INTRODUCTION TO DISTRIBUTED DATA PROCESSING AND REAL TIME SYSTEM

1. What is Data Processing?

Data processing is ingesting massive amounts of data in the system from several different sources such as IoT devices, social platforms, satellites, wireless networks, software logs etc. & running the business logic/algorithms on it to extract meaningful information from it.

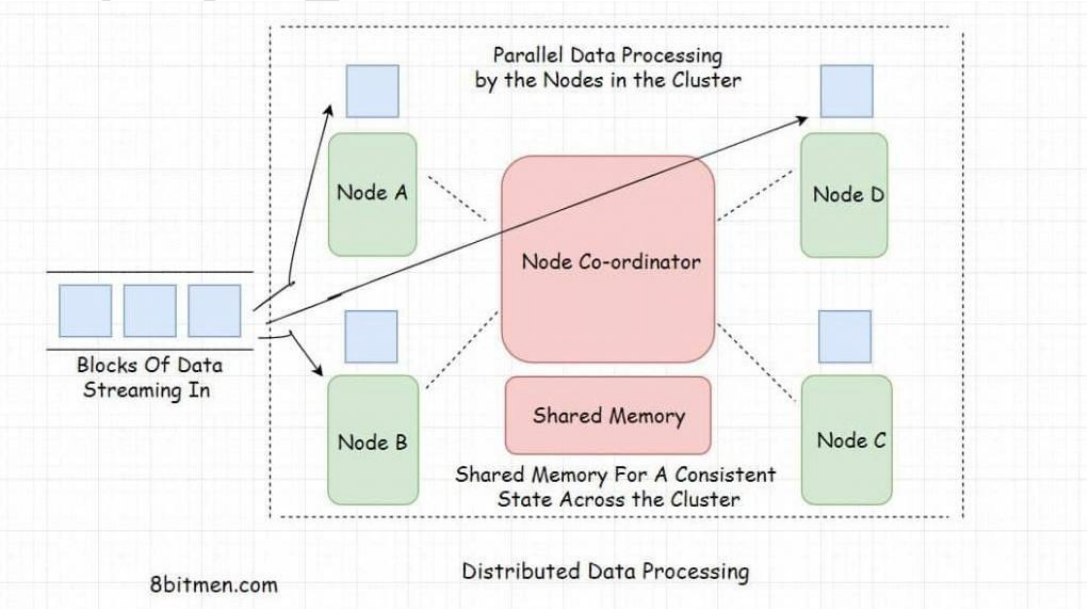
Running algorithms on the data & extracting information from it is also known as Data Analytics.

Data analytics helps businesses use the information extracted from the raw, unstructured, semi-structured data in terabytes, petabytes scale to create better products, understand what their customers want, understand their usage patterns, & subsequently evolve their service or the product.

2. What is Distributed Data Processing? How Different Is It to Centralized Data Processing?

Distributed data processing is diverging massive amount of data to several different nodes running in a cluster for processing.

All the nodes execute the task allotted parallelly, they work in conjunction with each other connected by a network. The entire set-up is scalable & highly available.



Why Process Data in a Distributed Environment? What Are the Upsides?

Processing data in a distributed environment helps accomplish the task in a significantly less amount of time as opposed to when running on a centralized data processing system solely due to the reason that here the task is shared by a number of resources/machines & executed parallelly instead of being run synchronously arranged in a queue.

Since the data is processed in lesser time, it is cost-effective for businesses & helps them to move fast.

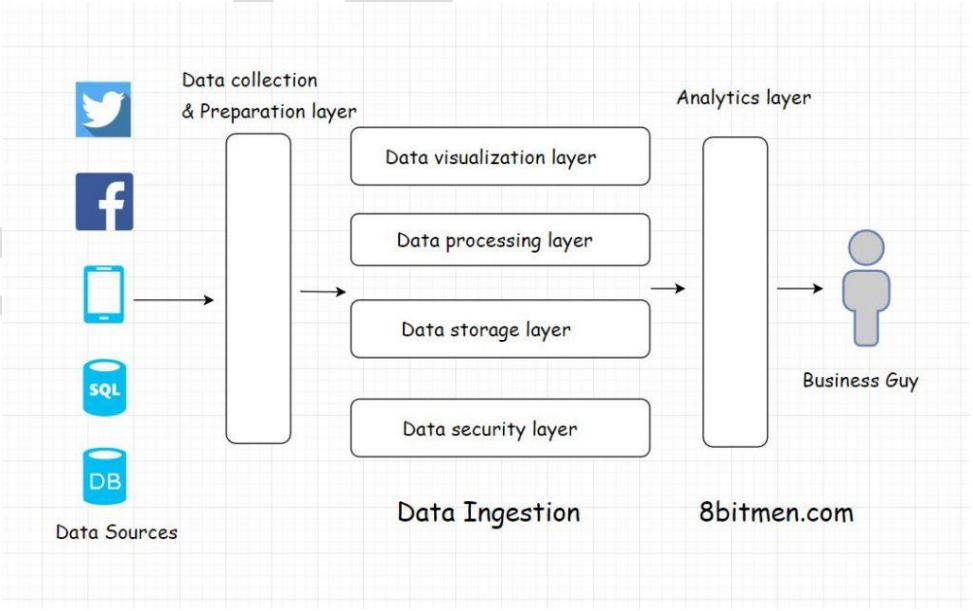
Running a workload in a distributed environment also makes it more scalable, elastic & available. There is no single point of failure. The workload can be scaled both horizontally & vertically.

