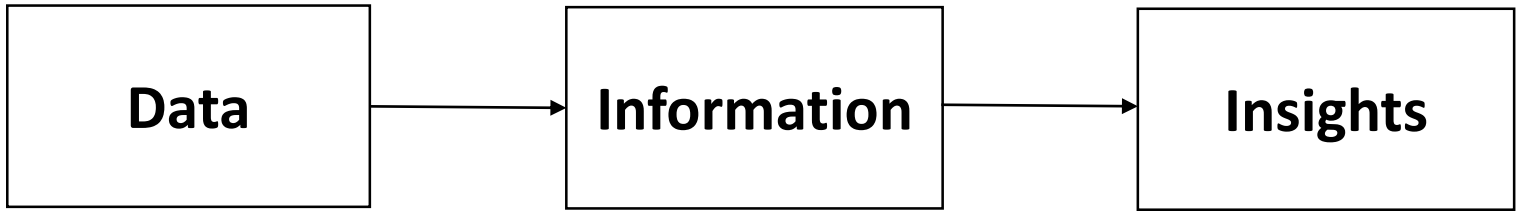
Big Data Analytics

Module-1

Prepared By:
Rajesh Nayak

1. Types of Digital Data

- Irrespective of the type of the enterprise, data continues to be precious and irreplaceable asset.
- Data can be present internal to the enterprise and also exists outside the four wall and firewalls of the enterprise.
- Data is present in homogeneous sources and heterogeneous sources.
- The need of the hour is to understand, manage, process, and take data for analysis to draw valuable insights.



1.1 Classification of Digital Data

- Digital data is classified into 3 categories.
  1. Unstructured data – This is the data which does not conform to a data model or is not in a form which can be used easily by a computer program.

     Ex: Memos, chat rooms, PPTs, images, videos, body of the email, etc.
  2. Semi-structured data – This is the data which does not conform to a data model but has some structure. Ex: HTML, E-mail, etc.
  3. Structured data – This is the data which is in an organized form, and can be used by a computer program. Ex: Relations in RDBMS.
- Structured Data: This is the data which is in an organized form (e.g., in rows and columns) and can be easily used by a computer program.
- Ex: Relation table in RDBMS with rows/tuples, columns/attributes, constraints. Each record in the table will have exactly same structure.

1.1 Classification of Digital Data

- Sources of structured data: If your data is highly structured, one can look at leveraging any of the available RDBMS (Oracle Corp.-Oracle, IBM-DB2, Microsoft-Microsoft SQL Server, EMC-Greenplum, Teradata-Teradata, MySQL (open source), PostgreSQL (advanced open source), etc.) to house it.

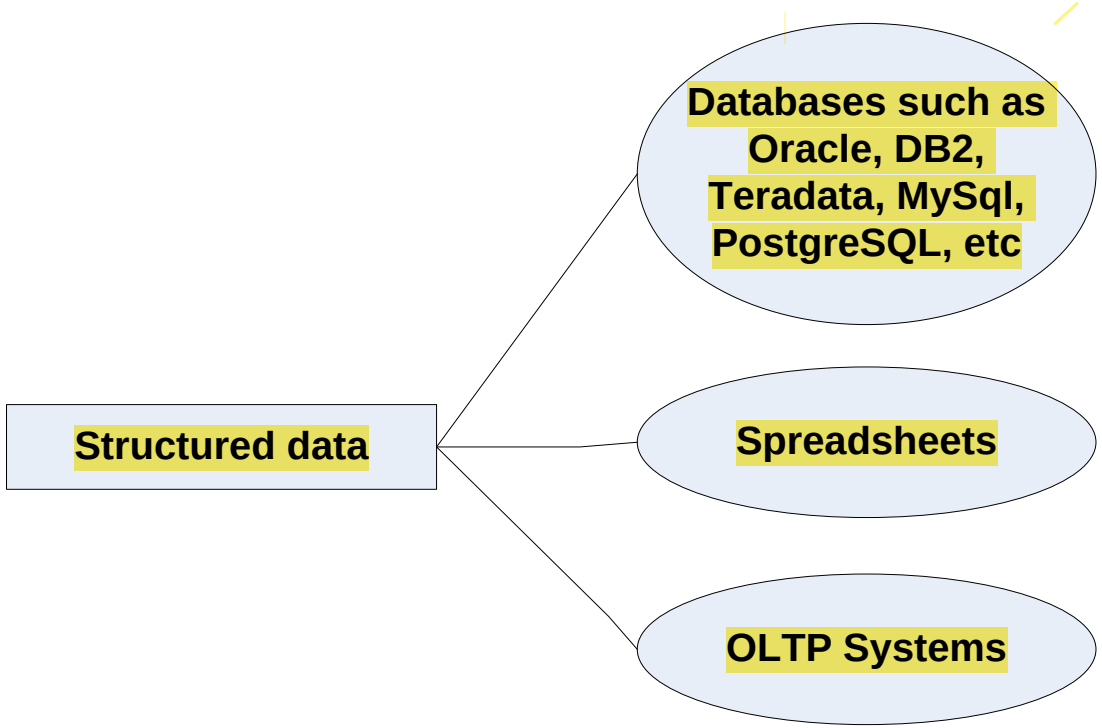


Fig.: Sources of structured data

1.1 Classification of Digital Data

- Ease of working with structured data:

