PAGE NO.
DATE / /

Lambda Architecture

Q1) Explain the factors leading to Big Data. List and explain major sources of Big Data?

A) Factors leading to Big Data: The emergence of Big Data is driven by several key factors that have collectively transformed the way data is generated, collected:

i) Increase in Data Volume: The exponential growth in data generated from various sources, such as social media, sensors, and transaction records, has led to the creation of massive datasets.

ii) Variety of Data: Data comes in various forms, including structured, semi-structured & unstructured data.

iii) Velocity of Data Generation: The speed at which data is generated and needs to be processed is crucial factor.

iv) Advances in Storage Technology: The reduction in the cost of storage & the development of scalable storage solutions.

v) Development of Advanced Analytics: The needs for more complex analytics & machine learning algorithms to derive insights from large datasets.

vi) Cloud Computing: The advent of cloud computing has provided the necessary infrastructure to store and process large datasets.

vii) Increased connectivity: The proliferation of internet connectivity & mobile devices has led to surge in data generation.

* Major sources of Big Data.

i) Social media platforms ii) IoT Devices and sensors.

iii) Transactional data iv) Healthcare v) Telecommunication

E-Commerce Data, Scientific Research

Q2) List and explain the characteristics of Big Data.
BigData is typically defined by its unique characteristics, often referred to as the "3Vs".
i) Volume: refers to the vast amount of data generated every second from various sources.
The sheer size of data being produced is enormous, ranging from terabytes to petabytes and even zettabytes.
ii) Variety: refers to the different types of data that are generated from multiple sources.
Data comes in various forms, including structured data, semi structured data, and unstructured data.
iii) Velocity: refers to the speed at which data is generated collected and processed.
Real time data streams from social media feeds, financial markets, and sensor networks requires immediate processing and analysis to extract actionable insights.
"These are the 3 V's of Big data but over time, additional 'Vs' has been added to encapsulate its complexity.
iv) Veracity: refers to uncertainty or trustworthiness of data. Not all data is accurate or reliable. Veracity deals with the quality & accuracy of data as well as the challenges posed by inconsistencies, biases & noise in data.
v) Value: refers to the potential economic or organizational benefits derived from data. The ultimate goal of analyzing Big Data is to extract valuable insights that can lead to better decision making.
vi) Variability: refers to the inconsistencies & fluctuations in data over time. Data can be highly inconsistent, with peaks and troughs in data flow.
vii) Visualization: refers to the ability to represent complex data in a visual format that is easy to understand & interpret. With the vast amounts of data involved, traditional data visualization methods often fall short.

Q3) Major challenges of Big Data Systems.
i) Data volume Management: One of the primary challenges is handling the sheer volume of data.
ii) Data Variety: Managing this variety is challenging because different data types require different processing techniques.
iii) Data velocity: Real time data sources require immediate analysis to provide actionable insights.
iv) Data Quality & Veracity: Big data often includes incomplete, noisy and inconsistent data, which led to incorrect analyses & poor decision making.
v) Scalability: This includes scaling up storage capacity, computation resources & data processing capabilities.
vi) Data Security & privacy: Ensuring privacy is challenging due to distributed nature of Big Data Systems.

vii) Data Integration: Data may come from different databases, each with its own format schema & semantics.
viii) Data Governance: it is challenging due to the distributed & complex nature of the data.
ix) Data processing & Analysis: Traditional analytical tools often struggles with the scale & diversity of big data.
x) Cost Management: Building & maintaining BDS can be expensive.

Q4) Discuss the problems faced by traditional database systems?
i) Scalability limitations: RDBMS often involves expensive & complex hardware upgrades (vertical scaling)
ii) Handling unstructured data: RDBMS can not handle unstructured data because of rigid schema structure of traditional database.
iii) Performance Bottlenecks: RDBMS can be overwhelm for real time data processing & may slowdown or delayed.
iv) Complex Data Integration: RDBMS struggle to integrate and manage data from different diverse sources.
v) Lack of flexibility and Agility: Any change to schema, such as adding new fields or modifying existing ones can be time consuming & disruptive.
vi) High cost of Maintenance: The need for powerful hardware specialized database administrators, and regular upgrades adds to the overall cost.
vii) Limited Support for Distributed Computing: RDBMS lacks support for distributed computing.
viii) Challenges with Real-Time Processing: RDBMS inability to handle real time data streams effectively limits the use.
ix) Complexity in Handling High Volume Transaction: ensuring ACID in high volume transactional systems may lead to bottlenecks.
x) Data Silos: RDBMS can lead to the creation of data silos, where data is isolated within different departments or systems, making it difficult to integrate.

