UNIT 3 - NOSQL Databases.

Introduction

- NoSQL databases are designed to manage large volumes of unstructured & semi-structured data
- offer more flexibility
- "not only SQL"
- Supports consistency rather than strict ACID properties of RDBMS
- generally avoids join operations.

History

- first used by Carlo Strozzi in 1998
- he mention this name for his open source Database System, where no provision of SQL query interface.
- 2009 NoSQL becomes in practice

Advantages

- good resource salability, low cost, support semi-structured data, no static schema, fast processing, etc.

Disadvantages

- not a defined standard
- Limited query capability.

Companies using NoSQL
- Google
- facebook
- LinkedIn
- Amazon.

Four Types of No-SQL Database.

1. Key-value store database
2. Document Database
3. Column-store database
4. Graph database

Key-Value store database

- Data is stored as pairs of keys & values.
- keys are unique and their corresponding values can be any type of data (string, JSON, binary, etc.)
- specially designed for storing data as schema-free data

Advantages :—
- Simple & efficient
- Highly scalable & fault tolerant

Disadvantages :—
- Limited query capability ie complex query is challenge
- for many-to-many relationship, it show poor performance

Examples :—
- Redis
- DynamoDB
- Cassandra.
- Azure Table Storage

Document stores database.

↳ Data is stored in documents that contain complex structures, including nested objects and arrays.

↳ formats like JSON or XML.

↳ each document contain nested structures & various data types, allowing for flexibility

Advantages :-

↳ excellent for managing semi-structured data

↳ Easy retrival & indexing of documents.

Disadvantages :-

↳ Handling multiple docs is complex

↳ Aggregate operations may not work accurately

Examples :-

MongoDB

CouchDB

Column store database / column-family Stores

↳ data is stored in columns rather than rows, organized into column family.

↳ Each column family contain rows identified by unique key, with each row having different set of columns.

Advantages :-

↳ Highly scalable & efficient

↳ good support for large query

Disadvantages :-

↳ limited support for ad-hoc & transactions queries

↳ complex in data modelling

Examples :-

↳ Apache Cassandra

↳ HBase

↳ Big Table

↳ Hyper Table

Graph Database :-

↳ Data is represented as nodes (entity) and edges (relationships), making it ideal for complex interconnected data

OR

Data is stored as graph and their relationships are stored as link between them whereas entity acts as a node.

Advantages :-

↳ Flexible model

↳ Effective for applications like social networks

Disadvantages :-

↳ may struggle with heavy write workloads

↳ limited support for adhoc query

↳ does not offer better choice over others.

Applications

↳ Neo 4j

↳ OrientDB

MongoDB

↳ open source document database management system.
↳ one of the popular NoSQL database, written in C++.
↳ use Binary JSON (BSON) to store documents, which allows complex data types & structures.
↳ Imp. features include High performance, High availability.
↳ Has its own ad-hoc query language
↳ As a cache memory, it automatically uses all free memory available on the machine.

Features
→ High performance
→ Good support for embedded data models.
→ Replication & High availability
→ Document oriented model
→ Schema less Design
→ Support horizontal scalability
→ Aggregation framework
→ Flexible Quering
→ Integration with Big Data Technologies

Architecture

Database & Collection - MongoDB organizes data into database containing multiple collections. Each collection hold multiple documents.
Documents - Each document is a self-contained unit with its own structure, represend as key-value pairs
BSON Format - Documents are stored in BSON format, which supports various datatypes like arrays & nested objects.

Use Cases
→ CMS (Content Management Systems)
→ Real Time Analytics
→ IOT
→ E-commerce Applications
→ Social Networks.

Limitations
→ High memory usage
→ Limited Nesting
→ Joins not supported
→ Limited Data Size.

Column-Oriented Database :—

Apache Cassandra

↳ Wide column store, which combines features of both key-value and tabular databases (ie, column oriented)
↳ designed to handle large amount of data across many servers with high availability and no single point of failure.

features
→ High Availability
→ Scalability
→ Partitioned Row Store
→ Consistency
→ CQL (Cassandra QL)
→ Flexible Schema

Use Cases
↳ Real time Analytics ↳ IOT Appl" ↳ Content management System.
↳ Social Media ↳ E-commerce

RDBMS & NOSQL database Comparision.

