Unit 10

Emerging Trend in Databases

NoSQL Databases

- ✓ NoSQL stands for Not only SQL.
- ✓ NoSQL is a non-relational database that is used to store the data in the nontabular form.
- ✓ It is designed to handle and store large volumes of unstructured and semi-structured data.
- ✓ Unlike traditional relational databases that use tables with pre-defined schemas to store data, NoSQL databases use flexible data models that can adapt to changes in data structures and are capable of scaling horizontally to handle growing amounts of data.

Categories of NoSQL

NoSQL databases are generally classified into four main **categories:**

1. **Document based databases:** The document-based database is a non-relational database. Instead of storing the data in rows and columns (tables), it uses the documents to store the data in the database. A document database stores data in JSON, BSON, or XML documents.

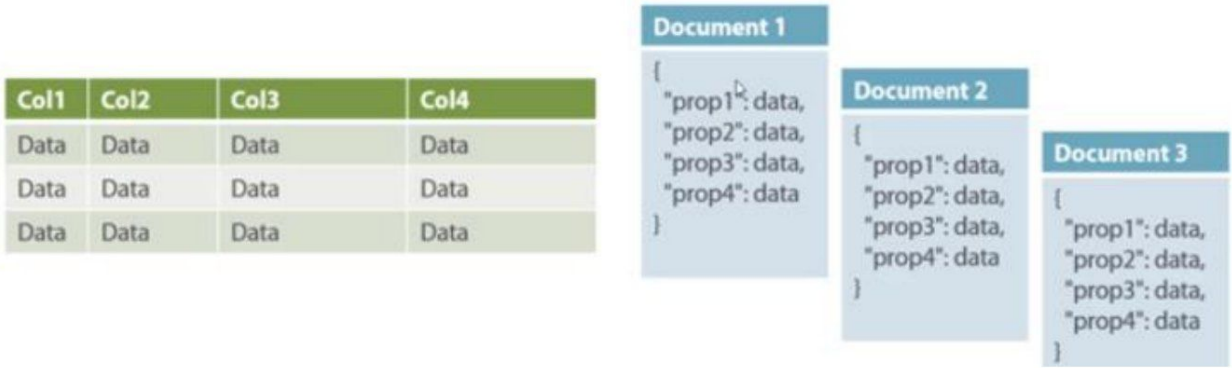


Figure: relational vs document

- ✓ In above diagram, left side we can see that we have rows and columns, and in the right, we have a document database which has a similar structure to JSON.
- ✓ Documents in the database has a flexible schema. It means the documents in the database need not be the same schema.

2. **Key-value stores:**
   - ✓ Every data element in the database is stored in key-value pairs.
   - ✓ The data can be retrieved by using a unique key allotted to each element in the database. The values can be simple data types like strings and numbers or complex objects.
   - ✓ Redis, Dynamo, Riak are some NoSQL examples of key-value store Databases.
   - ✓ For example, a key-value pair may contain a key like "Name" associated with a value like "Ramesh".

Key	Value
Name	Ramesh
Address	Kathmandu
Age	30
Occupation	Teacher

3. **Column-Oriented databases**

   These databases store data in columns instead of rows and are optimized for managing large amounts of data.

   That means when we want to run analytics on a small number of columns, you can read those columns directly without consuming memory with the unwanted data.

   They deliver high performance on aggregation queries like SUM, COUNT, AVG, MIN etc. as the data is readily available in a column.

   HBase, Cassandra, HBase, Hypertable are NoSQL query examples of column based database.

ColumnFamily			
Row Key	Column Name		
	Key	Key	Key
	Value	Value	Value
	Column Name		
	Key	Key	Key
	Value	Value	Value

Column based NoSQL database

Example:

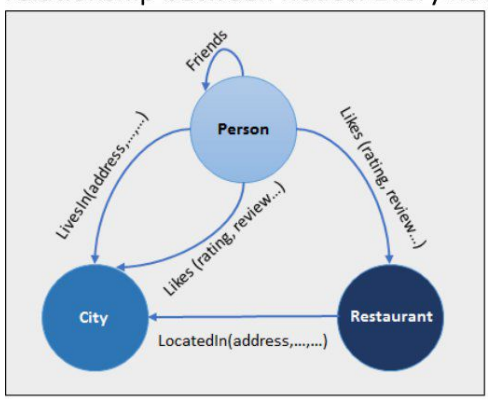
Row oriented (Relational)

Students		
ID	**First name**	**Last name**
1	Luna	Lovegood
2	Hermione	Granger
3	Ron	Weasley

Column oriented

Students		
ID	**First name**	**Last name**
1	Luna	Lovegood
2	Hermione	Granger
3	Ron	Weasley

4. **Graph databases:** A graph type database stores entities as well the relations amongst those entities. The entity is stored as a node with the relationship as edges. An edge gives a relationship between nodes. Every node and edge has a unique identifier.



