# **Normalization**

Normalization is a database design technique that organizes tables in a manner that reduces redundancy and dependency of data. Normalization divides larger tables into smaller tables and links them using relationships.

The purpose of Normalization is to eliminate redundant (useless) data and ensure data is stored logically.

|  |  |  |  |
| --- | --- | --- | --- |
| student\_id | subject\_id | marks | subject\_name |
| 1 | 1 | 80 | Database |
| 2 | 1 | 70 | Database |
| 1 | 2 | 90 | Java |

Database Normal Forms

Database Normalization

Un-normalized: There are multivalued attributes or repeating groups

1 NF – No multivalued attributes or repeating groups.

2 NF – 1 NF plus no partial dependencies

3 NF – 2 NF plus remove non-Key dependencies or transitive dependencies

|  |  |
| --- | --- |
| UNF | UNF Level |
| BillNo | 1 |
| Date | 1 |
| CustomerCode | 1 |
| CustomerName | 1 |
| ItemCode | 2 |
| ItemName | 2 |
| Rate | 2 |
| Qty | 2 |

**Remove Repeating Group Information**

* BillNo
* Date
* CustomerCode
* CustomerName
* BillNo
* ItemCode
* ItemName
* Rate
* Qty

**Remove Partial Key Dependencies**

* BillNo
* Date
* CustomerCode
* CustomerName
* ItemCode
* ItemName
* Rate
* BillNo
* ItemCode
* Qty

**Remove Non-Key Dependencies or Transitive Dependencies**

* BillNo
* Date
* CustomerCode
* CustomerCode
* CustomerName
* ItemCode
* ItemName
* Rate
* BillNo
* ItemCode
* Qty

# **Database**

A database is a systematic collection of data. They support electronic storage and manipulation of data. Databases make data management easy.

Let us discuss a database example: An online telephone directory uses a database to store data of people, phone numbers, and other contact details. Your electricity service provider uses a database to manage billing, client-related issues, handle fault data, etc.

Let us also consider Facebook. It needs to store, manipulate, and present data related to members, their friends, member activities, messages, advertisements, and a lot more. We can provide a countless number of examples for the usage of databases.

**Database Management System (DBMS)** is a collection of programs that enable its users to access databases, manipulate data, report, and represent data. It also helps to control access to the database.

Databases are used just about everywhere including banks, retail, websites and warehouses. Banks use databases to keep track of customer accounts, balances and deposits. Retail stores can use databases to store prices, customer information, sales information and quantity on hand. Websites use databases to store content, customer login information and preferences and may also store saved user input. Warehouses use databases to manage inventory levels and storage location. Databases are used anywhere that data needs to be stored and easily retrieved. The filing cabinet has all but been replaced by databases.

Because databases are stored digitally, multiple users in different locations can view the data in more than once place. Because banks store their customer information and balances in a database, you can use any branch for deposits and withdrawals. Databases allow more flexibility because they are in a digital format. Companies use databases for inventory and item pricing. A retail chain can see when stores are low in inventory and automatically order more. Prices can be updated across the country instantly as compared to having to manually do it at each store. Databases are used to distribute data quickly and easily because they are only updated once and can be read by many users.

# **SQL & No-SQL**

1. SQL databases are relational, NoSQL databases are non-relational.
2. SQL databases use structured query language and have a predefined schema. NoSQL databases have dynamic schemas for unstructured data.
3. SQL databases are vertically scalable, while NoSQL databases are horizontally scalable.
4. SQL databases are table-based, while NoSQL databases are document, key-value, graph, or wide-column stores.
5. SQL databases are better for multi-row transactions, while NoSQL is better for unstructured data like documents or JSON.

# **Big Data**

Big data is the term for collection of data sets so large and complex that it becomes difficult to process using on-hand database system tools or traditional data processing applications.

5’s V of big data

**Volume**

Volume, the first of the 5 V's of big data, refers to the amount of data that exists. Volume is like the base of big data, as it is the initial size and amount of data that is collected. If the volume of data is large enough, it can be considered big data. What is considered to be big data is relative, though, and will change depending on the available computing power that's on the market.

