1. NoSQL Databases

NoSQL (Not only SQL) is a type of database management and persistence system (DBMS).

NoSQL is an approach to database design that can accommodate a wide variety of data models, including key-value, document, columnar and graph formats. NoSQL, which stand for "not only SQL," is an alternative to traditional relational databases. NoSQL databases are especially useful for working with large sets of distributed data. Below are some key characteristics of NoSQL Databases:

- Does not use a relational model.
- Schema less i.e. no fixed schema.
- Open source.
- Does not use SQL as a querying language.
- Distributed fault tolerance architecture – runs very well on clusters.
- No joins needed.
- It does not replace RDBMS but compliments it.

2. Types of NoSQL Databases

Below are types of NoSQL databases:

- Key value Databases (DynamoDB).
- Column Family Databases (Big Table, HBase, Cassandra).
- Document Databases (CouchDB, MongoDB).
- Graph Databases (Neo4J).

- **Key value databases**: Data is stored in the form of key value pair. Key – Value is based on a hash table where there is a unique key and a pointer to a particular item of data (value).

Mappings are usually accompanied by cache mechanisms to maximize performance.
API is typically simple - implementation is often complex. The values are not queryable.
DynamoDB is an example of Key Value database.

Key	Value
Name	Gunjan Arora
Location	Faridabad

- **Column family databases**: The database key points to column families comprising of multiple columns.

Google's Big Table, HBase, Cassandra are some examples of Column family databases.

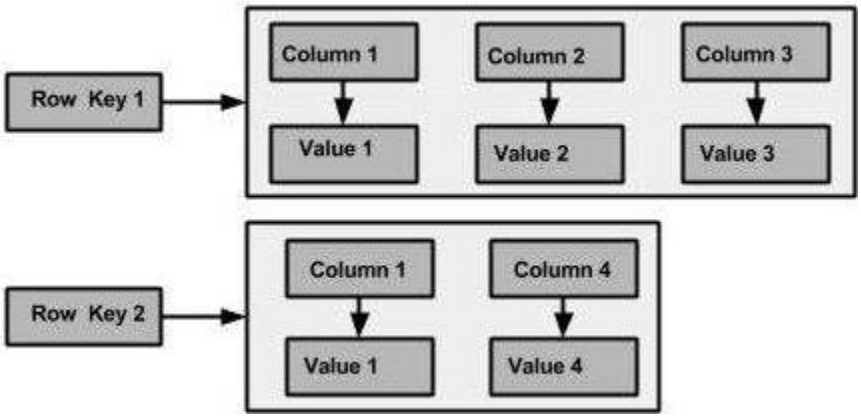

Fig 2.1: Column Family logical storage

- **Document based storage databases**: Documents are addressed in the database via a unique key that represents that document. The structure of documents can be XML, JSON or BSON formatted, for instance. In addition to the key, documents can be retrieved with queries within the values.

Example of Document based storage databases are CouchDB, MongoDB etc.

```
{
  name: "ABC",
  phone: 1234567890,
  address:
          {
           street: "1234 Some_XYZ Pkwy" ,
           Apt: 1001,
           City: "Pune",
           State: "Pune"
          }
}
```

- **Graph Storage Databases**: Graph Databases are built with nodes, relationships between nodes (edges) and the properties of nodes. Nodes represent entities (e.g. "Bob" or "Alice").

Similar in nature to the objects as in object-oriented programming.
Properties are pertinent information related to nodes (e.g. age: 18).
Edges connect nodes to nodes or nodes to properties.
Edges represent the relationship between the two nodes.

Example of Graph storage databases are Neo4J, FlockDB etc.