- ✓ Graph based database mostly used for social networks.
- ✓ Neo4J, Infinite Graph, OrientDB, FlockDB are some popular graph-based databases.

Characteristics of No SQL

Although there are different ways that can be incorporated to understand how NoSQL databases work, we will now look at some of the most common features that define a basic NoSQL database.

- ❖ **Complex-free working:** Unlike SQL databases, NoSQL databases are not complicated. They store data in an unstructured or a semi-structured form that requires no relational or tabular arrangement. Perhaps they are easier to use and can be accomplished by all.
- ❖ **Independent of Schema:** NoSQL databases are independent of schemas which implies that they can be run over without any predetermined schemas.
- ❖ **Better Scalability:** NoSQL databases scale easily, meaning they can grow with data demands. They're also flexible in how they store data, making it easier to adapt to changing needs. This makes them ideal for managing diverse and large datasets effectively.
- ❖ **Flexible to accommodate:** Since such databases can accommodate heterogeneous data that requires no structuring, they are claimed to be flexible in terms of their usage and reliability.
- ❖ **Durable:** NoSQL databases are highly durable as they can accommodate data ranging from heterogeneous to homogeneous. They can also incorporate unstructured data that requires no query language. Undoubtedly, these databases are durable and efficient.

Advantages of NoSQL:

There are many advantages of working with NoSQL databases such as MongoDB and Cassandra. The main advantages are high scalability and high availability.

- **Scalability:** NoSQL databases are highly scalable, which means that they can handle large amounts of data and traffic with ease.
- **Flexibility**: NoSQL databases are designed to handle unstructured or semi-structured data, which means that they can accommodate dynamic changes to the data model.
- **High availability**: The auto, replication feature in NoSQL databases makes it highly available because in case of any failure data replicates itself to the previous consistent state.
- **Performance:** NoSQL databases are designed to handle large amounts of data and traffic, which means that they can offer improved performance compared to traditional relational databases.
- **Cost-effectiveness**: NoSQL databases are often more cost-effective than traditional relational databases, as they are typically less complex and do not require expensive hardware or software.

Disadvantages of NoSQL

NoSQL has the following disadvantages.

- **Lack of standardization:** There are many different types of NoSQL databases, each with its own unique strengths and weaknesses. This lack of standardization can make it difficult to choose the right database for a specific application.
- **Lack of ACID compliance:** NoSQL databases are not fully ACID-compliant, which means that they do not guarantee the consistency, integrity, and durability of data. This can be a drawback for applications that require strong data consistency guarantees.
- **Narrow focus:** NoSQL databases have a very narrow focus as it is mainly designed for storage but it provides very little functionality. Relational databases are a better choice in the field of Transaction Management than NoSQL.
- **Lack of support for complex queries**: NoSQL databases are not designed to handle complex queries, which means that they are not a good fit for applications that require complex data analysis or reporting.
- **Lack of maturity:** NoSQL databases are relatively new and lack the maturity of traditional relational databases. This can make them less reliable and less secure than traditional databases.
- **Management challenge:** The purpose of big data tools is to make the management of a large amount of data as simple as possible. But it is not so easy. Data management in NoSQL is much more complex than in a relational database.
- **Large document size:** Some database systems like MongoDB and CouchDB store data in JSON format. This means that documents are quite large (BigData, network bandwidth, speed).

Object Relational Mapping (ORM)

- ✓ Object-relational mapping is a technique that maps objects onto a relational database, converting the object into data that can be stored, retrieved and reconstructed when needed.
- ✓ In another way, we can see the ORM as the layer that connects object oriented programming (OOP) to relational databases.
- ✓ Through ORM, changes made to an object are shared with the database, which then updates the data to reflect these changes.

Why use an ORM(Object- Relational Mapping)?