**Velocity**

The next of the 5 V's of big data is velocity. It refers to how quickly data is generated and how quickly that data moves. This is an important aspect for companies need that need their data to flow quickly, so it's available at the right times to make the best business decisions possible.

An organization that uses big data will have a large and continuous flow of data that is being created and sent to its end destination. Data could flow from sources such as machines, networks, smartphones or social media. This data needs to be digested and analyzed quickly, and sometimes in near real time.

As an example, in healthcare, there are many medical devices made today to monitor patients and collect data. From in-hospital medical equipment to wearable devices, collected data needs to be sent to its destination and analyzed quickly.

In some cases, however, it may be better to have a limited set of collected data than to collect more data than an organization can handle -- since this can lead to slower data velocities.

**Variety**

The next V in the five 5 V's of big data is variety. Variety refers to the diversity of [data types](https://searchapparchitecture.techtarget.com/definition/data-type). An organization might obtain data from a number of different data sources, which may vary in value. Data can come from sources in and outside an enterprise as well. The challenge in variety concerns the standardization and distribution of all data being collected.

Collected data can be unstructured, semi-structured or structured in nature. Unstructured data is data that is unorganized and comes in different files or formats. Typically, unstructured data is not a good fit for a mainstream [relational database](https://searchdatamanagement.techtarget.com/definition/relational-database) because it doesn't fit into conventional data models. Semi-structured data is data that has not been organized into a specialized [repository](https://searchoracle.techtarget.com/definition/repository) but has associated information, such as [metadata](https://whatis.techtarget.com/definition/metadata). This makes it easier to process than unstructured data. Structured data, meanwhile, is data that has been organized into a formatted repository. This means the data is made more addressable for effective data processing and analysis.

**Veracity**

Veracity is the fourth V in the 5 V's of big data. It refers to the quality and accuracy of data. Gathered data could have missing pieces, may be inaccurate or may not be able to provide real, valuable insight. Veracity, overall, refers to the level of trust there is in the collected data.

Data can sometimes become messy and difficult to use. A large amount of data can cause more confusion than insights if it's incomplete. For example, concerning the medical field, if data about what drugs a patient is taking is incomplete, then the patient's life may be endangered.

Both value and veracity help define the quality and insights gathered from data.

**Value**

The last V in the 5 V's of big data is value. This refers to the value that big data can provide, and it relates directly to what organizations can do with that collected data. Being able to pull value from big data is a requirement, as the value of big data increases significantly depending on the insights that can be gained from them.

Organizations can use the same big data tools to gather and analyze the data, but how they derive value from that data should be unique to them.

A screenshot of a cell phone

Description automatically generated

# **Hadoop, HDFS, MapReduce**

Hadoop is a framework that allows us to store and process large data sets in parallel and distributed fashion. HDFS is the storage layer of Hadoop Ecosystem, while MapReduce is the processing layer of the ecosystem. All the data in Hadoop is stored in Hadoop Distributed File System. It has 3 main components that are Name node, Secondary Name node and Slavenode.

