

## Chapter 24 Summary: NoSQL and Big Data Storage Systems

NoSQL databases (short for “Not only SQL”) are made to withstand large amounts of data. They focus on speed, growth, availability, and flexibility, rather than simple rules for data consistency. These systems are distributed across multiple servers to balance load and maintain reliability.

There exist four main types of NoSQL databases:

1. Document-based databases (MongoDB, CouchDB) store data as documents, normally in JSON format. This allows for versatility and a change of schemas.
2. Key-value stores (Voldemort) store data in simple pairs: a key and its value. These are fast, scalable, and easy to replicate across multiple servers at once. For example, Voldemort uses consistent hashing to decide location, maintain reliability, and utilizes vector clocks to handle updates from multiple sources.
3. Column-based stores (BigTable, HBase) organize data into different columns, which condenses them into large datasets for easier understanding. HBase, for instance, stores its data in tables and columns, which keeps different versions of data, and uses systems like HDFS and Zookeeper to manage replication and storage.
4. Graph-based databases (Neo4j) represent data as nodes and edges, which helps with the exploration of relationships. Neo4j can also be schema-free or partially structured, which allows for a fair distribution, caching, and clustering.

NoSQL databases often use eventual consistency; in other words, they may not always present accurate up-to-date data, but they stay fast and reliable. These are commonly seen in social media, e-commerce, and mapping, which allow for the storage of massive data and quick access without strict rules.

### Real-World Application: Voldemort Key-Value Store:

Voldemort is a key-value NoSQL database formally used in LinkedIn to store vast amounts of data across its many servers. It shows us how NoSQL can handle speed and scalability in real-world applications.

#### How it works:

Voldemort first stores each piece of data or value with a unique key. Then, data is spread across different servers using the same consistent hashing, which eases the

search for data without having to look at every server. Each piece of data is stored on many servers to ensure maximum availability.

#### **Handling updates and failures:**

Servers can either go down or receive updates at the same time. Voldemort makes use of vector clocks, a system that uses multiple versions to track different forms of its own data. This way, a system can detect conflicts and allow them to be fixed by its applications. Servers that fail or respond slowly are blocked to ensure high performance.

#### **Why it matters:**

Voldemort is a great example of a distributed key-value store. Why? Because it balances load, keeps data intact, and allows for fast rewrites. This matches the chapter's idea of NoSQL being a scalable and flexible system for big data applications such as LinkedIn.

## **Chapter 25: Big Data Technologies Based on MapReduce and Hadoop:**

#### **Big Data Tech:**

Big data tech rose from Google's Google File System (GFS) and MapReduce. This helped with the distribution of petabyte-scale datasets. Hadoop is the open-source implementation, with HDFS, fault-tolerant storage, and batch processing of large data. It handles structured, semi-structured, and unstructured data efficiently while supporting different formats (CSV and XML formats).

#### **MapReduce:**

MapReduce is a programming model that divides tasks into map and reduce for cluster processing. Higher-level tools like Pig and Hive help simplify analytics and complex jobs. Frameworks such as Apache Tez optimize these and allow for unnecessary HDFS writes while improving SQL queries and Pig scripts.

#### **Hadoop v2/ Yarn:**

Hadoop v2 separates resource management from application execution. This allows multiple frameworks to run on the same cluster while optimizing workflow and large-scale processing. Yarn transforms Hadoop into a flexible platform for batch processing, streaming, and graph analytics.

#### **Big Data in Cloud Computing:**

Cloud computing enhances big data's ability to provide elastic and globally distributed sources. These services allow scalability on demand while reducing costs and enabling rapid deployment for large-scale analytics. The combination of cloud

storage and computing with local processes reduces network overload and improves performance.

### **Yarn as a Data Service Platform:**

YARN enables multiple analytic services to exist at once while leveraging HDFS data locality. Aside from MapReduce, services like Spark or Storm support machine learning, graph processing, and SQL analytics. This unified approach helps reduce static resource allocation issues and eases the deployment of various data applications.

## **Real-World Application: Netflix Hadoop Genie**

Netflix uses Hadoop to process massive amounts of streaming and user data, and Genie is the lead management system used to handle these workloads at scale. It shows us how big data (Hadoop) can be applied to a real-world scenario to handle speed, scalability, and advanced data processing.

### **How it works:**

Netflix makes use of Hadoop clustering, which is used to run cloud data and tailor different types of workloads. Genie is then used to match different jobs from data scientists to then appropriately cluster and handle the technical details of Hadoop, Hive, and Pig jobs. This allows users to only focus on analysis rather than plain infrastructure.

### **Handling updates and failures:**

Clusters can often take too long or fail to allocate resources appropriately. Thus, the genie is in charge of tracking metadata, ensuring jobs are executed, and maintaining reliability.

### **Why it matters:**

Netflix Hadoop is a perfect example of how distributed data (Hive and Pig) processing can be applied in the real world. It emphasizes key chapter 25 concepts: scalable storage, parallel processing, and handling big data. By automating these, Genie can extract big data insights at a huge scale while maintaining its reliability.