Q5) Discuss the required properties for Big Data Systems?
* Scalability: it is the fundamental property of BDS, enabling them to handle increasing amounts of data & computational load.
* Flexibility: ability to adapt to changing data schemas, sources & formats without requiring significant reconfiguration or downtime.
* Fault tolerance: designed to detect & recover from failures automatically minimizing downtime & data loss.
* High throughput: ensure that the system can handle the ingestion, processing & analysis of data at a rate of real time.
* Low latency: refers to minimal delay between data input & output enabling rapid processing & immediate response.
* Data consistency: that all users & applications see the same version of the data at any given time.
* Data Security & privacy: This includes encryption, access control authentication, & compliance with data privacy regulations.
* Efficient data storage and management: use of distributed files systems, data compression, that balance speed & capacity.

* Advanced data processing capabilities: Enables organizations to extract valuable insights from large complex datasets.
* Interoperability & Integration: enabling data sharing, collaboration & comprehensive analysis across platforms.
* Cost Efficiency = systems should be designed to optimize resource utilization minimize operational costs
* Ease of use & Management: Provides intuitive interfaces, automated management tools & support for easy deployment.

Q(b). Explain the different layers of lambda architecture.
Lambda Architecture is a data processing architecture designed to handle massive quantities of data by utilizing both batch & real time processing methods.
1) Batch layer: is responsible for managing the historical data & processing it in large batches. This layer ensures that the data is processed in its entirety to produce a comprehensive accurate view of the data
* Components.
Master datasets: This is the immutable, append only dataset that stores the entire historical data. It acts as the source of truth of the system.
Batch processing systems: This component process the master datasets periodically, typically using distributed computing frameworks like Hadoop or Spark. The output is a set of precomputed views or models, which are stored and used to answer queries that require complete and accurate data analysis.
output: The results of the Batch layer is usually a batch view or a precomputed model that can be queried for in depth insights
2) Speed layer: is designed to handle real time data processing and provide immediate results. It complements the batch layer by processing data as it arrives, allowing for quick responses to data events.
Components:
* Real time processing System: This system processes incoming data streams in real time using technologies like Apache Storm, Apache Kafka Streams, or Apache flink. It generates low latency real time views of data, which may not be as accurate or comprehensive as those produced by the batch layer but are available immediately.
output: The speed layer outputs real time views that are quickly accessible and provide immediate insights into recent data.
3) Serving layer: The serving layer is responsible for merging and presenting the output from both the batch layer & the speed layer. It enables querying of the data in a way that combines the accuracy of batch processing with the immediacy of real time processing
Components:
query systems: This component allows users to query the merged views from both the batch layer & speed layer. It may involve databases or indexes that can handle large volumes of data & provide fast query results.

View Merging: The serving layer must intelligently merge the results from the batch and speed layers, often giving precedence to the real-time data but ensuring that the batch data is used to provide accurate long-term insights.
*Output: The serving layer provides users with a unified view of the data, combining the completeness of batch views with the timeliness of real-time views.

7)) Differentiate b/w Re-computation algorithm & incremental algorithm

	Re-computation	Incremental
Definition	Recalculates the entire dataset whenever an update occurs.	updates only the affected part of the datasets.
process	process all data from scratch upon every update	process only the new or modified data.
Advantages	- Higher accuracy - Simpler to implement	- more efficient. - Low latency.
Disadvantages	- Computationally expensive - High latency	- complex to implement - Risks of inconsistency.
Efficiency	Low, especially for large datasets	High, due to processing only changes
Latency	High, due to reprocessing the entire dataset	low as only a small portion of data is processed
complexity	- Simple to implement	more complex, requires tracking of changes.
usecases	- Batch processing - Infrequent updates	- Real time analytics - Frequent data updates.