	RDBMS	NOSQL
①	Structured data in tables (rows & columns	Unstructured / semi-structured data
②	Pre defined schema; rigid structure.	Dynamic ~~structu~~ schema; flexible structure
③	Vertically scalable (add more power to existing hardware)	Horizontally scalable (add more power to servers)
④	SQL is standard language	Varies by database type.
⑤	Support complex transactions with ACID properties	simpler transactions; may not fully support ACID.
⑥	Optimized for complex queries & joins.	High performance for read/ write operations.
⑦	Strong data integrity	Weak data integrity.
⑧	Use case - banking system	Use case - social media, IOT
⑨	Eg. MySQL, Oracle, MS SQL Server	Eg. MongoDB, Cassandra, Redis, Couchbase

NOSQL Database Development Tools.

① Map Reduce
↳ A programming model used for processing & generating large datasets with a distributed algorithm on cluster.

Components :-
Map function - processes input data & produce key-value pairs.
Reduce function - aggregates results from Map function to produce a final output.

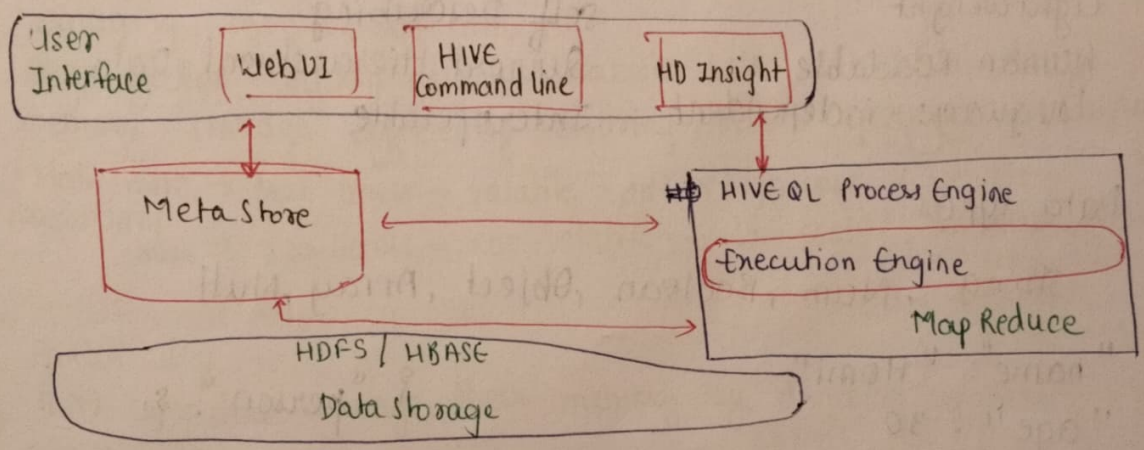
↳ used for batch processing of large datasets.
↳ used commonly in conjunction with Hadoop to analyze big data

Advantages -
↳ Scalable to handle vast amount of data across distributed systems.
↳ fault-tolerant.

HIVE

↳ data warehouse infrastructure tool.
↳ built on top of Hadoop.
↳ designed for querying & managing large datasets using SQL like interface called HiveQL.
↳ used to summarize big Data, analysis of big data.
↳ supports ad-hoc queries.
↳ SQL type scripts can be created for Mapreduce operations using HIVE

features :-
↳ SQL-like query language.
↳ Batch Processing.
↳ Schema management.
↳ scalable
↳ fast



Working from TextBook Pg 5-16.

XML - Extensible Markup Language

Store & share data in structured format.
looks like HTML but is more flexible

features → Tags : opening & closing tags
Hierarchy : data is stored in tree like structure
Attributes : extra info can be added to tags
Custom Tags : create your own tags

Eg:
```
<person>
    <name> John </name>
    <age > 30 < / age>
</person>
```

Types of XML schemas

DTD - Document Type Defn
XSD - XML Type Defn.

	DTD	XSD
Syntax	Non-XML	XML based
Datatype support	Basic	Extensive (numbers, dates)
Namespace support	No	Yes
	less flexible	more flexible

JSON (Java Script Object Notation)

stores & exchanges lightweight data
key value pairs

features :-

Lightweight
Human Readable
Language-independent
Self Describing
Support Hierarchical Data
Interoperable

Data types

String, Num, Boolean, Object, Array, Null

```
"name": "Homi"
"age" : 30
"isStudent": true
"address": {"city":"NY"}
"marks": [85,90,95]
"middlename": null.
```

```
{ "person": {
    "name": "Homi",
    "age" : 30 }
}
```