- ✓ Databases and object-oriented programs use different programming languages and store data differently, which can lead to communication difficulties between languages.
- ✓ ORM maps objects onto the relational database table so that when we save an object to the database, it's broken down into smaller parts that the database can store. These parts are then saved in a logical order. When we access the object again, the program can retrieve the parts from the database to reconstruct the object.

How Does Object-Relational Mapping Work?

- ✓ ORM produces a structured map that reveals the relationships between objects and tables, or data, without getting into the details of the data structure. Objects are transformed into simpler, manageable pieces that the database can easily process and store for later retrieval.
- ✓ ORM connects object-oriented programming languages or applications with a relational database, communicating any changes made to an object to the database, which then alters the data accordingly.

Object-Relational Mapping Example

Imagine a dog model (i.e. class). Each dog object has the following attributes: name, age and breed. Here is how the dog class and dog objects look in Java.

```java
public class Dog {

  private String name;
  private String age;
  private String breed;

  public Dog(String name, String age, String breed) {

    this.name = name;
    this.age = age;
    this.breed = breed;
  }

  public String getName() {
    return name;
  }

  public String getAge() {
    return age;
  }

  public String getBreed() {
    return breed;
  }

  public void setName(String name) {
    this.name = name;
  }

  public void setAge(String age) {
    this.age = age;
  }

  public void setBreed(String breed) {
    this.breed = breed;
  }

}
```

A Java class that defines a dog by its name, age and parameter.

To store all of these dog objects in the database, we create a table to represent the dog class. The table rows correspond to dog instances and the columns correspond to the dog class' attributes. When we create a dog record in the table, it's automatically assigned an ID value.

ID	Name	Age	Breed
1	Bailey	5	Labrador Retriever
2	Max	8	Poodle
3	Charlie	3	Boxer
4	Luna	10	Husky
5	Rocky	4	German Shepard

The table stores the attributes of the different dog classes.

Advantages of Object-Relational Mapping

1. **Handles the logic required to interact with database:** ORM tools automatically generate the necessary SQL queries and handle database transactions, so developers don't need to write SQL code themselves.
2. **Speed up development time for teams:** ORM tools abstract away the low-level details of working with a database, which allows developers to focus on writing business logic instead of SQL queries. This shift results in faster development times because developers can write and test code more quickly.
3. **Decreases the cost of development**: Because ORM tools automate many of the repetitive tasks involved in working with databases, developers can write more code in less time, thereby leading to a lower cost of development.
4. **Improves security:** ORM tools provide built-in security features that help prevent SQL injection attacks, which are a common security vulnerability in database-driven applications.
5. **Requires less code:** ORM tools allow developers to interact with databases using a higher-level programming language like Java or Python instead of writing SQL queries. This makes the code easier for other team members to read; the code is also easier to maintain because developers don't need to write as much SQL code.

Disadvantages of Object-Relational Mapping

1. **Steep learning curve**

- ✓ ORM tools can have a steep learning curve, especially for developers who are not familiar with object-oriented programming concepts.
- ✓ Developers have to understand how to map their object-oriented code to the database schema, which can require a significant amount of time and effort to learn.
- ✓ Additionally, some ORM tools have their own specific syntax and APIs, which developers must learn to use.

2. **Struggles with complex queries**

- ✓ ORM tools simplify working with databases, but they might not be as good as writing SQL directly, especially for complex queries.
- ✓ Some ORM tools don't support advanced SQL features, which can be a limitation for certain applications.

3. **Slow performance speed**

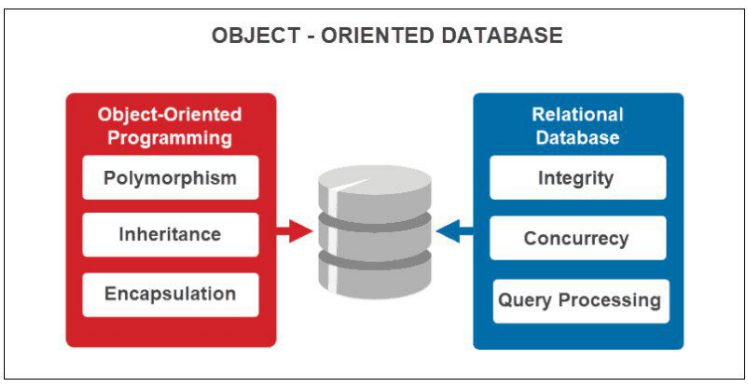
- ✓ ORM tools generate SQL queries automatically, which can result in queries that are less efficient than those written by an experienced SQL developer. This lack of efficiency can result in slower performance for database-driven applications.
- ✓ Additionally, ORM tools often generate more database queries than necessary, which can slow down performance even further.