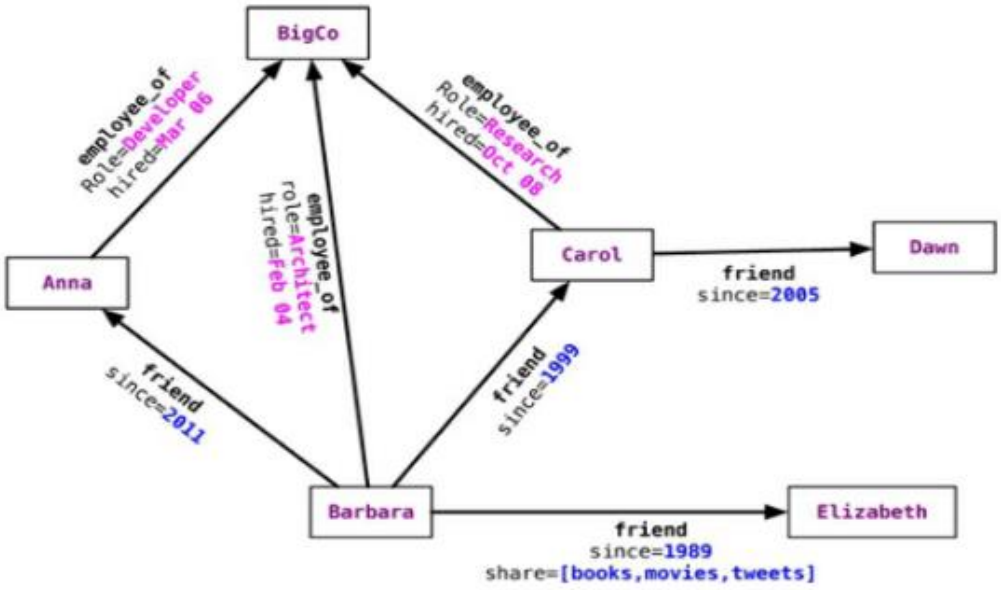

Fig 2.2: Graph based storage

3. CAP Theorem

CAP stands for **Consistency Availability Partition tolerance.** In theoretical computer science, the **CAP theorem**, also named Brewer's **theorem** after computer scientist Eric Brewer, states that it is impossible for a distributed data store to simultaneously provide more than two out of the following three guarantees: **Consistency, Availability, Partition tolerance**.

Consistency: Every read receives most recent write or an error.

Availability: Every request receives a (non-error) response – without guarantee that it contains the most recent write.

Partition tolerance: The system continues to operate despite an arbitrary number of messages being dropped (or delayed) by the network between nodes.

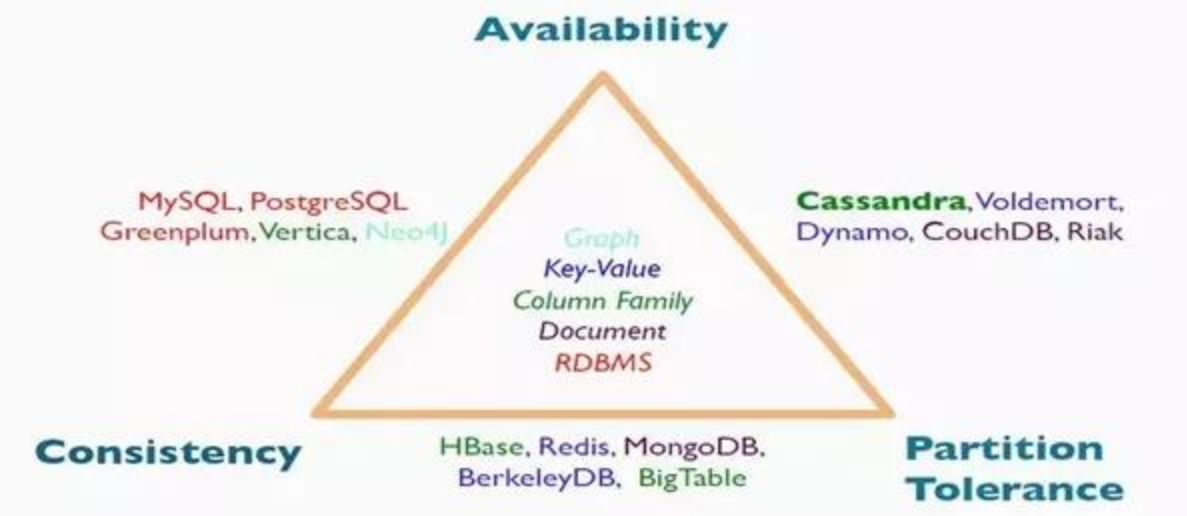

Fig 3.1CAP Theorem

4. HBase Architecture

● HBase has 3 major components, clients, HBase Master and Region Servers.

● Region Servers can be added or removed as per the requirement.

● When accessing data, clients connect to the Region Servers directly.

