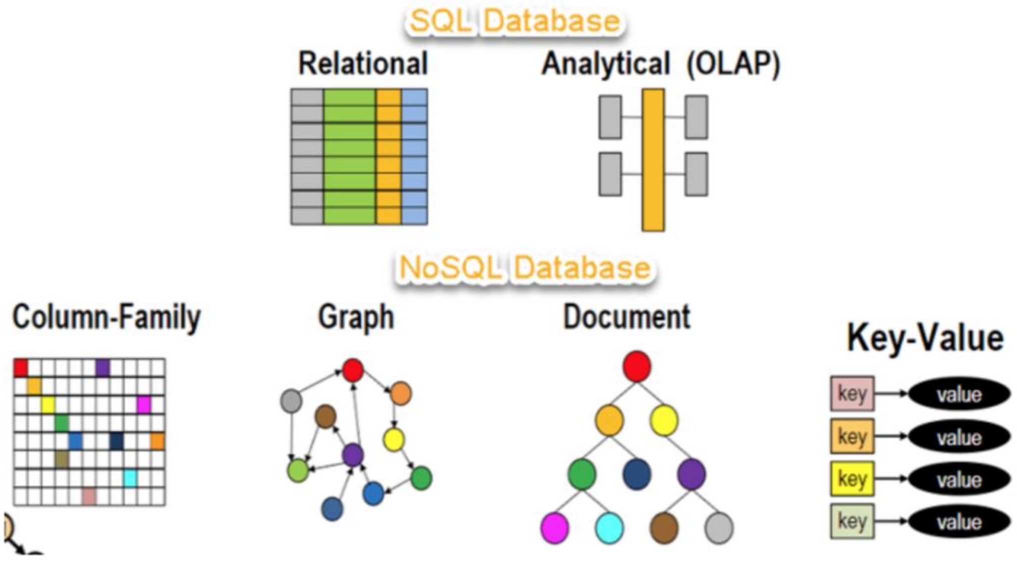
Types of NoSQL Models



In terms of big data modeling, Key/value stores and document stores are very important. Key/value stores and document stores are both operational database models, but there is an important distinction between them. First, they are what is called NoSQL databases. NoSQL databases are non-relational database models and are an alternative to relational databases (Drake, 2019).

Relational databases are very good for sorting and storing structured data. However, in modern times, there is more and more unstructured data. Thus this, along with other limitations, played a part in the search for an alternative solution (Drake, 2019). This is how NoSQL databases became in demand. Like its name, NoSQL databases don’t use SQL.

Key/value stores are a type of NoSQL operational database model. Examples of the database management systems that implement this model include Redis, MemcacheDB. It uses associative arrays (which are dictionaries or hash tables) to form a collection of key-value pairs (Drake, 2019). Each key is the unique identifier needed to retrieve the value connected to it, hence the term key-value pair.