ORM tools

- ➢ ***JAVA:*** Hibernate, Apache OpenJPA, EclipseLink, jOOQ, Oracle TopLink
- ➢ ***PYTHON:*** Django, web2py, SQLObject, SQLAlchemy

Object Oriented Databases

- ✓ Object-oriented databases add database functionality to object programming languages, creating more manageable code bases.
- ✓ An object-oriented database is managed by an object-oriented database management system (OODBMS). The database combines object-oriented programming concepts with relational database principles.
- ✓ Objects are the basic building block and an instance of a class, where the type is either built-in or user defined.
- ✓ Classes provide a schema or blueprint for objects, defining the behavior.
- ✓ Methods determine the behavior of a class.
- ✓ Pointers help access elements of an object database and establish relations between objects.



Object-Oriented Programming Concepts

Object-oriented databases closely relate to object-oriented programming concepts. The four main ideas of object-oriented programming are:

- ➢ Polymorphism
- ➢ Inheritance
- ➢ Encapsulation
- ➢ Abstraction

Example of Object-oriented databases

- ➢ GemStone/S
- ➢ ObjectDB
- ➢ ObjectDatabase++
- ➢ Objectivity/DB
- ➢ ObjectStore

Advantages

- ✓ Complex data and a wider variety of data types compared to MySQL data types.
- ✓ Easy to save and retrieve data quickly.
- ✓ Seamless integration with object-oriented programming languages.
- ✓ Easier to model the advanced real-world problems.
- ✓ Extensible with custom data types.

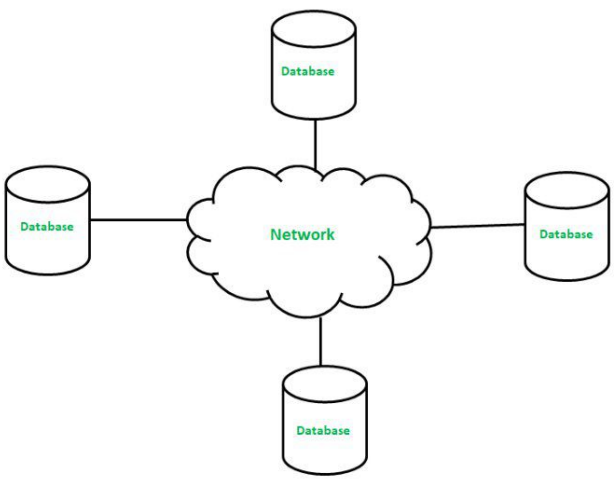
Disadvantages

- ✓ Not as widely adopted as relational databases.
- ✓ No universal data model. Lacks theoretical foundations and standards.
- ✓ Does not support views.
- ✓ High complexity causes performance issues.
- ✓ An adequate security mechanism and access rights to objects do not exist.

Distributed Databases

A distributed database is basically a type of database which consists of multiple databases that are connected with each other and are spread across different physical locations. The data that is stored in various physical locations can thus be managed independently of other physical locations. The communication between databases at different physical locations is thus done by a computer network.

This may be required when a particular database needs to be accessed by various users globally. It needs to be managed such that for the users it looks like one single database.



Types:

1. Homogeneous Database:

In a homogeneous database, all different sites store databases identically. The operating system, database management system, and the data structures used – all are the same at all sites. Hence, they're easy to manage.

2. Heterogeneous Database:

In a heterogeneous distributed database, different sites can use different schema and software that can lead to problems in query processing and transactions. Also, a particular site might be completely unaware of the other sites. Different computers may use a different operating system, different database application. They may even use different data models for the database.

Distributed Data Storage

There are two ways in which data can be stored at different sites. These are,

i) **Replication:** As the name suggests, the system stores copies of data at different sites. If an entire database is available on multiple sites, it is a fully redundant database.

ii) **Fragmentation**

In Fragmentation, the relations are fragmented, which means they are split into smaller parts. Each of the fragments is stored on a different site, where it is required. In this, the data is not replicated, and no copies are created.

