Overview of NoSQL

What you will learn

- Define the term NoSQL
- Describe NoSQL technology
- Describe the history of NoSQL
- List four reasons for using NoSQL databases

What is NoSQL?

- The NoSQL name was introduced during an open-source event on distributed databases
- NoSQL doesn't mean 'No SQL'
  NoSQL means '**N**ot **o**nly SQL'

NoSQL

⬇

'**N**ot **o**nly **SQL**'

What is NoSQL?

- Refers to family of databases that vary widely in style and technology
- However, they share common traits:
  - Non-relational
  - Not standard row and column type RDBMS
- Could be referred to as 'Non-relational databases'

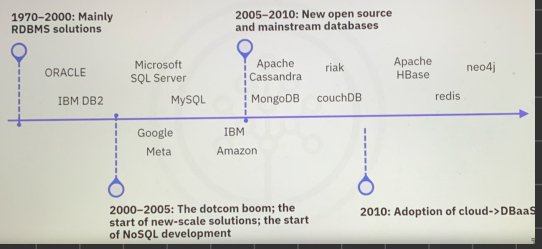
What is NoSQL?

NoSQL databases:

- Provide new ways of storing and querying data that
- Address several issues for modern apps
- Provide a flexible schema suitable for evolving use cases
- Scale to meet demand → more easily, it's scalable for data and traffic as demand grow
- Are distributed systems
  ↓provide native fault tolerance and availability

NoSQL Databases

History of NoSQL



Why NoSQL?

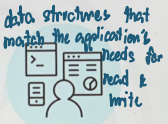
can store any data, typed more easily

• work faster with data structures that match the application's needs for read & write

• provide high availability and fault tolerance

Flexible data models

Built-in scalability

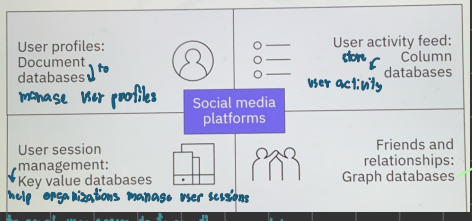


Developer productivity

Distributed databases

NoSQL in use: Social media example

Businesses can use a mix of NoSQL databases



Characteristics of NoSQL Databases

Objectives

After watching this video, you will be able to:

- Describe the concepts and characteristics of NoSQL databases
- Explain the primary benefits of adopting a NoSQL database

NoSQL database categories

- The most common trait among NoSQL databases is that they are non-relational in architecture
- What types of NoSQL databases are available?
- What is common to them?

NoSQL database categories

General consensus is...

Quiz

...NoSQL databases fit into four categories

Key-Value

Document

There is some overlap among these types, so the definition isn't always clear

Column

Graph

style NoSQL, lol

NoSQL database characteristics

Open Source Communities

But what ties NoSQL databases together?

- Majority have their roots in the open source community
- Many have been used and leveraged in an open source manner
- Open source community support is fundamental to their industry growth

NoSQL database characteristics

- Companies often develop a commercial version of the database alongside the open source version
- Examples include:
  - IBM Cloudant → couchDB
  - DATASTAX → Apache Cassandra
  - MongoDB

NoSQL database characteristics

- Differ technically, but have a few commonalities
- Most NoSQL databases:
  - Are built to scale horizontally
  - Share data more easily than RDBMS
  - Use a global unique key to simplify data sharding
  - Are more use case specific than RDBMS
  - Are more developer friendly than RDBMS
  - Allow more agile development via flexible schemas

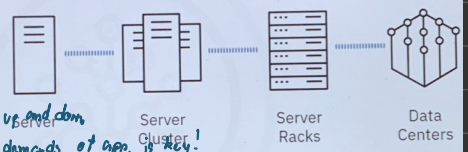
Benefits of NoSQL databases

Why use a NoSQL database?

Why is their popularity growing so rapidly?

Benefits of NoSQL databases

Scalability :

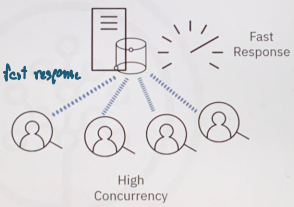


The elasticity of scaling both up and down to meet the various demands of app. is key!