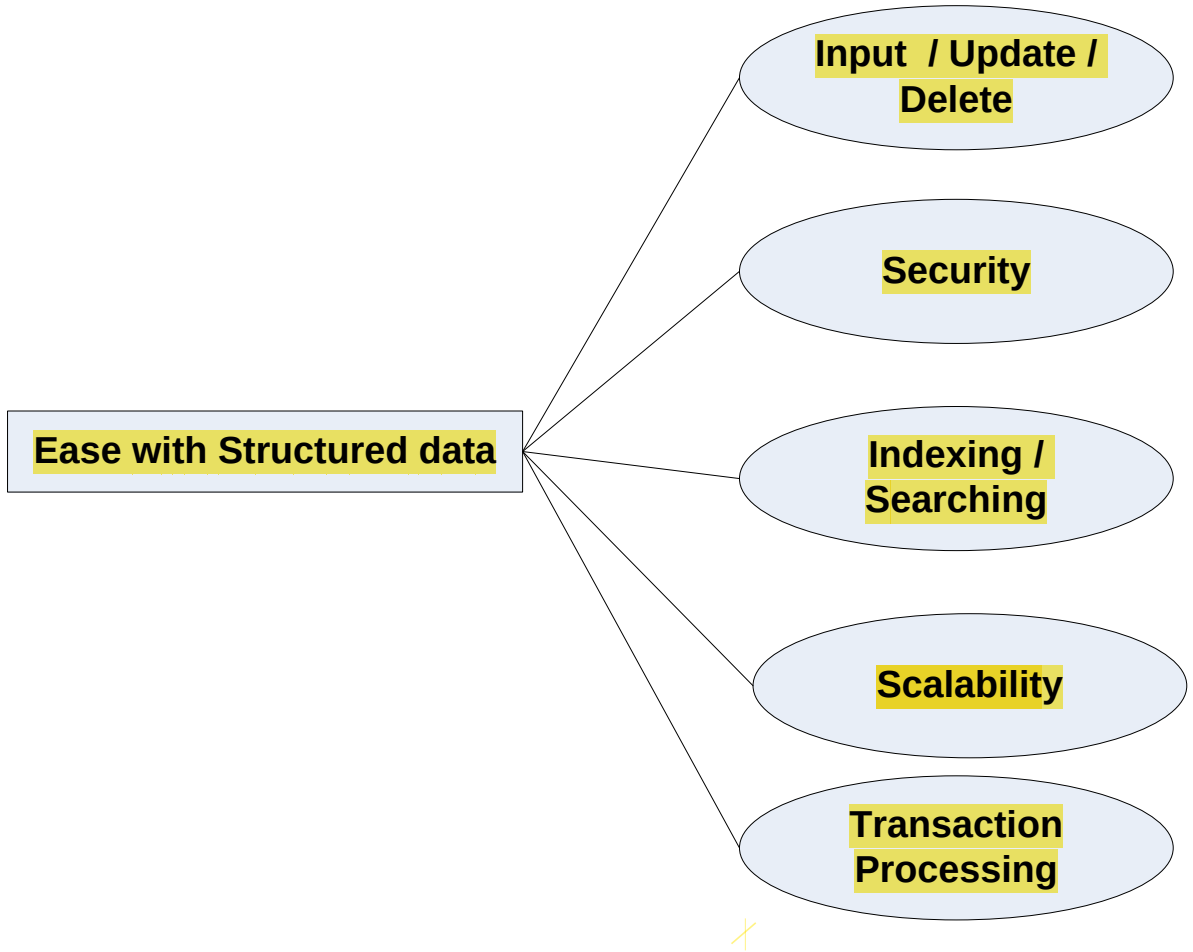


Fig.: Ease of working with structured data

1.1 Classification of Digital Data

- Semi-Structured Data: Semi-structured data is also referred to as self-describing structure. It has following features:

1. It does not conform to data models that one typically associates with relational database or any other form of data tables.
2. It uses tags to segregate semantic elements.
3. Tags are also used to enforce hierarchies of the records an fields within data.
4. There is no separation between the data and the schema.
5. In semi-structured data, entities belonging to the same class and also grouped together need not necessarily have the same set of attributes. If at all, they have same set of attributes, the order may not be similar.

1.1 Classification of Digital Data

- Semi-Structured Data:

