Big Data

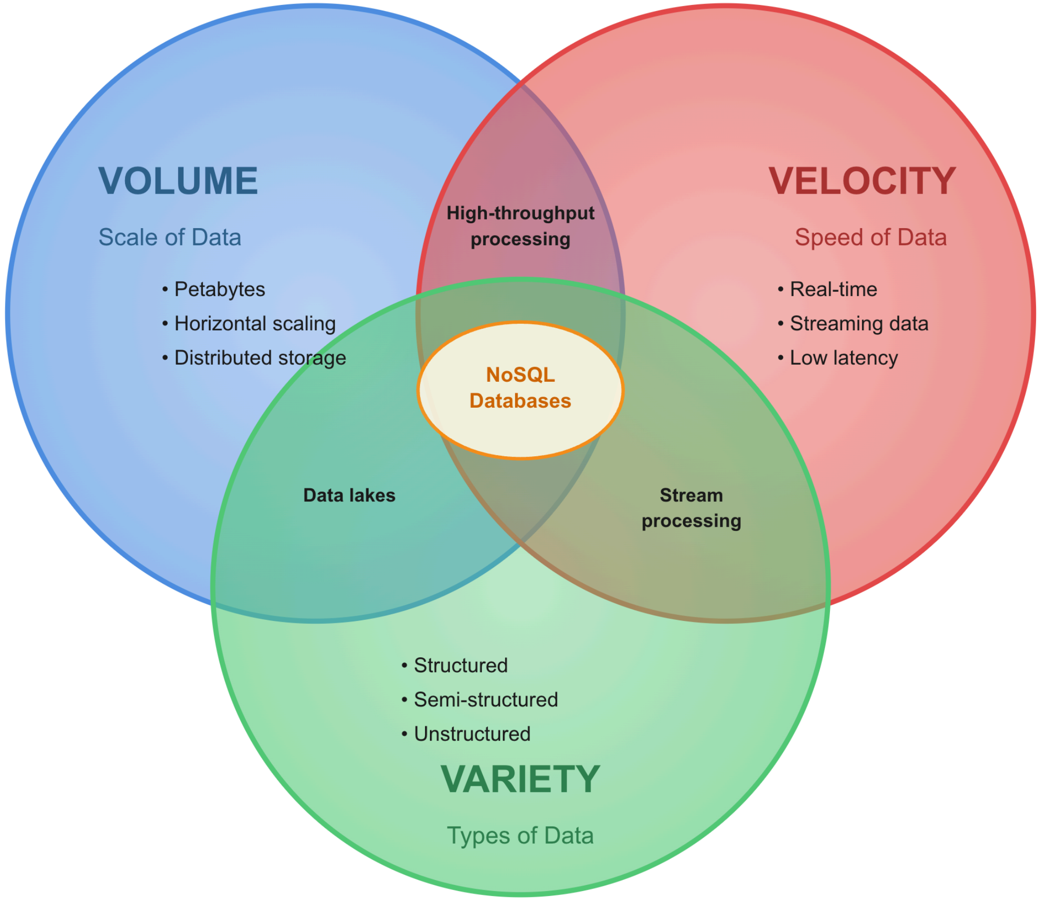
Volume, Velocity, Variety

What is Big Data?

> **Big Data** refers to data that displays the characteristics of volume, velocity, and variety (**the 3 Vs**) to an extent that makes the data unsuitable for management by a relational database management system.

- **Volume**: the amount of data to be stored
- **Velocity**: the speed at which data is entering the system
- **Variety**: the variations in the structure of the data to be stored

Big Data Characteristics Diagram



Pioneers of Big Data Technology

Google (to index the web)

- Google File System (GFS)
- MapReduce (distributed data processing)
- BigTable (key-value store)

Amazon (for web commerce at scale)

- Dynamo (key-value store)

Facebook (for social graph processing)

- Cassandra

Today: Tech advancement has increased the opportunity for organizations to generate and track data (e.g. via personal connected devices)

Volume: Handling Large Amounts of Data

Units of Data Volume

Amount of Data	Name	Abbreviation	Order of Magnitude
1024 (2^{10}) bytes	kibibyte	KiB	~10^3 bytes
1024 KiB	mebibyte	MiB	~10^6 bytes
1024 MiB	gibibyte	GiB	~10^9 bytes
1024 GiB	tebibyte	TiB	~10^{12} bytes
1024 TiB	pebibyte	PiB	~10^{15} bytes
1024 PiB	exbibyte	EiB	~10^{18} bytes
1024 EiB	zebibyte	ZiB	~10^{21} bytes

