Nidhi Singh

Introduction to NO SQL & Columnar Database

~~1). Difference between SQL and NOSQL:-~~

★ NOSQL :-

- A non-relational database (no tables)
- stands for NOT ONLY SQL
- A flexible database used for big data & real-time web apps.

1).

★ Difference between SQL and NOSQL:

a). SQL databases are relational, NOSQL are non-relational in nature i.e. they do not consists any tables.

b). SQL databases have structured query language & have a predefined schema. NOSQL databases have dynamic schemas for unstructured data.

c). SQL databases are vertically scalable. NOSQL databases are horizontally scalable.

d). SQL databases are table based, while NOSQL databases are document, key-value, graph or wide-column stores.

c). SQL databases are better for multi-row transactions, NOSQL are better for unstructured data like documents or JSON.

★ Scaling of a Database:

- Scaling of a database means allowing the database to do more work than it was originally designed to, without taking a huge performance hit.
- It is the ability of a database ~~design~~ system to handle more work than it typically performs in an efficient way.

★ Different ways to scale a database:
There are two ways to scale a database:

- Horizontal Scaling
- Vertical Scaling

Horizontal Scaling: It means that we ~~add~~ scale the database by adding more machines into the pool of resources.

Vertical Scaling: It means that scaling is done by adding more power (CPU, RAM) to an existing machine.

★ RDBMS are vertically scalable. When load increase on RDBMS databases then we scale the database by increasing server hardware power.

Review 6

2).

* Different types of NoSQL databases:
There are 4 main types of NoSQL databases :-
  - Document databases
  - Key Value Stores
  - Column-Oriented databases
  - Graph Databases

a). Document Databases : In a document database, ~~documents~~ data are stored in JSON, BSON or XML documents. In this database:
- documents can be nested
- particular elements can be indexed for faster querying.

b). Key Value Stores :
- It is the simplest type of NoSQL database.
- Every data element in the dB is stored as a key value pair consisting of an attribute name & a value.
- It is like a relational dB with only two columns : key or attribute name & value

c). Column Oriented Databases :
- It is organised as set of columns.
- It can quickly aggregate the value of a given column.
- It makes reading faster without consuming memory with unwanted data.

d). Graph Databases:

- It focuses on the relationship between data elements.
- Each element is stored as a node.
- The connection b/w elements are known as links.
- It is optimized to capture & search the connections between data elements.

★ Advantages & Disadvantages of NOSQL:

* Advantages:-

- NOSQL is non-relational. Thus, they provide the ease of management while ensuring a high level of flexibility.
- NOSQL is low cost. It is an open-source dB that provides awesome sol^n for smaller enterprises.
- Scalability is easier. It is gaining popularity because of the elasticity & scalability it offers.

* Disadvantages:-

- less community support, as its new.
- Standardization - It lacks a standardized platform like SQL which prevents its further expanding.
- Interfaces & Interoperability:- It is another concern that is faced by NOSQL, which needs fixing immediately.

* When to Use NOSQL?
We should use NOSQL in following scenarios:

a). Constantly, new features are added.
b). When we are not concerned about data consistency.
c). When 100% data integerity is not our priority.
d). When we have a lot of data with many data types and it will grow over time.
e). When our data needs to scale up, out & down.

* When not to use NOSQL?

a). When we are working with complex queries & reports.
b). We have a high transaction applications.
c). When we need to ensure ACID property of the data Base i.e. Atomicity, Consistency, Durability, Isolation.
d). When we don't need a lot of changes or growth in our database.

3). Criteria to decide suitable DB for given application :-
Following are some of the major criteria to decide suitable DB :-

- check the data structure required.
- amount of data needed
- speed requirements
- data modeling to determine type of DB
- use of multiple databases
- throughput required.

4). Columnar databases :-

- Organised as set of columns.
- can quickly aggregate the value of a given column.
- It stores data tables column by column.

Row Oriented databases :-

- They are the traditional way of organizing data and provide key benefits for storing data quickly.
- It stores data table row by row.
- It is organised as set of rows.
- Storing of data is faster.

Explain how columnar database work:

- Columnar database stores data by columns. Thus, it is suitable for analytical query processing.
- They work to ~~get~~ provide the aggregate value of the given column.

5). Column-Oriented DBMS use column by column method to organise the data sets.
However, Row-Oriented DBMS ~~use~~ organise its data sets row by row.

Best Example of Row-Orient DBMS is Relational Database that stores data row-wise.

Best Example of Column-Oriented DBMS is HBase that stores its data column by column.

6). Applications where scaling of data comes into picture:

- Query optimization & connection pool implementation.
- Scale Up
- Command query Responsibility Segregation (CQRS)
- Multi primary replication.
- Partitioning, etc.

7). Different columnar - Oriented Database :-

- Maria DB
- Crate DB
- ClickHouse
- Greenplum DB
- Apache Hbase
- Apache Parquet
- Apache Kudu

* Explain Any two of them :-

a). Maria DB :-

- It is a powerful database server that is made from MySQL developers.
- It provides a platform for getting structured information from given data by use of a broad range of applications that range from websites to Banking.

b). ClickHouse :-

- It is an open-source column-oriented DBMS for online analytical processing.
- It is simple & works out of the box.
- It uses state-of-the-art algorithms.

* Why Column Oriented DB faster than Row Oriented

Column Oriented DB have faster query performance ~~based~~ because ~~of~~ the column design keeps the data closer together, which reduces seek time.
Thus, they are faster than row-oriented database.