastly, the one size fits all data storage modeling of SQL using tables with fixed rows and columns was not the best for all data types. NoSQL databases allow for multiple storage solutions including documents, Key-value, Wide Column, and Graph storage models. (MongoDB) This allows for a greater amount of data or variable types to be stored.

Now for all the advantages that NoSQL databases provide, there are drawbacks to using them. These disadvantages mostly tie in to how young the current generation of NoSQL databases are and their lack of standardization across the industry. First, consistency, NoSQL databases do not pass the ACID test on their own, that is they do not provide Atomic, Consistent, Isolated, Durability transactions. (BLUE CLAW) Which means that NoSQL databases cannot guarantee that their data operations are 100% dependable, without having to add custom code. Several NoSQL databases claim that they meet ACID transaction requirements with their additional programming but that leads into the next disadvantage of NoSQL databases, lack of maturity/support.

Since this current generation of NoSQL databases are still in their childhood to preteen years of development, they lack maturity when it comes to having all the features and specifications that SQL databases have built in. And even for the NoSQL databases that claim to have similar or congruent features, there aren't enough support staff or programmers to assist with ensuring they are meeting those expectations. (BLUE CLAW) Or worst-case scenario if a failure were to occur, is there enough support professionals to be able to repair the database.

This ties back into the first disadvantage, with no industry standards yet developed across the multitude of NoSQL databases and how they each solve their issues, there's not enough open-source knowledge or professionals that can safely repair all damaged NoSQL databases. This simply is not an issue for SQL databases that have the standards and industry of the past 50 years to support it.

One last disadvantage of NoSQL databases which might be its biggest disadvantage is efficiency. Not efficiency of power or resources, as you might recall, that is an advantage for NoSQL databases over SQL ones. Efficiency in analytics, knowing that if you request specific information about the data you store that you are getting that exact information. (BLUE CLAW) No matter which SQL database product your company chooses to utilize, you would use SQL language queries to draw your business insights from it, which provide you the reliability to know you are getting exactly what you are looking for. NoSQL databases on the other hand, have no industry standard when it comes to the language or interfaces used to enter data or run analytics on their databases. In fact, most NoSQL databases allow for multiple different interfaces or programming languages to be used. (Teplow) While some laud this as an advantage because it provides greater optimization for a company and their database, currently it creates an issue that if your database staff are not trained the same, two staff could potentially get different answers to the same question from a NoSQL database depending on the language and interface they used.

Christopher Burkhead
CS300-ON
Assignment 1

Now let's say that after weighing the pros and cons your company has decided to invest in a NoSQL database to manage its data. What type of NoSQL database do you choose? If you recall one of the advantages of NoSQL databases is that they have several different data storage models they can utilize depending on your companies needs. Below are some examples of how those data storage models work and the NoSQL database products that offer them.

Apache Couch DB uses a document-oriented database approach to store its data, which is every object to be stored has its own document that can contain whatever it wants. In addition, no two documents need be alike, they can store completely different information, in no particular order. These documents even allow for metadata to be stored in them. (Apache)This is different from a relational database as those require every record to contain the same fields in the same order. (Wikipedia) The one common thread between the documents is the encoding used to document the data. Couch DB uses a file format called JSON "that uses human-readable text to store and transmit data objects consisting of data attribute-value pairs and arrays." (Wikipedia) Other popular encoding formats for document-orientated databases include XML, YAML and BSON.

Redis (Remote dictionary server) is an example of a Key-value NoSQL database. Key-value pairings have been around the computing world since Charles Babbage and his Analytical Engine. (Wikipedia) Key-value databases are designed to store and manage hash tables. Hash tables also called dictionaries contain a series of records or objects, which are then broken down into fields, where the data is contained. A Key is then developed that is specific to that object so that it can be found in the database. Redis supports a variety of abstract data structures including bitmaps, spatial indices and sets, just to name a few. Redis most popular attribute is that it can act as storage and cache for data at the same time. It does this by constantly modifying data using the computer's RAM storage, but it would also store the data on the computer's disk in a format unreadable for RAM. It then reconstructs the data when the system gets restarted. This allows for real time manipulation of new data, while also ensuring it gets stored. (Wikipedia) Users do not use queries to interact with the database, instead they provide specific commands that are to be performed on the abstract data types. Redis is supported by almost all major programming languages making it a very popular if not the most popular Key-value database. (Wikipedia)