The prerequisite for fragmentation is to make sure that the fragments can later be reconstructed into the original relation without losing any data.

There are two types of fragmentation,

Horizontal Fragmentation – Splitting by rows.

Vertical fragmentation – Splitting by columns.

- ✓ **Horizontal Fragmentation**

The relation schema is fragmented into group of rows, and each group is then assigned to one fragment.

- ✓ **Vertical Fragmentation**

The relation schema is fragmented into group of columns, called smaller schemas. These smaller schemas are then assigned to each fragment.

Each fragment must contain a common candidate key to guarantee a lossless join.

Advantages of Distributed Database

Better Reliability: Distributed databases offers better reliability than centralized databases. When database failure occurs in a centralized database, the system comes to a complete stop. But in the case of distributed databases, the system functions even when a failure occurs, only performance-related issues occur which are negotiable.

Modular Development: It implies that the system can be expanded by adding new computers and local data to the new site and connecting them to the distributed system without interruption.

Lower Communication Cost: Locally storing data reduces communication costs for data manipulation in distributed databases. In centralized databases, local storage is not possible.

Better Response Time: As the data is distributed efficiently in distributed databases, this provides a better response time when user queries are met locally. While in the case of centralized databases, all of the queries have to pass through the central machine which increases response time.

Disadvantages of Distributed Database

Costly Software: Maintaining a distributed database is costly because we need to ensure data transparency, coordination across multiple sites which requires costly software.

Large Overhead: Many operations on multiple sites require complex and numerous calculations, causing a lot of processing overhead.

Improper Data Distribution: If data is not properly distributed across different sites, then responsiveness to user requests is affected. This in turn increases the response time.

Distributed Ledger Technology

- ✓ Distributed Ledger Technology (DLT) is also known as a "shared ledger" or simply distributed ledger.
- ✓ It is a digital system that lets users and systems record transactions related to assets. A distributed ledger technology stores the information at multiple locations at any given point of time.
- ✓ DLT, unlike traditional databases, does not have any central place to store information. This is what differentiates it from a traditional database.
- ✓ At the core, DLT originates from the peer-to-peer(P2P) network. In any P2P network, peers communicate with each other without the need for a centralized entity. Technically, distributed ledger technology is possible through a peer-to-peer network.

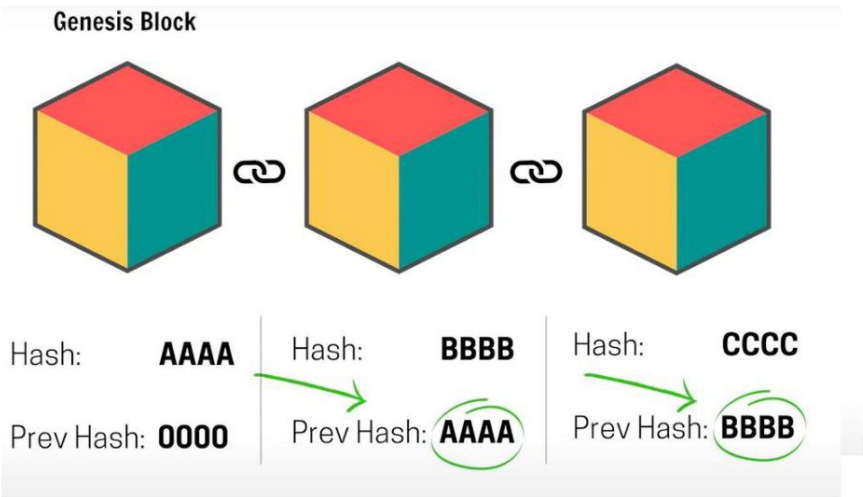
Key features of distributed ledger technology:

- ➢ **Decentralized:** It is a decentralized technology and every node will maintain the ledger, and if any data changes happen, the ledger will get updated. The process of updating takes place independently at each node. Even small updates or changes made to the ledger are reflected and the history of that change is sent to all participants in a matter of seconds.
- ➢ **Immutable:** Distributed ledger uses cryptography to create a secure database in which data once stored cannot be altered or changed. In this process, every node or contributor of the Ledger will try to verify the transactions with the various consensus algorithms or voting. The voting or participation of all the nodes depends on the rules of that Ledger. With Bitcoin, the Proof of Work consensus mechanism is used for the participation of each node.
- ➢ **Distributed:** This technology doesn't rely on a single central authority or server to manage its database, which makes it very open and transparent.
- ➢ **Fault Tolerance:** Distributed ledgers are highly fault-tolerant because of their decentralized nature. If one node or participant fails, the data remains available on other nodes.
- ➢ **Security:** Distributed ledgers are highly secure because of their cryptographic nature. Every transaction is recorded with a cryptographic signature that ensures that it cannot be altered. This makes the technology highly secure and resistant to fraud.