● Region Assignment, DDL creation (create, delete, updates) operations are handled by HBase Master Server process.

● Zookeeper, which is a part of HDFS, maintains a live cluster state.

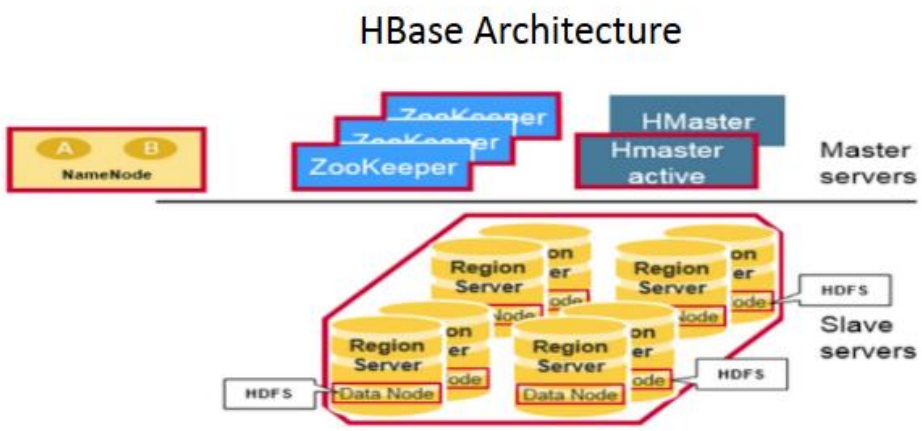

Fig 4.1 HBase Architecture

- HBase Tables are horizontally divided by row key range into what are called as "Regions".

- A region contains all rows in the table between the region's start key and end key.
- Regions are assigned to the nodes in the cluster on top of commodity machines and managed by Region Server Daemons. These daemons enable the data reads and writes.
- A region server can serve about 1000 regions.

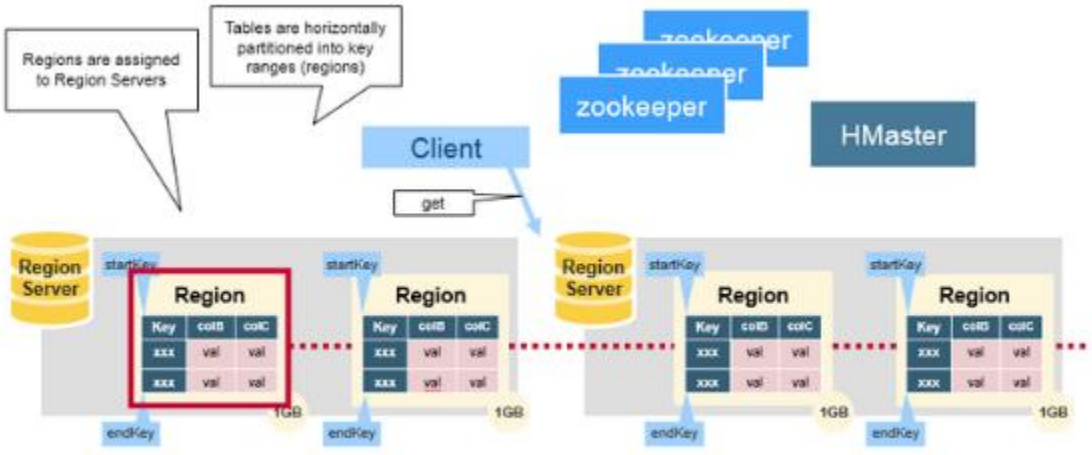

Fig 4.2 HBase Architecture – Regions and Regions Servers

5. HBase vs RDBMS

H Base	RDBMS
1. Column-oriented	1. Row-oriented(mostly)
2. Flexible schema, add columns on the Fly	2. Fixed schema
3. Good with sparse tables.	3. Not optimized for sparse tables.
4. No query language	4. SQL
5. Wide tables	5. Narrow tables
6. Joins using MR – not optimized	6. optimized for Joins(small, fast ones)
7. Tight – Integration with MR	7. Not really
8. De-normalize your data.	8. Normalize as you can
9. Horizontal scalability-just add hard war.	9. Hard to share and scale.
10. Consistent	10. Consistent
11. No transactions.	11. transactional
12. Good for semi-structured data as well as structured data.	12. Good for structured data.