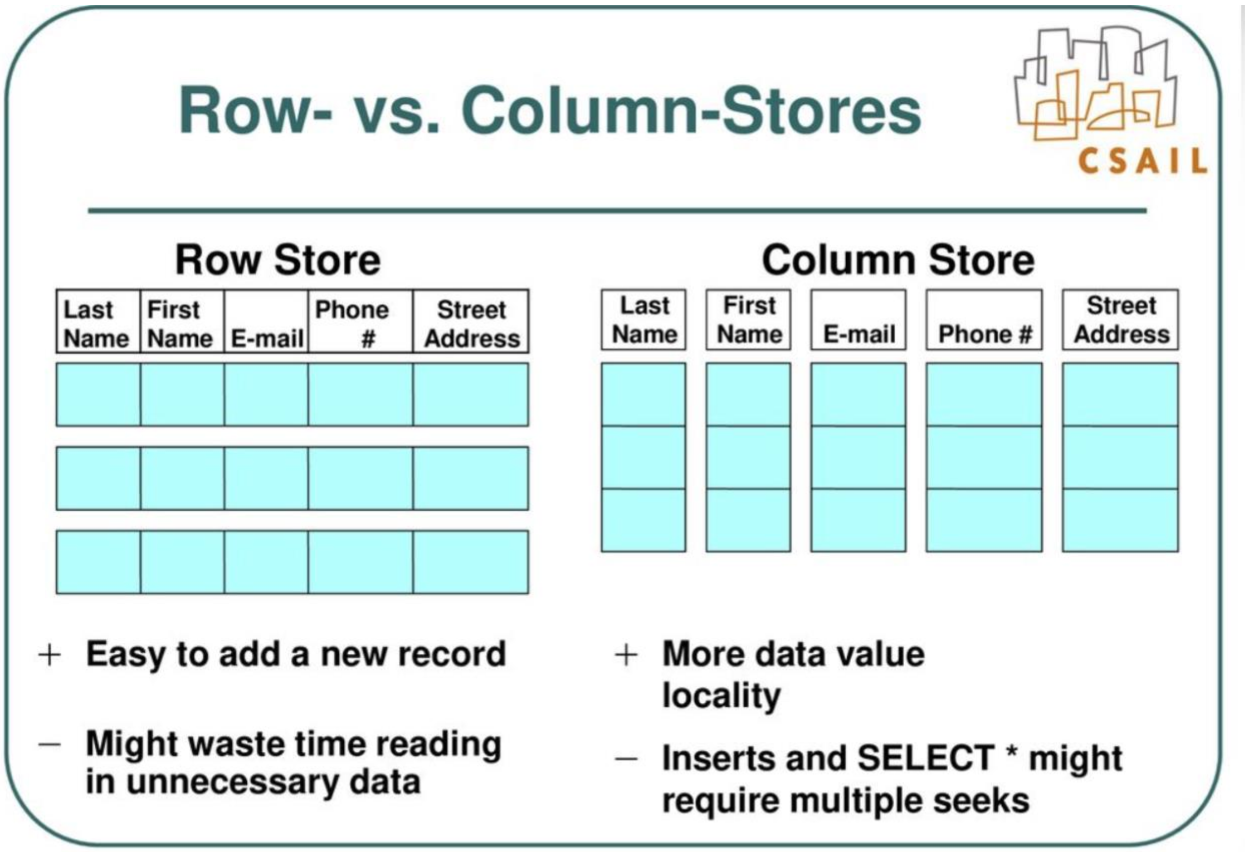
The data is stored without any sort of order or relationship. In other words, there's not much structure, if any. These databases are very efficient and scalable and can thus be used for a variety of purposes, and "Common use cases for key-value databases are caching, message queuing, and session management." (Drake, 2019).

Document stores are a specific type of key-value store. Examples of the database management systems that implement this model include MongoDB and Couchbase (Drake, 2019). The data they store are documents, so this would be the value itself.

The difference is in a regular key-value store, the data is "treated as opaque and the database doesn't know or care about the data held within it; it's up to the application to understand what data is stored." (Drake, 2019). Whereas, document stores always have metadata, or some structure to the data.

Other NoSQL models include row stores and column stores. There are also differences between them. A column store stores data in columns. Instead of storing data into tables, a column would be stored in its own file or space within the storage system (Drake, 2019). It has become more popular in data analytics due to its "fast query processing" (Drake, 2019).

A row store is basically the same thing as a column store, but instead of storing data in columns, it's in rows. Because of this horizontal orientation, the files can get big, which can slow down retrievals (Khan, 2021).


Row- vs. Column-Stores
CSAIL
Row Store
Last Name
First Name
E-mail
Phone #
Street Address
Column Store
Last Name
First Name
E-mail
Phone #
Street Address
+ Easy to add a new record
– Might waste time reading in unnecessary data
+ More data value locality
– Inserts and SELECT * might require multiple seeks

In deciding which store is better, it depends on what kind of data is needed. If you need a lot of rows and less columns, then go with a row store. If you need less rows and more columns, then go with a column store (Khan, 2021).

References

Faraz, Khan. (2021). *Columnar Vs. Row Oriented Databases - The Basics (2 min read).* Retrieved from https://www.linkedin.com/pulse/columnar-vs-row-oriented-databases-basics-2-min-read-faraz-khan

Mark, Drake. (2019). *A Comparison of NoSQL Database Management Systems and Models*. Retrieved from https://www.digitalocean.com/community/tutorials/a-comparison-of-nosql-database-management-systems-and-models

First Image: Taylor, David. (2022). NoSQL Tutorial: What is, Types of NoSQL Databases & Example. Retrieved from https://www.guru99.com/nosql-tutorial.html

Second Image: Abadi, Daniel. (2018). *Column Stores for Wide and Sparse Data*. Retrieved from https://slideplayer.com/slide/14613848/