1. What is the Difference Between Normal Data and Big Data?

POC	Normal Data	Big Data
Size	Small to moderate datasets that can be easily managed and analyzed on a single machine or server.	Very large datasets that exceed the processing and storage capacity of a single machine and require distributed storage and processing.
Volume	Limited amount of data that can be easily handled and processed using traditional methods.	Huge amount of data that may exceed petabytes or exabytes and require specialized tools and techniques for storage and processing.
Speed	Low data generation rate, typically generated at a manageable pace.	High data generation rate, often generated in real-time or near real-time, requiring quick processing and analysis.
Variety	Limited diversity in data types and sources, typically structured data.	Diverse data types including structured, semi-structured, and unstructured data from various sources such as social media, sensor data, text, images, etc.
Reliability	Relatively reliable data with minimal noise or errors.	Data may contain noise, errors, and inconsistencies due to its large volume and diverse sources, requiring thorough data cleaning and validation processes.
Storage	Can be easily stored on a single machine or server using traditional storage systems.	Requires distributed storage systems such as distributed file systems (e.g., Hadoop HDFS) or cloud-based storage solutions due to the large volume of data.
Processing	Can be processed using traditional methods and algorithms on a single machine or server.	Requires distributed computing and parallel processing using frameworks such as Apache Spark, Apache Hadoop, or specialized big data processing platforms to efficiently process and analyze data.
Analysis	Can be analyzed using traditional data analysis techniques and tools.	May require advanced analytics tools and techniques such as machine learning, natural language processing, and deep learning algorithms to gain insights from the large and complex data sets.
Data Integration	Relatively simple to integrate data from different sources using traditional data integration techniques.	Challenging to integrate data from multiple sources due to its size, diversity, and complexity, often requiring advanced data integration techniques such as data lake architectures, data pipelines, and data integration platforms.
Data Quality	Easier to ensure data quality and accuracy through traditional data quality measures.	May require additional efforts to ensure data quality and accuracy due to the large volume, diversity, and variability of big data, often requiring big data-specific data quality measures and tools.
Data Visualization	Traditional data visualization techniques can be used to represent data.	May require specialized visualization techniques and tools that can handle the large volume and complexity of big data, such as big data visualization libraries or dashboards.
Decision Making	Relatively straightforward to make decisions based on the data analysis.	May require advanced algorithms, techniques, and tools for decision making, including real-time or near real-time decision making based on streaming big data.
Example	Sales data for a small retail store with a few hundred thousand records.	Social media data with billions of posts, tweets, and comments generated every day.

2. What are Other Non-Relational Database Management Systems?

NoSQL databases, also referred to as non-relational databases, diverge from the conventional relational model of tables, rows, and columns. They employ diverse data models like key-value, column-family, document, and graph to facilitate data storage and retrieval. Some notable examples of NoSQL database management systems include:

DBMS	Data Model	Characteristics	Advantages	Disadvantages	Use Cases
MongoDB	Document	JSON-like documents, high scalability, flexible data model	• Easy scalability and horizontal sharding. • Flexible data model allows for dynamic schema changes. • Rich query capabilities • Built-in replication and fault tolerance	• May require additional efforts for data consistency. • Can consume high storage space. • Requires careful data modeling for optimal performance.	Large volumes of unstructured or semi-structured data, such as social media data, sensor data, and product catalogs
Cassandra	Column-Family	Highly scalable, distributed, write-heavy workloads	• High availability and fault tolerance • Linear scalability with high write throughput • Designed for distributed and decentralized architecture. • Tunable data consistency levels	• Limited support for complex queries • Requires careful data modeling. • Increased complexity in managing a distributed environment	High availability, fault tolerance, write-heavy workloads, such as in IoT, finance, and gaming
Redis	Key-Value, Lists, Sets, Sorted Sets	In-memory data store, high performance, low latency	• Blazing-fast read and write performance. • Supports a variety of data structures. • High throughput and low latency • Pub/Sub messaging system	• Limited support for complex queries • Data size constrained by available memory. • Data durability concerns as data is stored in memory	Real-time data processing, caching, messaging
Neo4j	Graph	Graph structures (nodes, edges, properties), complex relationship modeling	• Efficient handling of complex relationships and traversals • Supports flexible data modeling. • Powerful graph query language (Cypher) • ACID compliant	• May require specialized knowledge for data modeling. • Limited scalability compared to some other NoSQL databases. • Higher storage requirements for graph data	Social networks, recommendation systems, fraud detection