Benefits of NoSQL databases

Performance

the need to deliver fast response times



Benefits of NoSQL databases

Availability

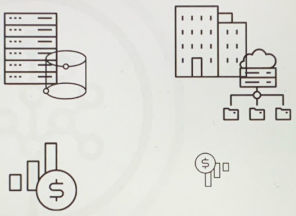
run on a cluster of servers with multiple copies of data makes for a more resilient solution

Benefits of NoSQL databases

Cloud Architecture

↙ modern

↘ massively reducing cost



Benefits of NoSQL databases

Cost

NoSQL

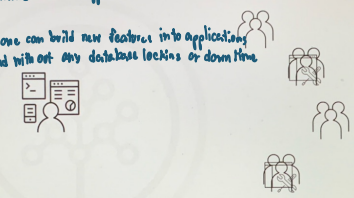
reduce cost and still able to get the same or better performance and functionality

Benefits of NoSQL databases

Flexible Schema

↙ dev love when wanting to build apps.

↘ mean that one can build new features into applications quickly and without any database locking or downtime



Benefits of NoSQL databases

Varied Data Structures

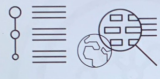
↑ quick lookup

doc. store for storing denormalized intuitive information

for associative data sets

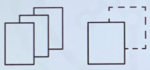
Benefits of NoSQL Databases

Specialized capabilities



Indexing and Querying

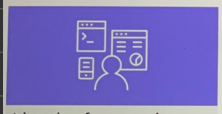
Modern HTTP APIs



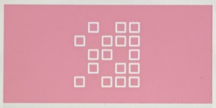
Data Replication Robustness

Key-Value NoSQL Databases

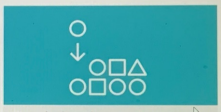
What you will learn



List the four main categories of NoSQL database



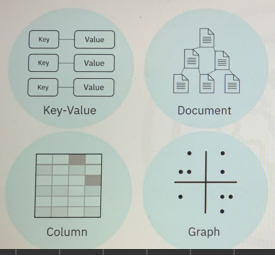
Describe the architecture of the Key-Value category of NoSQL database



Explain the primary use cases for the Key-Value NoSQL database category

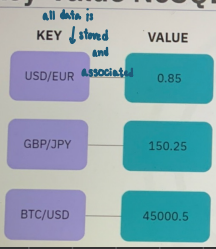
NoSQL database categories

great fits for diff types of applications



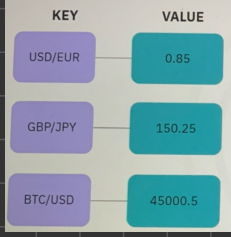
Each category has:
- Unique characteristics
- Architecture

Key-Value NoSQL database architecture



- Least complex (reason)
- Represented as a hashmap
- Ideal for basic CRUD operations
- Scales well
- Shards easily
  - each shard would contain a range of keys and their associated values

Key-Value NoSQL database architecture



- Not intended for complex queries
- Atomic for single key operations only
- Value blobs are opaque to database
  - Less flexible data indexing and querying

Key-Value NoSQL database use cases

Suitable use cases

- For quick basic CRUD operations on non-interconnected data (Ex)
  - Storing and retrieving session information for web applications
    - each user session would receive some sort of unique key and all data would be stored together in the opaque value blob
    - no need to query into, just unique key
- Storing in-app user profiles and preferences
- Shopping cart data for online stores

Key-Value NoSQL database use cases

Unsuitable use cases

- For data that is interconnected with many-to-many relationships
  - Social networks
  - Recommendation engines

key-value has poor performance for

- When high-level consistency is required for multi-operation transactions with multiple keys
  - Need a database that provides ACID transactions
- When apps run queries based on value vs. key
  - Consider using the 'Document' category of the NoSQL database

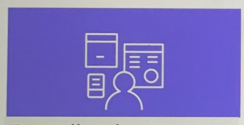
Key-Value NoSQL database examples

Popular key-value NoSQL databases include:

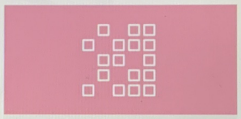
- Amazon DynamoDB
- Oracle NoSQL Database
- Redis
- Aerospike
- Riak KV, MemcacheDB

Document-Based NoSQL Databases

What you will learn



Describe the architecture of the document category of NoSQL databases



Explain the primary use cases for document NoSQL databases

Document-based NoSQL database architecture

↳ bvid off key-value model

- Values are visible and can be queried
- Each piece of data is considered a document
  - Typically JSON or XML
- Each document offers a flexible schema
  - No two documents need to contain the same information

Document-based NoSQL database architecture

- Content of document databases can be indexed and queried
  - Key and value range lookups and search
  - Analytical queries with MapReduce
- Horizontally scalable
- Allow sharding across multiple nodes
- Typically only guarantee atomic operations on single documents