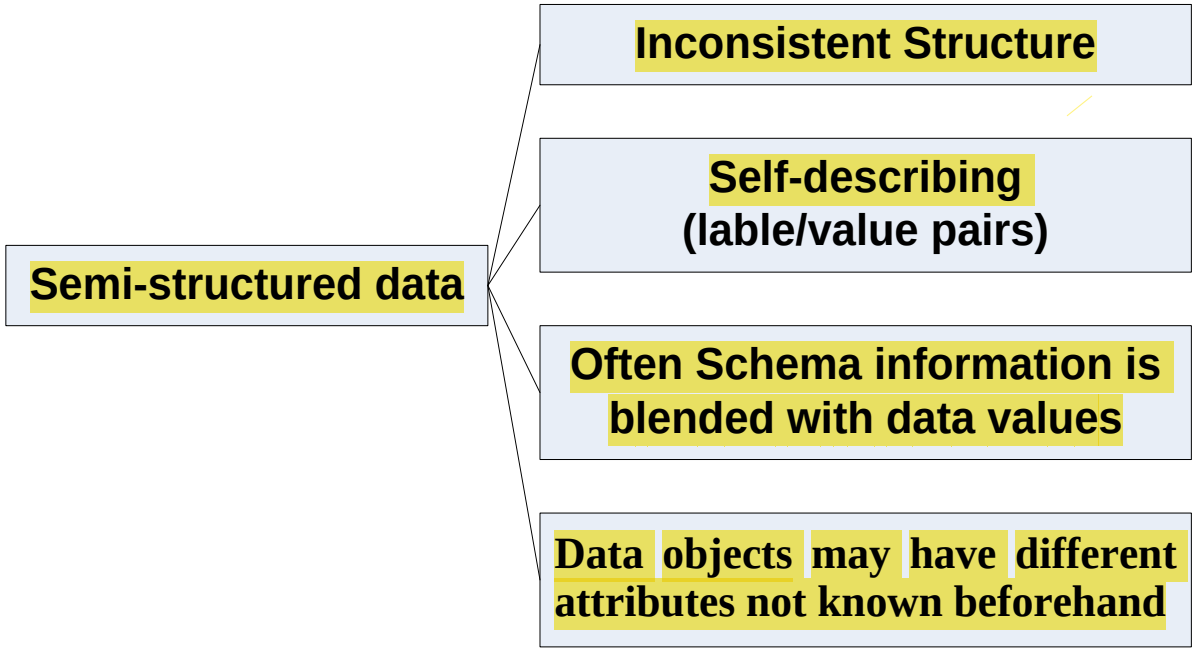


Fig.: Characteristics of semi-structured data

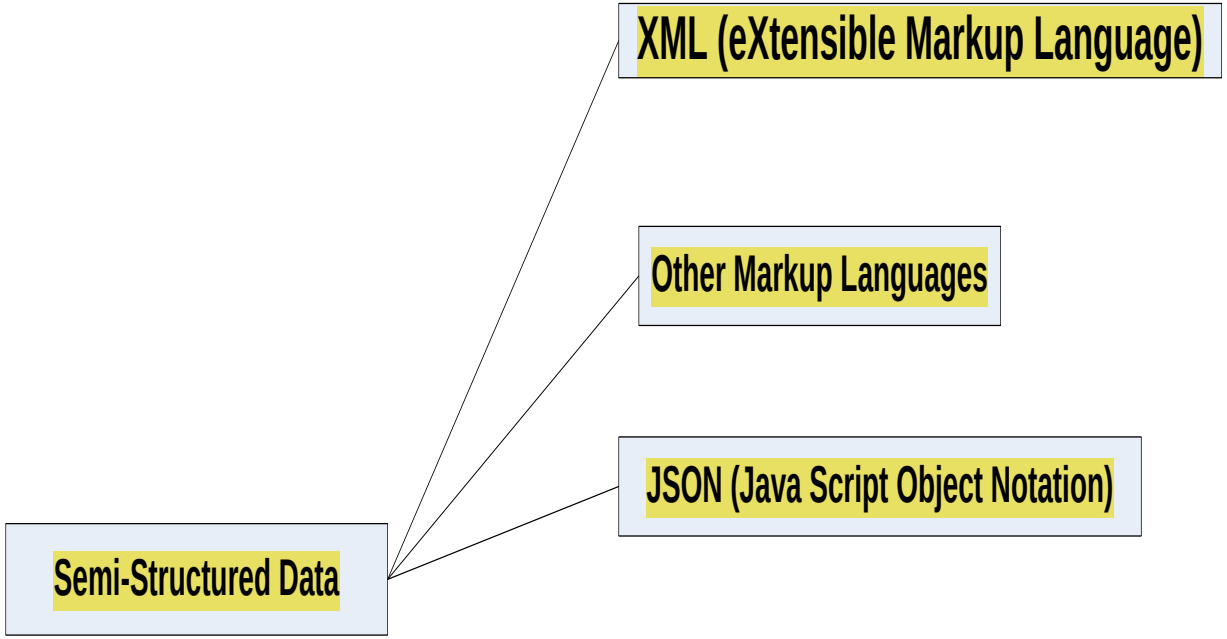


Fig.: Sources of semi-structured data

1.1 Classification of Digital Data

- Unstructured Data: Unstructured data does not conform to any pre-defined data model.
- Issues with unstructured data:

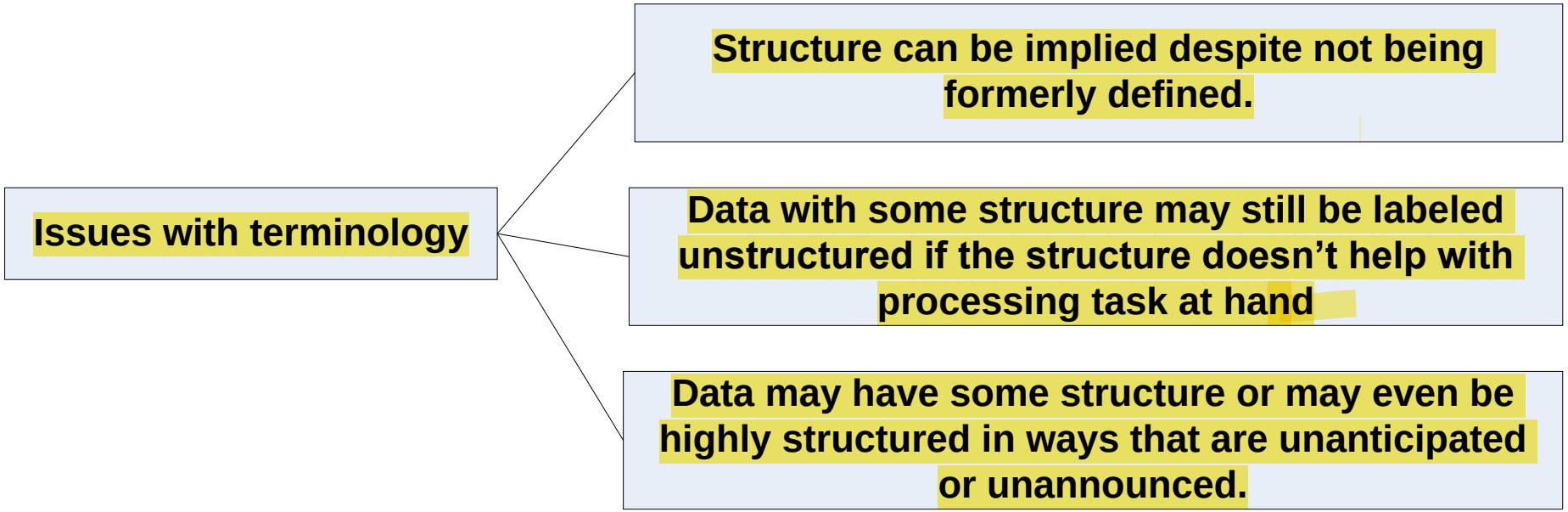


Fig.: Issues with terminology of unstructured data

1.1 Classification of Digital Data

- Unstructured Data:

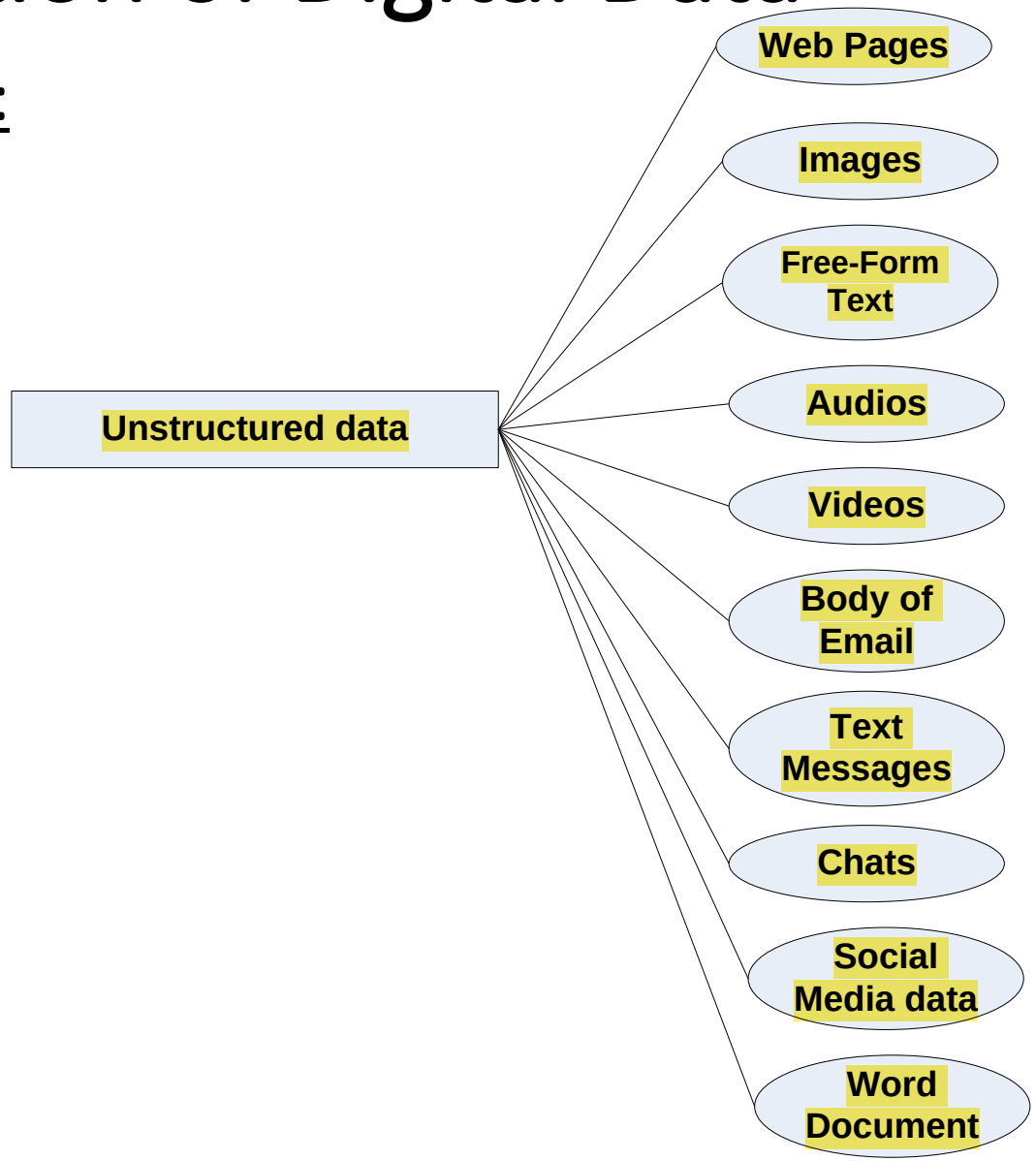


Fig.: Sources of unstructured data

1.1 Classification of Digital Data

- Issues with Unstructured Data: Although unstructured data is known not to conform to a pre-defined data model or be organized in a pre-defined manner, there are incidents wherein the structure of the data (placed in the unstructured category) can still be implied.
- How to deal with unstructured data?

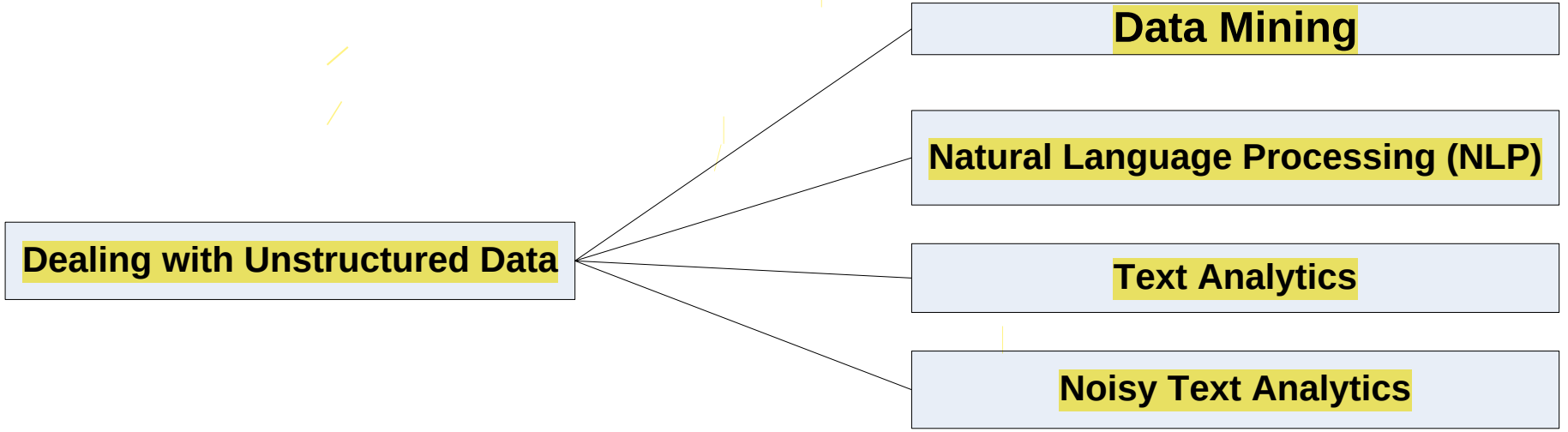


Fig.: Dealing with unstructured data

1.1 Classification of Digital Data

- How to deal with unstructured data?
- Data Mining:
  - Associate rule mining.
  - Regression analysis.
  - Collaborative filtering.
- Text analytics or text mining.
- Natural Language Processing (NLP).
- Noisy text analytics.

2. Introduction to Big Data

• 2.1 Characteristics of Data:

- Data has three key characteristics.

1. Composition: The composition of the data deals with the structure of the data, source of the data, the granularity, the types, and the nature of the data.
2. Condition: The condition of the data deals with the state of the data. That is "can one use this data as it is for analysis?" or "does it require cleansing for further enhancement and enrichment?"
3. Context: The context of the data deals with "where has this data been generated?" "why was this data generated?" "how sensitive is this data?" "what are the events associated with this data?" and so on.

- Small data is about certainty, and big data is about complexity.

2.2 Evolution of Big Data

- 1970s and before was the era of mainframes. The data was essentially primitive and structure. Relational databases evolved in 1980s and 1990s. The era of data intensive application. The WWW and IoT have led to an onslaught of structured, unstructured, and multimedia data.