Note: kibi-, mibi-, gibi- etc. increase by 1024x (2^{10}), while kilo-, mega-, giga- increase by 1000x

Scale of Modern Storage

- The largest storage systems today (e.g., cloud storage at Amazon, Google, Microsoft) are approaching a **Zebibyte**
- That's 1,024 Exbibytes!
- Or approximately 1,180,591,620,717,411,303,424 bytes

Two Approaches to Handle Volume

Scale Up (Vertical Scaling)

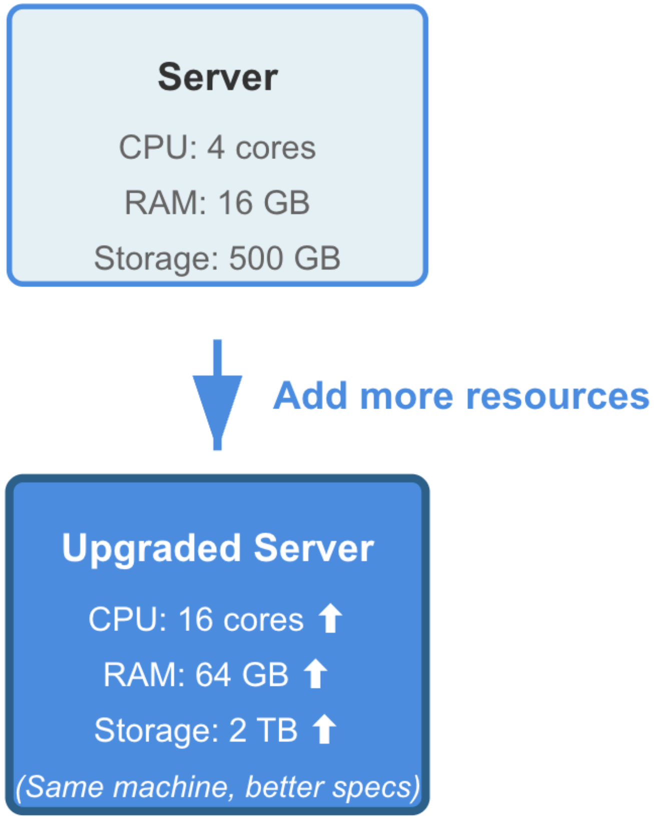
- Increase the CPU, RAM, Disk of each storage machine
- Keep the number of machines fixed

Scale Out (Horizontal Scaling)

- Keep the CPU, RAM, Disk of each machine fixed
- Increase the number of machines

Scaling Approaches Visualized

Vertical Scaling (Scale Up)