Document-based NoSQL database use cases

Suitable use cases

- Event logging for apps and processes: Each event instance is represented by a new document
- Online blogs: Each user, post, comment, like, or action is represented by a document
- Operational datasets and metadata for web and mobile apps: Designed with the internet in mind (JSON, RESTful APIs, unstructured data)

Document-based NoSQL database use cases

Unsuitable use cases

- When you require ACID transactions
  - Document databases can't handle transactions that operate over multiple documents
  - A relational database would be a better choice
- If your data is in an aggregate-oriented design
  - If data naturally falls into a normalized tabular model
  - A relational database would be a better choice

Document-based NoSQL database examples

↳ most widespread of the NoSQL databases

IBM Cloudant | MongoDB | Apache CouchDB | Terrastore

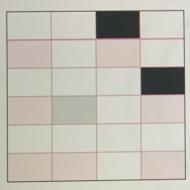
OrientDB | Couchbase | RavenDB

Column-Based NoSQL Databases

What you will learn

- Describe the architecture of the column-based NoSQL databases
- Describe technical advantages and disadvantages
- List suitable business uses

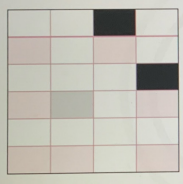
Column based NoSQL database architecture



- Spawned from Google's 'Bigtable'
- Bigtable clones, columnar, or wide-column databases
- Store data in columns or groups of columns

point

Column based NoSQL database architecture



- Column 'families' are several rows with unique keys belonging to one or more columns
  - Grouped in families as often accessed together
- Rows in a column family are not required to share the same columns
  - Can share all, a subset, or none
  - Columns can be added to any number of rows or not

Technical advantages: Suitable use cases

- Great for large amounts of sparse data
- Column databases better compress data and save storage space
- Handle deployment across clusters of nodes
- Used for event logging and blogs → each row can have its own columns and here each row key formatted to easy look up
- Counters are a unique use case for column databases
- Columns can have a time to live (TTL) parameter, making them useful for data with an expiration value like trial periods or ad timing

Technical disadvantages: Unsuitable use cases

- Traditional ACID transactions
  - Reads and writes are only atomic at the row level
- In early development, query patterns can change and require numerous changes to column-based databases
  - Can be costly
  - Can slow down the development timeline

Use cases: Corporate data warehousing

Use case: E-commerce

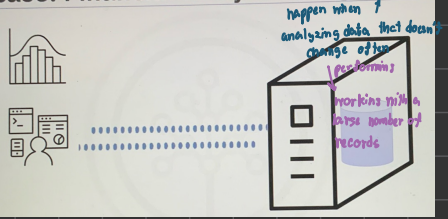
OrderID:
[1001, 1002]
CustomerID:
[C001, C002]
ProductID:
[P101, P102]
OrderDate:
[2023-09-01, 2023-09-02]
Quantity:
[2, 1]
TotalPrice:
{50.00, $30.00}

need

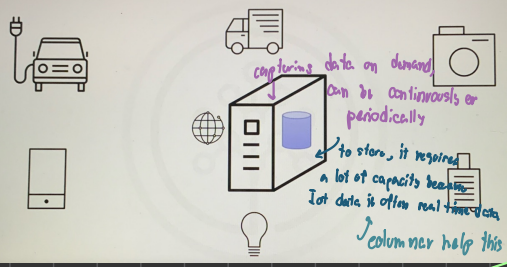
Query: Display the TotalPrice for all orders of Product ID P101

how DA can use columnar

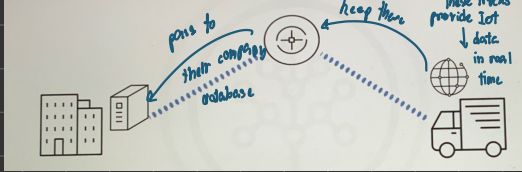
Use case: Financial analysis using OLAP



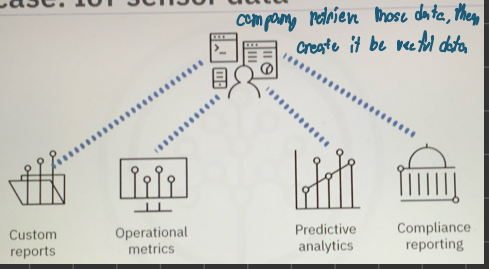
Use case: IoT sensor data



Use case: IoT sensor data



Use case: IoT sensor data

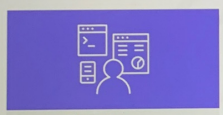


Popular NoSQL database vendors

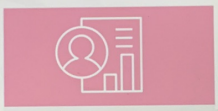
- Apache Cassandra
- Apache Hbase
- Hypertable
- Apache Accumulo