Table 2.1 The evolution of big data

	Data Generation and Storage	Data Utilization	Data Driven
Complex and Unstructured			Structured data, unstructured data, multimedia data
Complex and Relational		Relational databases: Data-intensive applications	
Primitive and Structured	Mainframes: Basic data storage		
	1970s and before	**Relational (1980s and 1990s)**	**2000s and beyond**

2.3 Definition of Big Data

- Definition: Big Data is high-volume, high-velocity, and high-variety information assets that demand cost effective, innovative forms of information processing for enhanced insight and decision making.
- Part I of the definition talks about voluminous data that may have great variety and will require a good speed/pace for storage, preparation, processing, and analysis.
- Part II talks about embracing new techniques and technologies to capture, store, process, persist, integrate, and visualize high volume, velocity, and variety data.
- Part III talks about deriving deeper, richer, and meaningful insights and then using these insights to make faster and better decisions to gain business value.

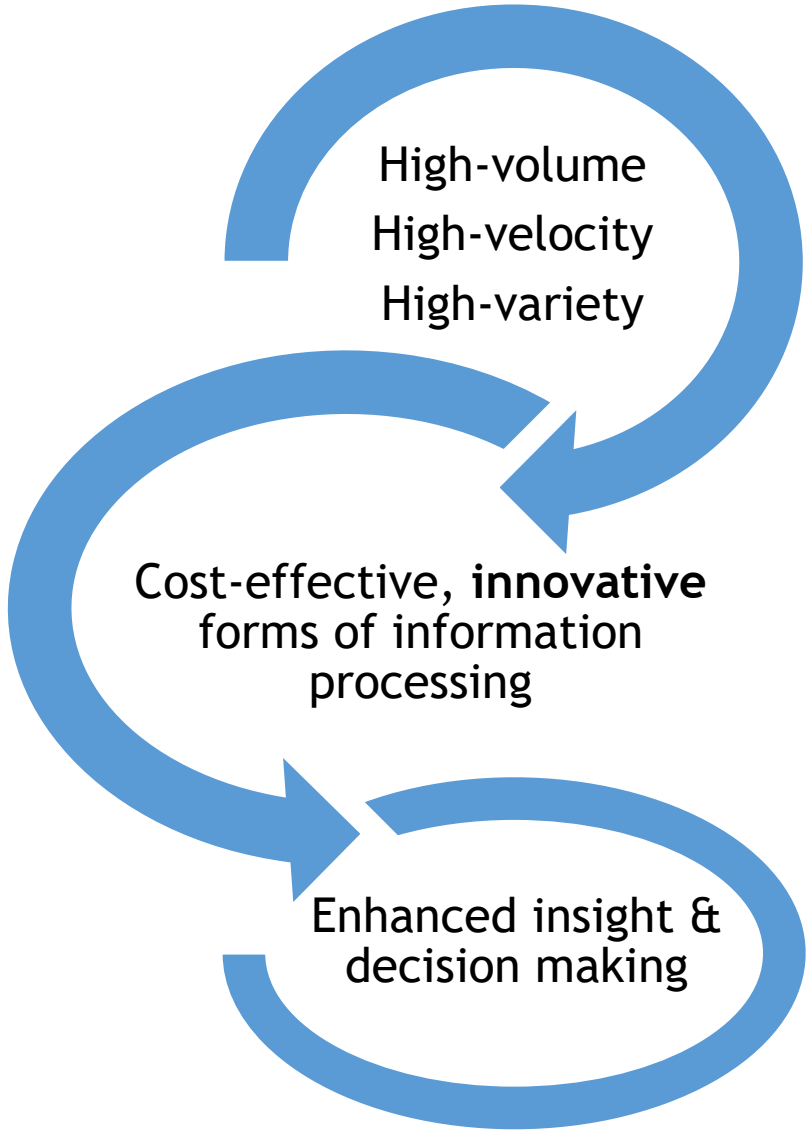


Fig.: Definition of big data

2.4 Challenges with Big Data

- Following are few challenges with big data:
  - Data today is growing at an exponential rate. The key question here are: “will all this data be useful for analysis?”, “do we work with all this data or subset of it?”, “ how will we separate the knowledge from the noise?”, etc.
  - Cloud computing is the answer to managing infrastructure for big data as far as cost-efficiency, elasticity, and easy upgrading/downgrading is concerned. This further complicates the decision to host big data solutions outside the enterprise.
  - The other challenge is to decide on the period of retention of big data. How long should one retain this data?
  - There is a dearth of skilled professionals who possess a high level proficiency in data sciences that is vital in implementing big data solutions.
  - Other challenges with respect to capture, storage, preparation, search, analysis, transfer, security, and visualization of big data.
  - Visualization of the data as business experts are concerned is difficult.