8)) List the requirements and responsibilities of batch layer

Requirements
* Data Storage: must store an immutable, append only master dataset
* Scalability: able to scale horizontally, handle ever increasing data
* Fault Tolerance: ensure data integrity & process continuity.
* Comprehensive data processing: Extensive data processing on the entire dataset to generate accurate & complete views.
* Processing Time: Should complete processing within a acceptable time

Responsibilities.
* Storing the master dataset: Contain all historical data in a immutable format
* Batch processing: Execute batch processing jobs on master dataset.
* Generating Batch views: Accurate & complete batch views that reflect full data
* Handling large scale data: Ensuring that the system can handle vast amount of information without performance degradations.
* Data Integrity & Consistency: ensure that the data processing results are consistent and accurate, reflecting true state of dataset
* providing Historical Data Analysis: Offer deep insights & trends.
* Resource Management: to optimize the processing of large datasets

9)) Explain the requirements of Serving layer in lambda architecture
* Low Latency Query Access: provide quick low latency access
* Scalability: able to scale horizontally to handle a large queries
* Data consistency & merging: must intelligently merge results from both layers
* fault tolerance & Reliability: must be resilient to failures.
* Efficient Data Storage: Support fast reads while optimizing storage space

Versioning & Time travel queries: providing access to different versions of data.
* Handling large data volumes: Efficiently handle large data volumes.
* Multi-user query support: should support concurrent querying of data
* Query flexibility: allow ad hoc pre defined complex queries.
* support for real time data: provide immediate access to real time data.

10Q) With example show how low latency and high throughput can be achieved in serving layer of lambda architecture.

* low latency ensure that users receive responses to their queries almost instantly:

* Precomputed Batch views: suppose you are running a retail analytics system where you need to show the total sales for each product category every hour. The batch layer can precompute the hourly sales totals for each product category and store these precomputed results in a key value store.
Low latency solutions: When a user queries the total sales for a specific product category, instead of recalculating from raw data, the serving layer can instantly retrieve the precomputed result from the key value store, ensuring quick response times.

* Real time updates from the speed layer: In a social media platform the speed layer provides real time updates for user activity (likes, comments, shares). Whenever a user makes a query for recent activity, the serving layer fetches these real time updates directly from the speed layers in memory store (like Redis or memcached)
* Low latency solutions = By storing real time updates in a fast in memory cache, queries can retrieve the latest data with minimal delay providing low latency response

* Indexing for faster querying: In a log monitoring system if users query for error logs in a specific time range, indexing the logs based on timestamps can significantly speed up retrieval. Instead of scanning the entire dataset, the index allows the system to directly access the relevant logs.
* (Low) Latency solutions: Indexing the data in the serving layer allows for quick lookups, reducing the time it takes to retrieve & display results to the user.
Achieving high throughput in serving layer.
* Distributed data storage:
In a video streaming platform millions of users may be querying video recommendations at the same time. By storing precomputed recommendations in a distributed database like Cassandra or Base, queries can be distributed across multiple nodes, preventing bottlenecks.
* High throughput solutions: A distributed key value store allows the system to balance the load across different nodes, handling many concurrent queries while maintaining fast response times.

Sharding and partitioning: In an e-commerce platform the serving layer stores product search data across multiple partitions based on product categories. When a user searches for products the system only queries the relevant partition.
High throughput solution: Sharding the dataset ensures that queries are distributed evenly across different partitions increasing the system's capacity to handle many queries at once.

11) a) List the requirements and responsibilities of speed layer
- low latency data processing: must process incoming data quickly
- efficient handling of real time data: handle continuous streams of data
- approximate and incomplete results:
- Scalability
- fault tolerance
- State Management
- Responsibilities.
- Real time Data Ingestion
- Real time processing
- providing Real Time Views
- Complementing the Batch Layer
- ensuring eventual consistency
- Stateful computation.
- fault Tolerant Real Time Data processing.

12) Differentiate b/w batch & speed layer

	Batch layer	Speed layer
function	process large volumes of historical data	Handle real time data to provide low latency.
paradigm	Batch processing	stream processing (continuous real time data)
-ing model	(large chunks of data)	
latency	High latency (results in hours)	(low latency (near real time events)
Accuracy	provide accurate, complete results	produce approximate results
data scope	works on large datasets	process real time data streams
scalability	scales for large datasets but not suitable for real time	scales to handle high throughput for real time.
fault tolerance	fault tolerant as data can be recomputed from dataset	Requires careful state management.
Responsibility	maintains master copy storage distributed file systems.	real time views to fill the gap use in memory
querying	suitable for complex queries	suitable for quick real time queries
use cases	historical trends.	Real time monitoring.
	all modes	alerting systems.