Server

CPU: 4 cores

RAM: 16 GB

Storage: 500 GB

Add more resources

Upgraded Server

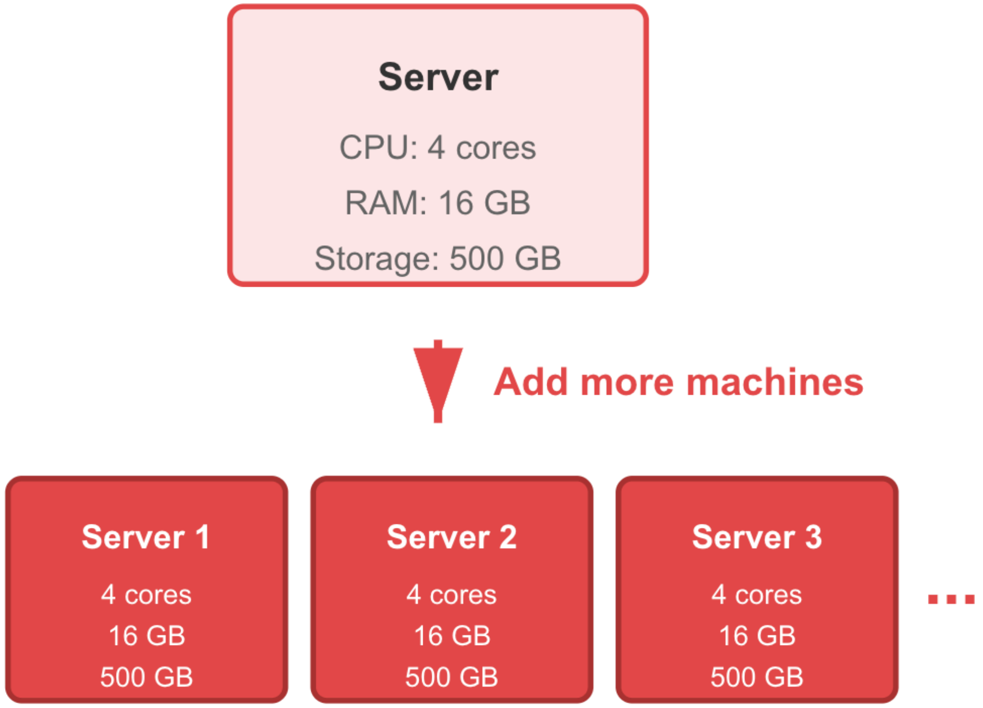
CPU: 16 cores ⬆

RAM: 64 GB ⬆

Storage: 2 TB ⬆

(Same machine, better specs)

Horizontal Scaling (Scale Out)



Server

CPU: 4 cores

RAM: 16 GB

Storage: 500 GB

Add more machines

Server 1	**Server 2**	**Server 3**	...
4 cores	4 cores	4 cores	
16 GB	16 GB	16 GB	
500 GB	500 GB	500 GB	

(Multiple machines work together)

Comparing Scaling Approaches

	Capacity	**Cost**	**Coordination**
Scale Up	❌ machine limit	❌ specialized hw	✅ few machines
Scale Out	✅ add machines	✅ commodity hw	❌ many machines

Why RDBMS Struggles with Volume

- **RDBMS requires high coordination**
    - Tables are related via common attributes
    - Maintaining referential integrity across distributed systems is challenging
- **Can only scale up**
    - Limited by physical machine constraints
    - Results in lower capacity and higher cost
- **Result**: RDBMS is unsuitable for Big Data Volume

NoSQL: The Volume Solution

- **NoSQL compromises on relational power**
    - Limited transaction support
- **In return, can scale out**
    - Add more commodity hardware
    - Higher capacity at lower cost
    - Better suited for handling volume

Velocity: Handling Speed of Data

Velocity Challenge Example

- **Example**: A cloud storage system like Google Cloud Storage handles **~10 million requests per second**
- **Problem**: If a scaled-up machine can handle ~10,000 requests per second, you would need **1,000 machines**!

Velocity Requires Scaling Out

- High velocity cannot be handled by a single machine
- **Must scale out** (same as for volume)
- RDBMS is not a good fit for velocity challenges
- NoSQL systems designed for distributed request handling

Variety: Handling Different Data Types

Structured vs. Unstructured Data

Structured Data

- Data that conforms to a predefined model (e.g., a table schema)
- RDBMS requires this!

Unstructured Data

- Can be anything, does not conform to a model
- Examples: videos, texts, emails, sensor data, social media posts

Semi-structured Data

- Parts are structured and parts are unstructured
- Examples: JSON documents, XML files

The Real World Challenge

- The real world is **full of unstructured data**
- Most valuable data doesn't fit neatly into tables
- Examples:
  - Customer reviews (text)
  - Product images (binary)
  - Click streams (logs)
  - IoT sensor readings (time series)

NoSQL Approach to Variety

Flexible Schema

- Ingest unstructured data first
- Impose structure as needed for applications
- Structure during retrieval and processing, not storage

Benefits

- Adapt to changing data formats
- Store diverse data types together
- No upfront schema design required

Key Takeaways

1. Big Data is defined by the **3 Vs**: Volume, Velocity, and Variety
2. **Scaling Out** (horizontal scaling) is needed for Volume and Velocity (not scaling up or vertical scaling)
3. **RDBMS** struggles with Big Data because
    i. It requires high coordination (=> horizontal scaling infeasible => can't handle Volume and/or Velocity)
    ii. It requires data to conform to model/schema (=> can't handle Variety)
4. **NoSQL** trades relational power for the ability to scale out (to handle Volume/Velocity) and to support a Variety of data
5. Modern cloud storage systems are approaching **Zebibyte** scale (~10^21 bytes)