2.4 Challenges with Big Data

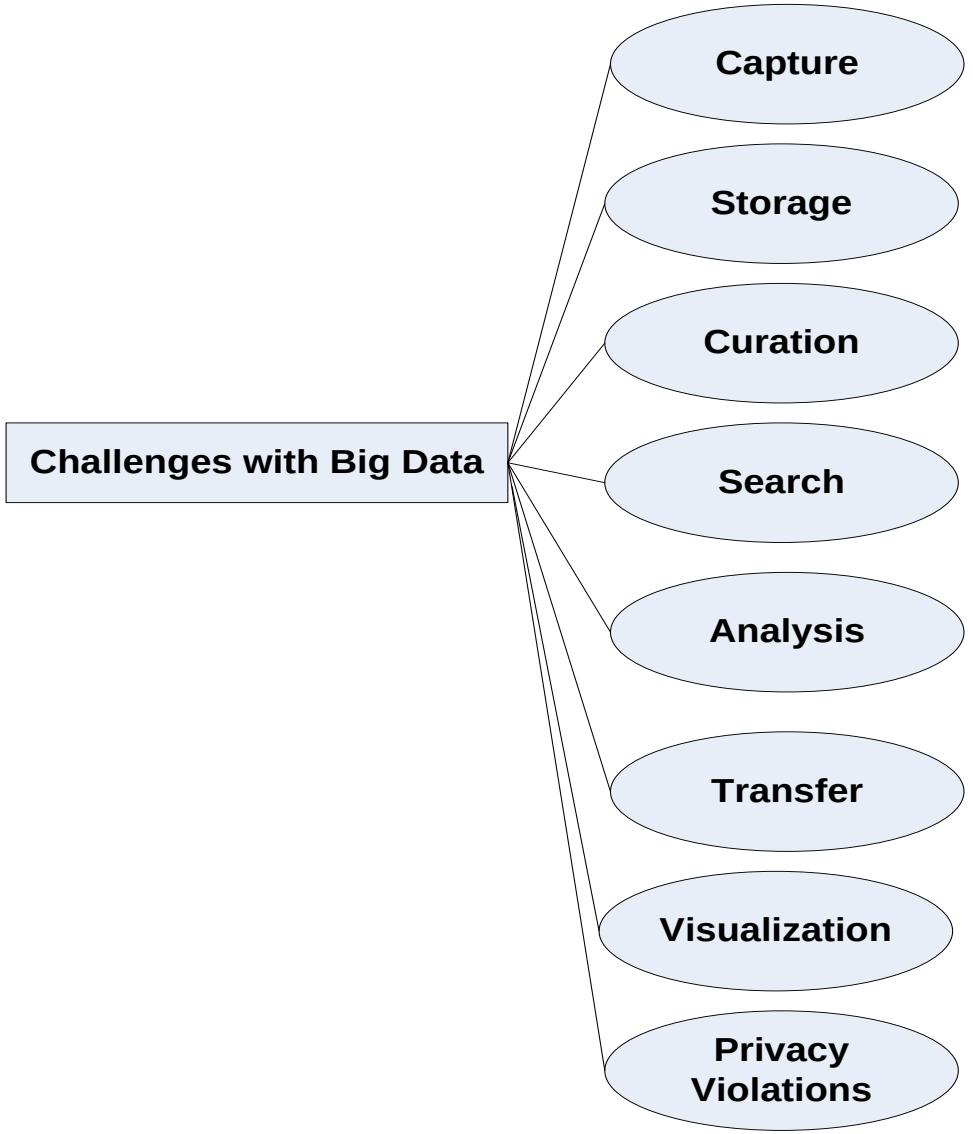


Fig.: Challenges with big data

2.5 What is Big Data?

- Big data is the data that is big in volume, velocity, and variety.
- **2.5.1 Volume:**

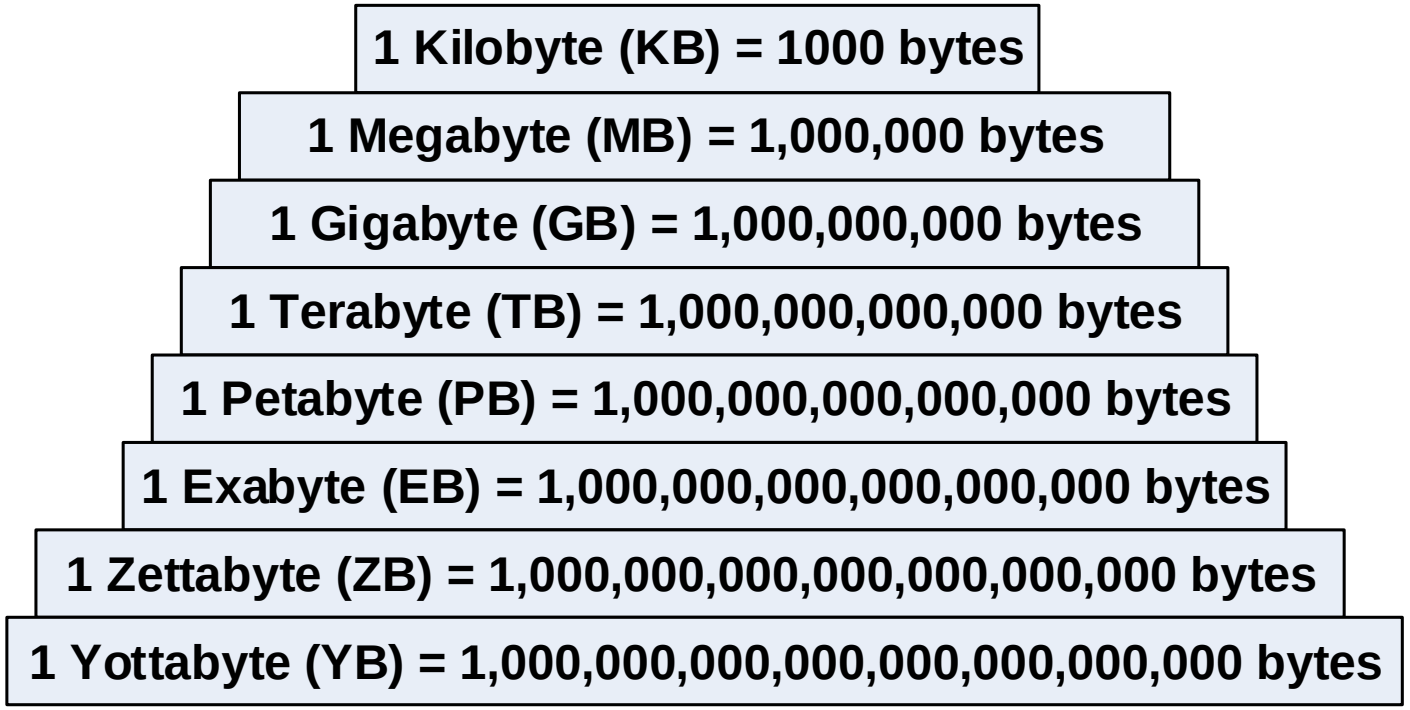


Fig.: A mountain of data

2.5 What is Big Data?

- **2.5.1.1 Where does this data get generated:**
- There are a multitude of sources for big data. An XLS, a DOC, a PDF, etc. is unstructured data, a video on YouTube, a chat conversation on Internet Messenger, a customer feedback form on an online retail website is unstructured data, a CCTV coverage, a weather forecast report is unstructured data too.

1. Typical internal data sources: It is a data which is present within an organization's firewall.
   - Data storage: File systems, SQL, NoSQL, and so on.
   - Archives: Archives of scanned documents, paper archives, customer correspondence records, patients' health records, students' admission records, students' assessment records, and so on.
2. External data sources: Data residing outside an organization's firewall.
   - Public web: Wikipedia, weather, regulatory, compliance, census, etc.

2.5 What is Big Data?

- **2.5.1.1 Where does this data get generated:**

3. Both (internal + external data sources)
   - Sensor data: Car sensors, smart electric meters, office buildings, air conditioning units, refrigerators, and so on.
   - Machine log data: Event logs, application logs, business process logs, audit logs, clickstream data, etc.
   - Social media: Twitter, blogs, Facebook, LinkedIn, YouTube, Instagram, etc.
   - Business apps: ERP, CRM, HR, Google Docs, and so on.
   - Media: Audio, video, image, podcast, etc.
   - Docs: Comma separated value (CSV), word documents, PDF, XLS, PPT, and so on.