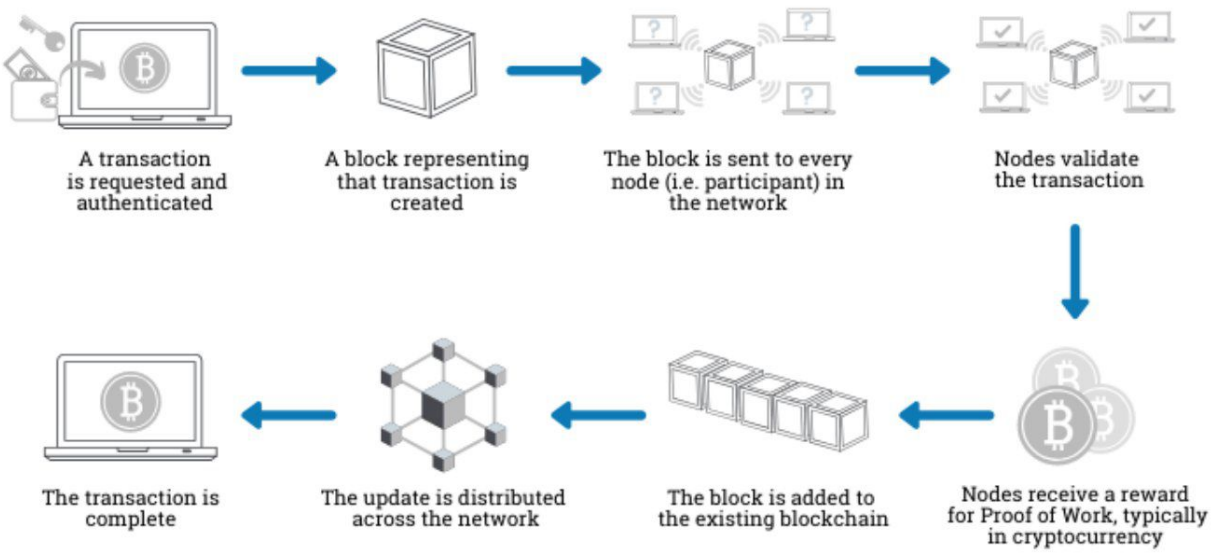
Blockchain

One of the technologies that are under the umbrella of DLT is blockchain. In 2008, an individual under the pseudonym Satoshi Nakamoto authored and released a white paper titled "Bitcoin: A Peer-to-Peer Electronic Cash System." This document introduced the concept of a distributed blockchain, expanding upon existing models. Nakamoto's proposal laid the foundation for decentralized digital currency systems, with Bitcoin being the first implementation.

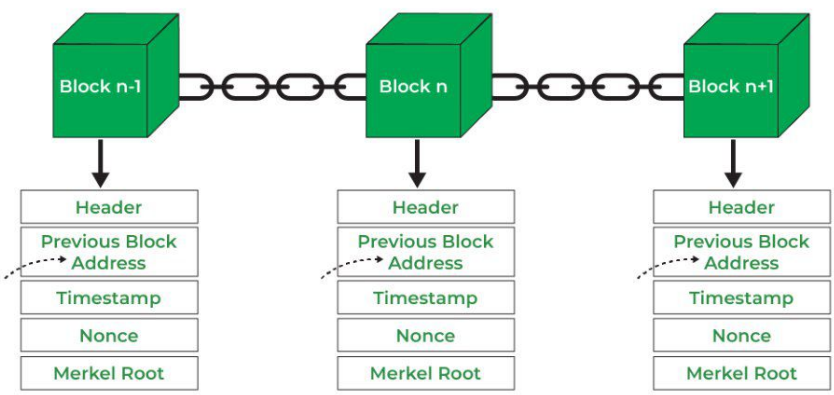
- ✓ A blockchain is a chain of blocks where each block stores information and is associated with a unique hash address, serving as proof of valid transactions.
- ✓ It is a distributed ledger that stores data, including transactions, and is publicly shared across all nodes within the network.
- ✓ Transactions are verified by a network of participating computers within the blockchain.
- ✓ Thus, there is no involvement of any central authority or middlemen, satisfying the decentralization property of blockchain.
- ✓ Once a block of information is created in the chain, it can't be changed or deleted. This makes the blockchain very secure and trustworthy.