Graph NoSQL Databases

What you will learn



Describe the architecture of the Graph category of the NoSQL database



Explain the primary use cases for the Graph NoSQL database category

Graph NoSQL database architecture

- Graph databases store information in entities (or nodes) and relationships (or edges)
- Graph databases are impressive when your data set resembles a graph-like data structure

Traversing all of relationships is quic

Graph NoSQL database architecture

- Graph databases do not shard well
  - Traversing a graph with nodes split across multiple servers can become difficult and hurt performance
- Graph databases are ACID transaction compliant
  - Unlike other NoSQL databases discussed

Graph NoSQL database typical use cases

Cases for a Graph NoSQL database: powerful on

- For highly connected and related data
- Social networking
- Routing, spatial, and map apps
- Recommendation engines leverage the close relationships and links between products to easily provide other options to their produc

Graph NoSQL database unsuitable use cases

- When looking for advantages offered by other NoSQL database categories
- When an application needs to scale horizontally
  - You will quickly reach the limitations associated with these types of data stores
- When trying to update all or a subset of nodes with a given parameter
  - These types of operations can prove to be difficult and non-trivial

Graph NoSQL database example vendors

Neo4j	OrientDB	Arango DB
Amazon Neptune	Apache Giraph	JanusGraph

Glossary

Term	Definition
ACID	This term is an acronym for Atomicity, Consistency, Isolation, and Durability, which is a set of properties that guarantee reliable processing of database transactions in traditional relational databases.
Atomic	In the context of database transactions, atomic means that an operation is indivisible and either completes fully or is completely rolled back. It ensures that the database remains in a consistent state.
BASE	An alternative to ACID. Stands for basically available, soft state, eventually consistent. BASE allows for greater system availability and scalability, sacrificing strict consistency in favor of performance.
Bigtable	A NoSQL database system developed by Google, designed for handling large amounts of data and providing high performance, scalability, and fault tolerance.
Caching	The temporary storage of frequently accessed data in high-speed memory reduces the need to fetch the data from the primary storage, which can significantly improve response times.
Cluster	A group of interconnected servers or nodes that work together to store and manage data in a NoSQL database, providing high availability and fault tolerance.
Column database	A NoSQL database model that stores data in column families rather than tables, making it suitable for storing and querying vast amounts of data with high scalability. Examples include Apache Cassandra and HBase.
CRUD	CRUD is an acronym for create, read, update, and delete, which are the basic operations for the basic operations for interacting with and manipulating data in a database.
DBaaS	This acronym stands for database as a service, a cloud-based service that provides managed database hosting, maintenance, and scalability, allowing users to focus on application development without managing the database infrastructure.
Document	A NoSQL database model that stores data in semi-structured documents, often in formats like JSON or BSON. These documents can vary in structure and are typically grouped within collections.
Graph database	A NoSQL database model optimized for storing and querying data with complex relationships, represented as nodes and edges. Examples include Neo4j and OrientDB.
Horizontal scaling	The process of adding more machines or nodes to a NoSQL database to improve its performance and capacity. This is typically achieved through techniques like sharding.
Indexing	The creation of data structures that improve query performance by allowing the database to quickly locate specific records based on certain fields or columns.
JSON	JSON is an acronym for JavaScript Object Notation, a lightweight data-interchange format used in NoSQL databases and other data systems. JSON is human-readable and easy for machines to parse.
Key-value	A NoSQL database model that stores data as key-value pairs. It's a simple and efficient way to store and retrieve data where each key is associated with a value.
Normalized	A database design practice where data is organized to minimize redundancy and maintain data integrity by breaking it into separate tables and forming relationships between them.
NoSQL	NoSQL stands for "not only SQL." A type of database that provides storage and retrieval of data that is modeled in ways other than the traditional relational tabular databases.
Sharding	Refers to the practice of partitioning a database into smaller, more manageable pieces called shards to distribute data across multiple servers. Sharding helps with horizontal scaling.
TTL	Stands for "Time to Live," which is a setting in NoSQL databases that determines how long a piece of data should be retained before it's automatically removed from the database.
XML	Stands for Extensible Markup Language, another data interchange format used in some NoSQL databases. It's also human-readable and can represent structured data.