2.5 What is Big Data?

- **2.5.1.1 Where does this data get generated:**

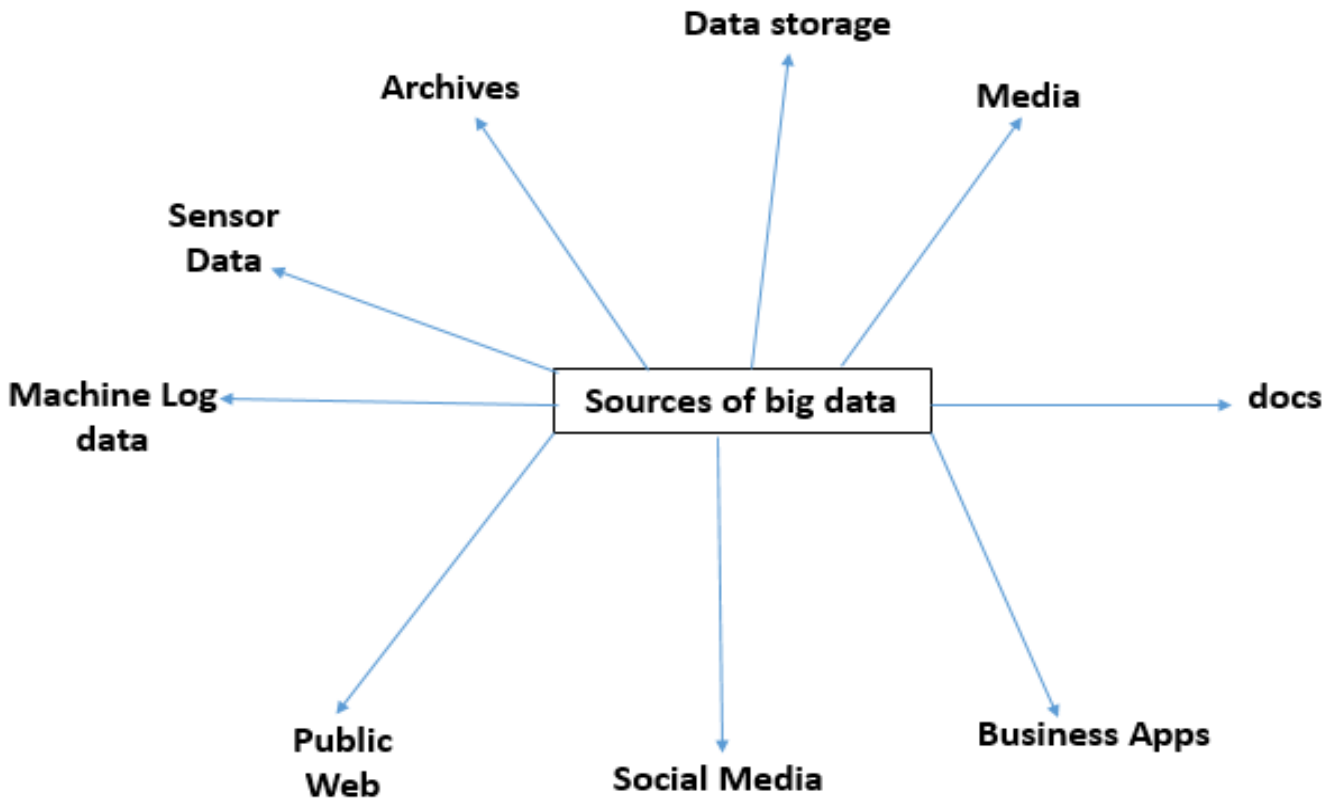


Fig.: Sources of Big Data

2.5 What is Big Data?

- **2.5.2 Velocity:**

Batch → Periodic → Near real time → Real-time processing

- **2.5.3 Variety:**
- Variety deals with a wide range of data types and sources of data. We will study this under three categories: Structured, semi-structured, and unstructured.

2.6 Other characteristics of Data which are not definitional traits of Big Data

- Few other characteristics of Big Data are as follows:

1. Veracity and validity: Veracity refers to biases, noise, and abnormality in data. The key question here is: “Is all the data that is being stored, mined, and analyzed meaningful and pertinent to the problem under consideration?” Validity refers to the accuracy and correctness of the data. Any data that is picked up for analysis needs to be accurate. It is not just true about big data.
2. Volatility: Volatility of data deals with, how long is the data valid? And how long should it be stored? There is some data that is required for long-term decisions and remains valid for longer periods of time. However, there are also pieces of data quickly become obsolete minutes after their generation.
3. Variability: Data flows can be highly inconsistent with periodic peaks.

2.7 Why Big Data?

- The more data we have for analysis, the greater will be the analytical accuracy and also the greater would be the confidence in our decisions based on these analytical findings.
- This will entail a greater positive impact in terms of enhancing operational efficiencies, reducing cost and time, and innovating on new products, new services, and optimizing existing services.