How does a transaction get into the blockchain?



Blockchain architecture



1. **Header:** It is used to identify the particular block in the entire blockchain. It handles all blocks in the blockchain.
2. **Previous Block Address/ Hash:** It is a reference to the hash of the previous (parent) block in the chain.
3. **Timestamp:** It is a system verify the data into the block and assigns a time or date of creation for digital documents.
4. **Nonce:** A nonce is a number used only once, crucial for the proof of work in a block. Miners test numerous nonces per second, aiming to find one that meets the required criteria and is valid.
5. **Merkle Root:** A Merkle Root is a data structure that organizes different blocks of data into a tree. It creates a digital fingerprint of all the transactions within a block, enabling users to verify transaction inclusion in a block.

Blockchain properties

- **Decentralization:** One of the fundamental properties of blockchain is decentralization. Instead of relying on a central authority (like a bank or a government), blockchain networks are distributed across a network of nodes. Each node typically maintains a copy of the entire blockchain, and decisions are made through consensus mechanisms rather than by a single central entity.
- **Immutability**: Once data is recorded on a blockchain, it's extremely difficult to alter or delete it. Each block contains a cryptographic hash of the previous block, creating a chain of blocks that are linked together. Any attempt to alter the data in a block would require changing all subsequent blocks in the chain, which is practically infeasible due to the computational resources required. This property ensures the integrity of the data stored on the blockchain.
- **Transparency:** Blockchain networks are often transparent, meaning that the data stored on the blockchain is visible to all participants in the network. Anyone can view the entire transaction history, which promotes trust and accountability within the network.
- **Security:** Blockchain networks use cryptographic techniques to secure transactions and maintain the integrity of the data. Consensus mechanisms such as Proof of Work (PoW) ensure that only valid transactions are added to the blockchain, and the decentralized nature of the network makes it resistant to censorship and attacks.
- **Irreversibility:** Once a transaction is confirmed and added to the blockchain, it becomes practically irreversible. This property is particularly valuable in financial transactions, where it eliminates the need for intermediaries and reduces the risk of fraud.
- **Consensus Mechanisms:** Blockchain networks rely on consensus mechanisms to achieve agreement among nodes on the validity of transactions and the state of the blockchain.
- **Smart Contracts:** Smart contracts are simply programs stored on a blockchain that run when predetermined conditions are met. They typically are used to automate the execution of an agreement so that all participants can be immediately certain of the outcome, without any intermediary's involvement or time loss.
- **Auditability:** Blockchain operates on the Unspent Transaction Output (UTXO) model, where each transaction refers to previous unspent transactions. Once a transaction is recorded in the blockchain, the referenced unspent transactions become spent. This process enables easy tracking of transactions and ensures data integrity between them.

Application areas of blockchain:

- **Banking:** Blockchain enables secure, low-cost peer-to-peer transactions, eliminating the need for intermediaries like banks. Cryptocurrencies facilitate borderless transactions.
- **Cyber Security**: Blockchain enhances security by quickly detecting and preventing cyber threats through decentralized, encrypted data storage. It ensures data integrity while protecting privacy.
- **Supply Chain Management:** Blockchain improves transparency and traceability in supply chains, enabling seamless verification of transactions across the entire chain. It reduce issues like counterfeiting and ensures product authenticity.
- **Healthcare**: Blockchain facilitates secure and instant access to medical data by decentralizing storage and encryption. It prevents data manipulation and enhances privacy and interoperability of medical records.
- **Government:** Blockchain revolutionizes voting systems by enabling secure, anonymous voting and ensuring the accuracy and transparency of election processes. It prevents fraud and manipulation, empowering citizens in democratic participation.