Data is made redundant & replicated across the cluster to avoid any sort of data loss.

3. How Does Distributed Data Processing Work?

In a distributed data processing system a massive amount of data flows through several different sources into the system. This process of data flow is known as Data ingestion.

Once the data streams in there are different layers in the system architecture which break down the entire processing into several different parts.



Let's quickly have an insight into what they are:

Data Collection & Preparation Layer

This layer takes care of collecting data from different external sources & preparing it to be processed by the system.

When the data streams in it has no standard structure. It is raw, unstructured or semi-structured in nature.

It may be a blob of text, audio, video, image format, tax return forms, insurance forms, medical bills etc.

The task of the data preparation layer is to convert the data into a consistent standard format, also to classify it as per the business logic to be processed by the system.

The layer is intelligent enough to achieve all this without any sort of human intervention.

Data Security Layer

Moving data is vulnerable to security breaches. The role of the data security layer is to ensure that the data transit is secure by watching over it throughout, applying security protocols, encryption & stuff.

Data Storage Layer

Once the data streams in it has to be persisted. There are different approaches to do this.

If the analytics is run on streaming data in real-time in-memory distributed caches are used to store & manage data.

On the contrary, if the data is being processed in a traditional way like batch processing distributed databases built for handling big data are used to store stuff.

Data Processing Layer

This is the layer contains logic which is the real deal, it is responsible for processing the data.

The layer runs business logic on the data to extract meaningful information from it. Machine learning, predictive, descriptive, decision modelling is primarily used for this.

Data Visualization Layer

All the information extracted is sent to the data visualization layer which typically contains browser-based dashboards which display the information in the form of graphs, charts & infographics etc.

Kibana is one good example of a data visualization tool, pretty popular in the industry.

4. Types of Distributed Data Processing

There are primarily two types of it. Batch Processing & Real-time streaming data processing.

Batch Processing

Batch processing is the traditional data processing technique where chunks of data are streamed in batches & processed. The processing is either scheduled for a certain time of a day or happens in regular intervals or is random but not real-time.

Real-time Streaming Data Processing

In this type of data processing, data is processed in real-time as it streams in. Analytics is run on the data to get insights from it.

A good use case of this is getting insights from sports data. As the game goes on the data ingested from social media & other sources is analyzed in real-time to figure the viewers' sentiments, players stats, predictions etc.

5. Pros & Cons of Distributed Data Processing

Pros

- Distributed data processing facilitates scalability, high availability, fault tolerance, replication, redundancy which is typically not available in centralized data processing systems.

- Parallel distributed of work facilitates faster execution of work.

- Enforcing security, authentication & authorization workflows becomes easier as the system is more loosely coupled.

Cons

- Setting up & working with a distributed system is complex. Well, that's expected having so many nodes working in conjunction with each other, maintaining a consistent shared state.

- The management of distributed systems is complex. Since the machines are distributed it entails additional network latency which engineering teams have to deal with.

- Strong consistency of data is hard to maintain when everything is so distributed.

Real Time Systems

Real time system means that the system is subjected to real time, i.e., response should be guaranteed within a specified timing constraint or system should meet the specified deadline. For example: flight control system, real time monitors etc.

Types of real time systems based on timing constraints:

Hard real time system:

This type of system can never miss its deadline. Missing the deadline may have disastrous consequences. The usefulness of result produced by a hard real time system decreases abruptly and may become negative if tardiness increases. Tardiness means how late a real time system completes its task with respect to its deadline. Example: Flight controller system.

Soft real time system:

This type of system can miss its deadline occasionally with some acceptably low probability. Missing the deadline have no disastrous consequences. The usefulness of result produced by a soft real time system decreases gradually with increase in tardiness. Example: Telephone switches.

Reference model of real time system: Our reference model is characterized by three elements:

1. **A workload model:** It specifies the application supported by system.

2. **A resource model:** It specifies the resources available to the application.

3. **Algorithms:** It specifies how the application system will use resources.

Terms related to real time system:

- **Job** – A job is a small piece of work that can be assigned to a processor and may or may not require resources.
- **Task** – A set of related jobs that jointly provide some system functionality.
- **Release time of a job** – It is the time at which job becomes ready for execution.
- **Execution time of a job** – It is the time taken by job to finish its execution.
- **Deadline of a job** – It is the time by which a job should finish its execution. Deadline is of two types: absolute deadline and relative deadline.
- **Response time of a job** – It is the length of time from release time of a job to the instant when it finishes.
- Maximum allowable response time of a job is called its relative deadline.
- Absolute deadline of a job is equal to its relative deadline plus its release time.
- Processors are also known as active resources. They are essential for execution of a job. A job must have one or more processors in order to execute and proceed towards completion. Example: computer, transmission links.
- Resources are also known as passive resources. A job may or may not require a resource during its execution. Example: memory, mutex
- Two resources are identical if they can be used interchangeably else they are heterogeneous.