The traditional relational database model (RDBMS) comes to mind when most people think about databases. While RDBMS continues to handle the majority of data processing, developers began looking for alternatives to the traditional relational data model due to its limitations and inadequacies in processing large volumes of non-structured data such as email messages, photographs, and videos, among other things, resulting in the rise of NoSQL databases. (Drake, 2019)

The term "NoSQL" has become a catch-all for any database that does not follow the relational model.

Document-oriented databases, or document stores, are NoSQL databases that are designed for storing, retrieving, and managing document-oriented or semi-structured data. (Sundhara Kumar et al, 2017)

Document stores are a type of key-value store (another form of NoSQL database). In a key-value database, the data is considered opaque, and the database is unaware of or unconcerned about the data it contains; it is up to the application to find out what data is being retained. In contrast, each document in a document store has metadata that provides some structure to the data. (Drake, 2019)

Unlike relational databases, which may distribute information about a single object across numerous tables or databases, a document-oriented database can store all of an object's data in a single document, and data is often stored as JSON, BSON, XML, or YAML document.

In recent years, document-oriented databases have become increasingly popular. Because of their flexible nature, they've found a lot of application in e-commerce and analytics platforms, as well as content management systems. Document stores are regarded extremely scalable and suited for storing vast amounts of unconnected, sophisticated data with various structures, with sharding being the most frequent horizontal scaling approach. (Drake, 2019)

References:

K. B. Sundhara Kumar, Srividya, S. Mohanavalli (2017). A performance comparison of document oriented NoSQL databases,International Conference on Computer, IEEE. Available at: https://ieeexplore.ieee.org/abstract/document/7944071 (accessed 03 September 2021)

Drake, Mark (2019). A Comparison of NoSQL Database Management Systems and Models, DigitalOcean. Available from: ttps://web.archive.org/web/20190813163612/https://www.digitalocean.com/community/tutorials/a-comparison-of-nosql-database-management-systems-and-models (accessed 03 September 2021)

Post by **Michael Botha**
20 days ago
Re: InitialPost

Hi Pavendran,
Thanks for the interesting post. If the document-oriented database model is built around the paradigm of semi-structured data, does that inherently mean that it is a lot slower than a relational database

Hi, Michael.

Thank you for your question. Here's my take on the subject.

Most NoSQL databases are simpler than SQL databases because they are more specialised than a SQL database's "general case" approach. This translates to fewer logic/code running on each operation, simpler data structures (which may require less IO), and, in general, lower overhead and higher speed.

Consider the following scenario: in a document store database, you'd design your schema so that a single document contains all you need to render a page. with a relational database the data would be normalised and distributed among multiple tables. You'd have to make a lot of queries to render the same page.

Although one document store database query may be slower than one relational database query, comparing one document store database query to numerous relational database queries will be substantially faster.

Post by **Oliver Buckley**
16 days ago
Re: InitialPost

Good post and overview of the subject.

What do you think the benefits of this kind of approach are and what do document stores allow us to do that we hadn't done before?

Thanks for your response Oli.

A self-described document format is used by document stores. As a result, these databases can do more validations than key-value stores or other unstructured

databases, while also being less constrained by rigid schemas compared with RDBMS.

And also secondary indexes within the value content can be defined, as the database knows the document structure.

Ability to create persisted views from a base document and store the same for analysis.

Ability to store dynamic data in unstructured, semi-structured, or structured formats.

Since document stores express the data as files in JSON or XML formats, it allows the same document to be parsed for multiple contexts and the results scrapped and added to the next iteration of the database data. An example usage: A document database can be used to store the results of online clicks, for example. A basic XML construct using the Page Name, Position Coordinates, Clicks, Keywords, Incoming and Outgoing sites, and Date Time will establish a simple model to query the number of clicks, keywords, date, and links for each log file that is parsed. This computing capacity is not available in a relational database management system (RDBMS).

Reference:

Krish K (2013) Introducing Big Data Technologies, Data Warehousing in the Age of Big Data, 2013