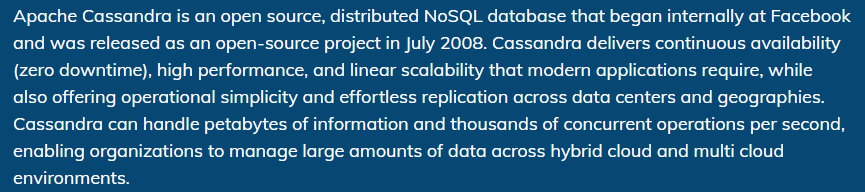
A close up of a logo

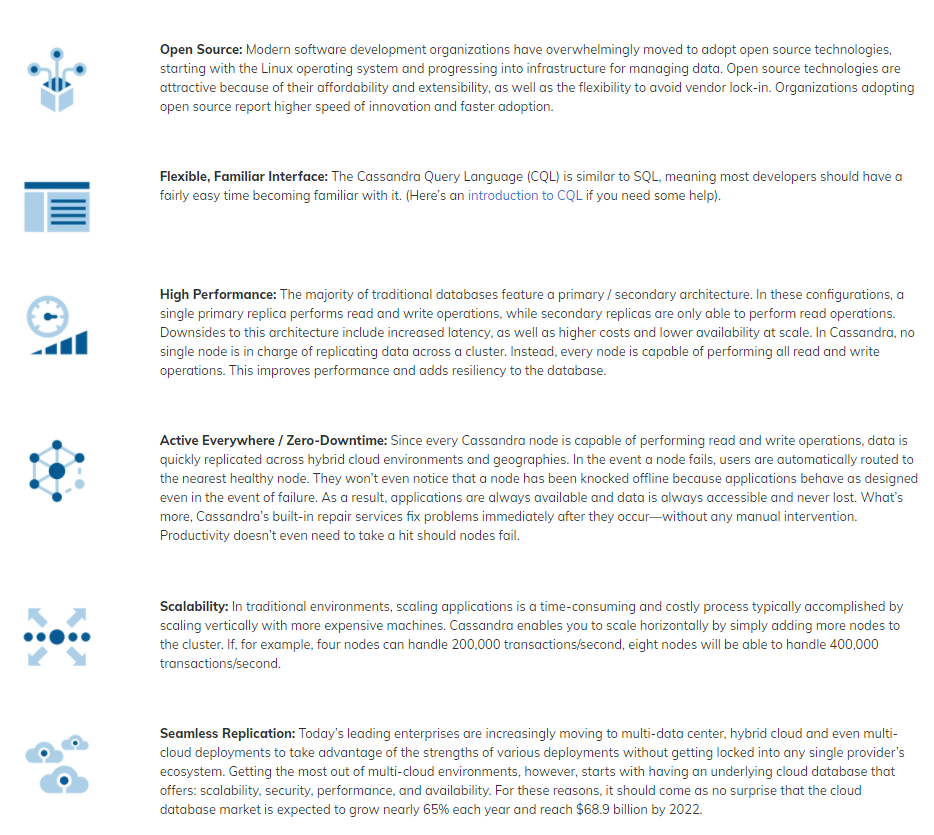
Description automatically generated

# **MongoDB and Graph DB Difference**

* Both Neo4j and MongoDB are NoSQL databases. However, Neo4J is a graph database.
* MongoDB is schemeless and provides holistic view of the data with eventual consistency with update information-in-place principle.
* MongoDB has aggregate framework, and Neo4J does not have aggregate framework.
* Graph database like Neo4J comply ACID framework with complete fine-grained atomicity of the transactions.
* MongoDB provides flexibility, high scalability in a distributed environment. Therefore, it requires eventual consistency of the database than instant ACID compliance.
* MongoDB does not create relationships between the database models, as each data set stored in the document store of the database is disaggregated and independent. A graph system requires handling the complex relationship of the database, while NoSQL database does not require handling the complex associations between the data models.
* Neo4J enables navigation through the graphs as a tree, where MongoDB cannot provide visualization of the document stores as graphs.

# **Casandra**





# **Row vs Column Oriented**

|  |  |  |
| --- | --- | --- |
| | Row oriented data stores | | --- | | Column oriented data stores |
| Data is stored and retrieved one row at a time and hence could read unnecessary data if some of the data in a row are required. | In this type of data stores, data are stored and retrieve in columns and hence it can only able to read only the relevant data if required. |
| Records in Row Oriented Data stores are easy to read and write. | In this type of data stores, read and write operations are slower as compared to row-oriented. |
| Row-oriented data stores are best suited for online transaction system. | Column-oriented stores are best suited for online analytical processing. |
| These are not efficient in performing operations applicable to the entire datasets and hence aggregation in row-oriented is an expensive job or operations. | These are efficient in performing operations applicable to the entire dataset and hence enables aggregation over many rows and columns. |
| Typical compression mechanisms which provide less efficient result than what we achieve from column-oriented data stores. | These type of data stores basically permits high compression rates due to little distinct or unique values in columns. |