Types of NoSQL Databases

This lesson discusses the five types of NoSQL databases: key-value, column, graph, document and multi-model stores.

WE'LL COVER THE FOLLOWING

Data Model

As previously mentioned, *NoSQL* databases have shifted to separate themselves from relational databases, as they are no longer a part of them but what does this mean?

This means that they **no** longer use a *relational* data model.

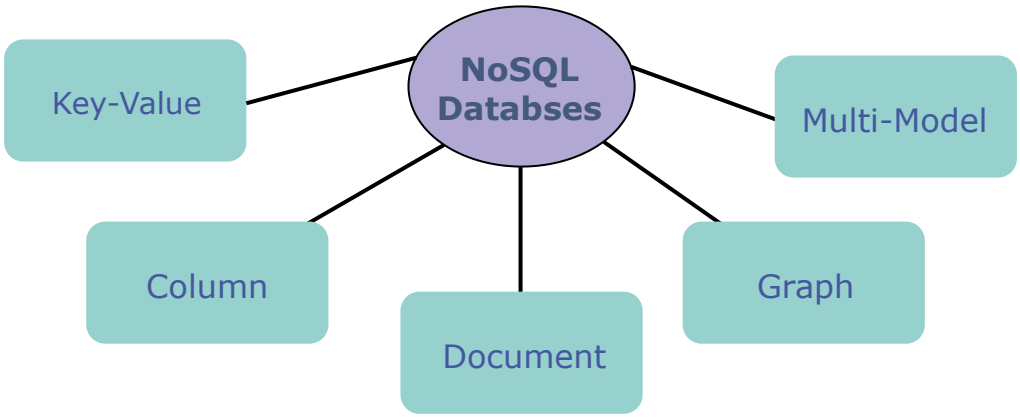
> A **database model** is a model through which we perceive data in the database.

A relational database model can be visualized as a set of tables in which each row represents a different record, a different entity.

Types of NoSQL Databases

NoSQL databases use a different approach. Based on a data model there are a few types of databases in the NoSQL world:

- *Key-Value Stores*
- *Column Stores*
- *Graph Stores*
- *Document Stores*
- *Multi-Model Databases*



Key-Value Stores

Key-Value Store is effectively an associative array stored on a disk; it is a single key lookup, a dictionary so to speak.

The good thing about these databases is that they can be read very quickly, but these databases are not so good for reverse lookups or additional analytics.

An example of this type of database is *Redis*.

Column Stores

Column Stores is the subset of *NoSQL* databases that kept, somewhat, to the tabular form.

So, what does this mean?

Column Stores vs Relational Databases

Relational Database Approach

Well, as you probably know, relational databases keep all their data in tabular form (where every row represents one entity). Since every row is saved separately on the disk, we could say that rows align the data.

When reading this kind of database, it always reads the whole row, even if not all of the data is necessary (i.e., if we only want one column of values).

Column Store Approach

Column stores, on the other hand, change this approach a bit: they store data in so-called *columned families* (i.e., in column order).

For example:

- First, Ids of all records are saved.
- Then, all of their names, etc.

Why is this a big deal?

This is a big deal because it is possible to get the whole column in a more efficient manner than it was when you got all of the rows and had to pull specific values from each one

Basically, we can get more information from the database in a single seek. Also, these databases can be easily compressed and, it goes without saying that writes are very expensive.

A typical example of these databases is *Cassandra*.

Graph Stores

Graph stores use *graph* structures for queries, with *nodes*, *edges*, and *properties*, to represent and store data.

They are used for storing a network of connections or relationships (e.g., social networks).

Graph stores are a bit different from other NoSQL databases since they originated from a different problem with relational databases–they have a number of small records with a lot of relationships between them.

An example of such a database is *AllegroGraph*.

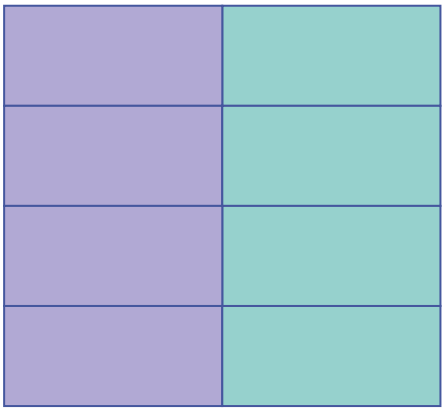
Document Stores

One of the most popular types of NoSQL databases is **Document stores**, which revolves around the concept of a document.

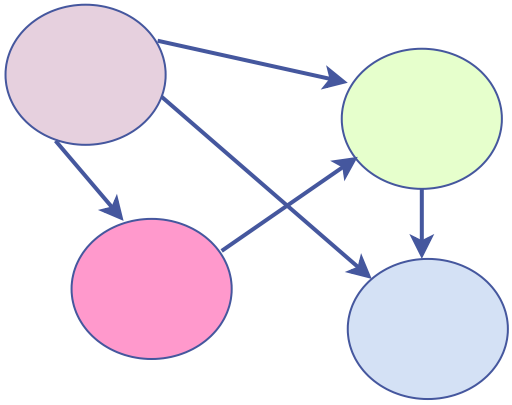
> **Documents** are self-describing structures and usually similar to each other, but they don't have to be the same.

Unlike the rows in relational databases, where every row has to follow the same schema, documents can vary from each other and still belong to the same collection.

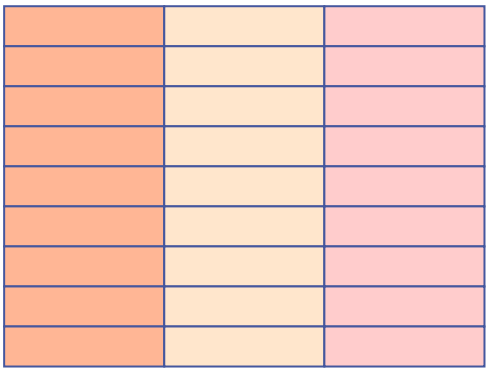
MongoDB and *Couchbase* are examples of document stores.



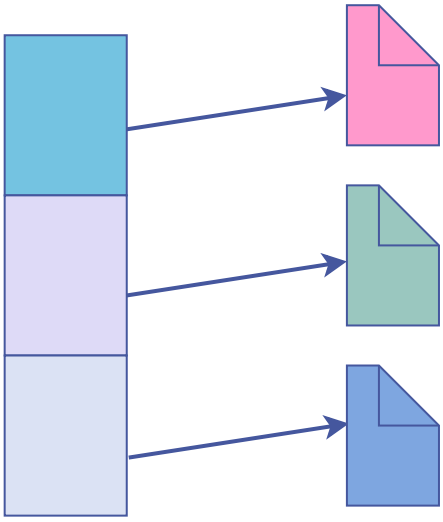
Key - Value



Graph



Column



Document

Types of NoSQL Databases

Multi-Model Databases

Multi-model databases are designed to handle *multiple* data models against a *single* integrated backend.

They are a brand-new in the NoSQL world, and there will be much more buzz around this type of database in the future.

Now that you've learned about various types of NoSQL databases, we will discuss the concept of *Polyglot Persistence*.