Wide column databases take the table format of relational databases of using rows and columns but gets rid the of formatting restrictions that relational databases require. The names and format of each column can vary from

row to row, providing greater flexibility to the database. (Wikipedia) One of the most well-know examples of a wide-column NoSQL database is Scylla DB, which claims to be one of the fastest databases on the market. They achieve their remarkable high speed and low latency by having each core of a CPU manage a different subset of data, never sharing the data with the other CPUS and only communicating with each other when they absolutely have too. (Wikipedia)

The last type of NoSQL database to discuss is a Graph database. The goal of a graph database is to display the data in an abstract way, so that the relationships between data elements is mapped out for the user to see. Graph databases accomplishes this by using a system of nodes and edges. Nodes represent the actual instance that the data is to capture, whether that be a business, an account number, a person, etc. it is the equivalent of a row in SQL databases. Nodes are linked together using edges, which are lines drawn in between nodes to delineate that a relationship exists. The edges are then labeled to explain what the relationship is. A graph can be further detailed as either being directed or undirected. In a directed graph arrows are added to the edges to display that the edges have different meanings, in an undirected graph the edges have a single meaning. (Wikipedia)

One of the main problems that graph databases are trying to solve is the lack of prioritizing relationships with NoSQL databases and trying to enhance the relationships that SQL databases manage. With companies today gathering so much complex data, of different variable types, it's difficult to see the inherent relationships that exist within the data. This is true whether you are using a NoSQL or SQL database. While a SQL will eventually find the relationship, it takes several complex joins that involve developing tables and indexes to get the answer. (Wikipedia) Graph databases on the other hand are capable of quickly and efficiently finding relationships between complex data points. It does this by finding the data points or nodes requested and then following edges that exist between the nodes. These edges then clearly define the relationship that exists between the nodes.

Finding these relationships in the complex data sets was a priority for certain companies, namely retail, social media, and security companies. (Joyce) They had massive data sets and assumed there was relationships between them, but how to find them? Using a SQL database, you had to ask a specific question or hypothesis to see if a relationship existed between data points. With a graph database though, you can just ask it to find any relationships that exists between data points, and it will then clearly map. This presented companies with a whole new research tool to investigate their data sets with. It eliminated a lot of the guess and check work that is part of

using a relational database for making business decisions, which could be slow and costly. Graph databases allow companies to make split second decisions based on the relationships being presented to them in real time. (Joyce)

Another problem that graph databases were hoping to solve was how to visualize the relationship between data points in an easy digestible format. Graphs have always been a great tool to visualize data sets, but as the data sets grew more complex the harder it was for users of SQL databases to demonstrate the relationships, especially if they were presenting this information to others not familiar with how the database worked. Graph databases simple construction of using nodes and edges are easily formattable in a variety of presentation styles, and with the nature of the relationships already being stored on the edges, only made the connections clearer for others to understand.

An example of a graph database is Neo4j which is said to be the most famous DBMS system. (Geeks for Geeks) It does not require a predefined schema; it allows the user to define any relations between nodes that they want . Neo4j also claims to adhere to full ACID rules which is a big plus for NoSQL databases. Lastly, it uses the Cypher query language, which is a declarative type using commands in an easy to learn human readable format, which allows the user to represent the graph visually. (Tutorialspoint)

OrientDB is another graph database, which also supports document, key/value and object models, but all the relationships in those other model types are managed using the graph database. For example, relationships between documents are established using edges, allowing for quick retrieval without having to use JOIN statements. (Wikipedia) It also claims it is fully ACID compliant for all transactions. Best of all OrientDB is open-source software that can be downloaded and used by any one for free.

As mentioned above there are clear advantages to using graph databases, namely its ability to find relationships amongst complex data sets, being able to represent those relationships in an easy-to-understand visual presentation and its ability to incorporate real time data. But like all technology nothing is ever perfect so what are the disadvantages to using graph databases?