SOME WIDELY USED TERMS IN BLOCKCHAIN

Proof of Work: It is a consensus mechanism used in blockchain networks to validate and confirm transactions and produce new blocks. In this system, miners compete to solve complex mathematical puzzles, requiring significant computational power. The first miner to solve the puzzle broadcasts their solution to the network. Other miners verify the solution, and if it's correct, the new block is added to the blockchain.

Consensus mechanism /Algorithm:A consensus mechanism is a procedure through which all peers of the blockchain network reach a common agreement about the current state of the distributed ledger. In this way, consensus mechanisms achieve reliability in the blockchain network and establish trust between unknown peers in a distributed computing environment. Essentially, the consensus protocol ensures that every new block added to the blockchain is the one and only version of the truth agreed upon by all nodes in the blockchain

Miners: Miners are individuals or entities that participate in the process of validating and adding transactions to the blockchain. They play a crucial role in maintaining the integrity and security of the blockchain network. Miners use specialized hardware and software to solve complex mathematical puzzles in a process known as mining.

When a miner successfully solves a puzzle, they create a new block of transactions, which is then added to the blockchain. In return for their efforts, miners are rewarded with cryptocurrency tokens, such as Bitcoin, as well as transaction fees associated with the transactions included in the block.

Miners compete with each other to be the first to solve the puzzle and add a new block to the blockchain. This competition ensures that no single entity has control over the network and helps to prevent fraudulent or malicious activities.

Cryptocurrency

- ✓ Cryptocurrency refers to a digital or virtual form of currency that utilizes cryptography for security and operates independently of a central authority, such as a government or financial institution.
- ✓ Cryptocurrencies typically operate on decentralized networks based on blockchain technology, which is a distributed ledger that records all transactions across a network of computers.
- ➢ The most well-known cryptocurrency is Bitcoin, which was created in 2009 by an unknown person or group using the pseudonym. Other examples of cryptocurrency are Ethereum, Litecoin Ripple
- ✓ Cryptocurrencies can be stored in digital wallets, which provide users with a private key to access and manage their funds securely.

People can acquire cryptocurrencies through various means:

- ❖ **Mining:** Cryptocurrencies are generated through a process known as mining. Miners use specialized computer systems to solve complex mathematical puzzles. Upon successful completion, they are rewarded with bitcoins or other cryptocurrencies.
- ❖ **Buying, Selling, and Storing:** Users have the option to purchase cryptocurrencies from centralized exchanges, brokers, or individual sellers. Similarly, they can sell cryptocurrencies through these entities. Cryptocurrencies are typically stored in digital wallets, which may be online, offline, hardware-based, or software-based.
- ❖ **Investing:** Cryptocurrencies can be transferred between digital wallets for various purposes, including:
  - ✓ Purchasing goods and services.
  - ✓ Trading on cryptocurrency exchanges.
  - ✓ Converting them into traditional fiat currencies.

Benefits of using cryptocurrency

- ➢ **Decentralization**: Cryptocurrencies operate on decentralized networks, typically utilizing blockchain technology. This means that they are not controlled by any single authority, such as a government or financial institution. This decentralization can lead to increased transparency, security, and resilience against censorship or manipulation.
- ➢ **Lower Transaction Fees:** Traditional financial transactions often involve intermediary institutions that charge fees for their services. Cryptocurrency transactions, on the other hand, can be conducted peer-to-peer, reducing or eliminating the need for intermediaries and lowering transaction fees.
- ➢ **Fast and Borderless Transactions:** Cryptocurrency transactions can be processed quickly, often within minutes, regardless of geographic location. This can be particularly advantageous for international transfers, as cryptocurrencies are not subject to the same delays and fees associated with traditional cross-border transactions.

- **Financial Inclusion:** Cryptocurrencies have the potential to provide access to financial services for individuals who are underserved or excluded by traditional banking systems. People without access to bank accounts or credit cards can participate in the global economy through the use of cryptocurrencies and digital wallets.
- **Security and Privacy:** Cryptocurrency transactions are secured through cryptographic techniques and recorded on immutable, tamper-resistant ledgers (blockchains). This provides a high level of security against fraud and counterfeiting. Additionally, while transactions are recorded on public blockchains, the identities of the parties involved can remain pseudonymous, offering a degree of privacy.
- **Inflation Hedge**: Some cryptocurrencies, like Bitcoin, are designed with a fixed supply or a controlled inflation rate. This makes them potentially attractive as a hedge against inflation, as their value may not be eroded by the debasement of fiat currencies.