Disadvantages to graph databases are almost identical to the ones shared amongst all NoSQL databases. First, lack of standardization, especially as it pertains to query languages. Each graph database platform creates their own query language, and while some claim their languages are easy to use or learn (OrientDB use Java code for its query language) this lack of standardization hurts the industry as database managers must learn a new language for

every new database they encounter. This leads to another previously discussed disadvantage lack of support. Graph databases like most NoSQL databases are still young in their development cycles, this leads to a dearth of knowledge regarding how they individually operate and professionals that can provide support when needed. (PhoenixNAP) One last disadvantage of graph databases is that they are too focused on relationships, which make them ideal for social networking sites, but not for transactional or operational purposes. Also, the relationships are defined by the users entering in the data. This could lead to human error of mislabeling the relationships, which would ultimately provide incorrect analysis.

Christopher Burkhead
CS300-ON
Assignment 1

Works Cited

Apache. *Couchdb.apache.org*. UNK UNK 2022. website. 2nd September 2022. <https://couchdb.apache.org/#about>.

BLUE CLAW . *NOSQL ADVANTAGES AND DISADVANTAGES*. UNK UNK 2022. Article. 2nd September 2022. <https://blueclawdb.com/nosql/nosql-advantages-disadvantages/>.

Geeks for Geeks. *Neo4j Introduction*. 16th August 2019. Article. 3rd September 2022. <https://www.geeksforgeeks.org/neo4j-introduction/#:~:text=Neo4j%20stores%20and%20present%20the%20data%20in%20the,making%20it%20unique%20from%20other%20database%20management%20system.>.

Joyce, Kara E. *Graph database vs. relational database: Key differences*. 15th January 2021. Article. 3rd September 2022. <https://www.techtarget.com/searchdatamanagement/feature/Graph-database-vs-relational-database-Key-differences>.

Microsoft. *Microsoft Azure*. UNK UNK 2022. Webpage. 31 August 2022. <https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-nosql-database/>.

MongoDB. *MOndoDB.com*. UNK UNK 2022. artilce. 2nd September 2022. <https://www.mongodb.com/nosql-explained/nosql-vs-sql#:~:text=What%20are%20the%20benefits%20of%20NoSQL%20databases%3F%201,databases%20can%20be%20faster%20than%20SQL%20databases.%20>.

PhoenixNAP. *What Is a Graph Database?* 22nd April 2021. Article. 3rd September 2022. <https://phoenixnap.com/kb/graph-database#:~:text=The%20general%20disadvantages%20of%20graph%20databases%20are%3A%201,to%20find%20support%20when%20running%20into%20a%20problem.>.

Team Zuar. *What's the Difference Between SQL & NoSQL*. 13th December 2021. 31st August 2022. <https://www.zuar.com/blog/whats-the-difference-between-sql-nosql/#:~:text=More%20developers%20starting%20working%20on%20NoSQL%20databases%20to,stem%20from%20the%20constraints%20of%20most%20relational%20databases.>.

Teplow, David. *The Problem with NoSQL Databases*. 21st June 2016. Article. 3rd September 2022. <https://www.linkedin.com/pulse/problem-nosql-databases-david-teplow#:~:text=So%2C%20once%20again%2C%20I%E2%80%99m%20a%20huge%20proponent%20of,common%20and%20standard%20way%20of%20working%20with%20them.>.

Tutorialspoint. *Neo4j - Overview*. UNK UNK 2022. Article. 3rd September 2022. <https://www.tutorialspoint.com/neo4j/neo4j_overview.htm>.

Wikipedia. *Graph database*. 23rd August 2022. Artile. 3rd September 2022. <https://en.wikipedia.org/wiki/Graph_database>.

—. *Key-value Database*. 10 August 2022. Website. 2nd September 2022. <https://en.wikipedia.org/wiki/Key%E2%80%93value_database>.

—. *NoSQL*. 22nd August 2022. Webpage. 31st August 2022. <https://en.wikipedia.org/wiki/NoSQL>.

—. *OrientDB*. 5th August 2022. Article. 3rd September 2022. <https://en.wikipedia.org/wiki/OrientDB>.

—. *Redis*. 28th July 2022. website. 2nd September 2022. <https://en.wikipedia.org/wiki/Redis>.

—. *ScyllaDB*. 5th August 2022. Article. 2nd September 2022. <https://en.wikipedia.org/wiki/ScyllaDB>.

—. *Wide-column store*. 17th July 2022. Article. 2nd September 2022. <https://en.wikipedia.org/wiki/Wide-column_store>.

—. *Wikipedia.org*. 22nd June 2022. Website. 2nd September 2022. <https://en.wikipedia.org/wiki/Document-oriented_database>.