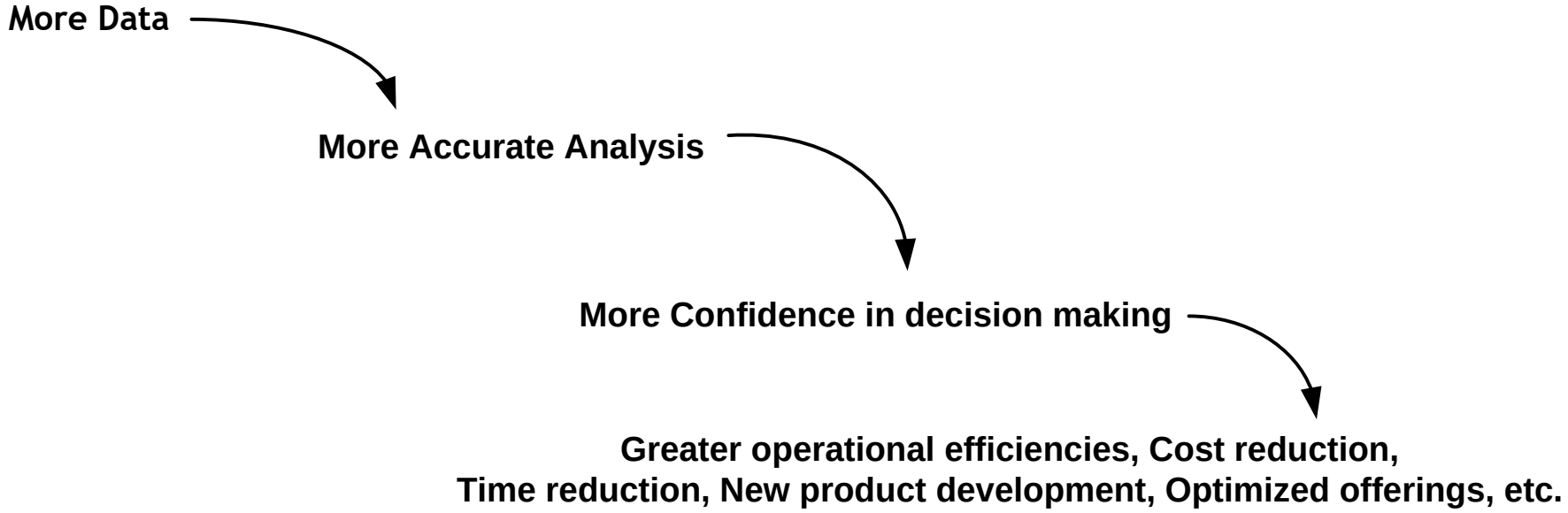


Fig.: Why Big Data?

2.9 Traditional Business Intelligence (BI) versus Big Data

- Some of the differences that one encounters dealing with traditional BI and big data.

1. In traditional BI environment, data resides in a central server whereas in big data environment, data resides in a distributed file system. The distributed file system scales by scaling in or out horizontally as compared to typical database server that scales vertically.
2. In traditional BI environment, data is analyzed in offline mode whereas in big data environment data is analyzed in both real time as well as offline mode.
3. Traditional BI is about structured data and it is here that data is taken to processing functions (move data to code) whereas big data is about variety, and here the processing functions are taken to the data (move code to data).

2.10 A Typical Data Warehouse Environment

- In a typical DW environment, data is collected from multiple disparate sources located in the same geography or different geographies.
- This data is then integrated, cleaned up, transformed, and standardized through the process of extraction, transformation, and loading before loading it to a data warehouse.
- A host of market leading BI and analytics tools are then used to enable decision making from the use of ad-hoc queries, SQL, enterprise dashboard, data mining, etc.

2.10 A Typical Data Warehouse Environment

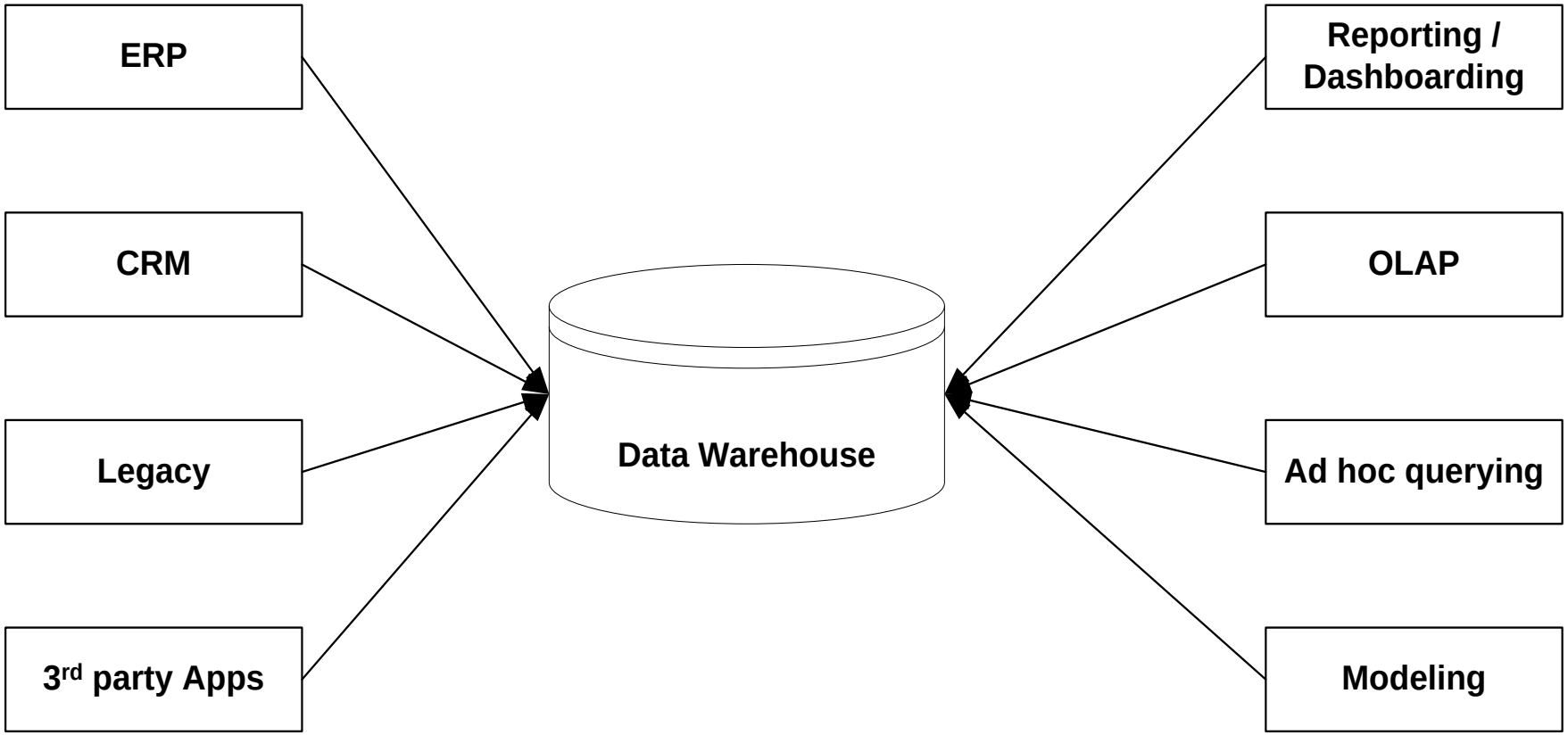


Fig.: A typical data warehouse environment

2.11 A Typical Hadoop Environment

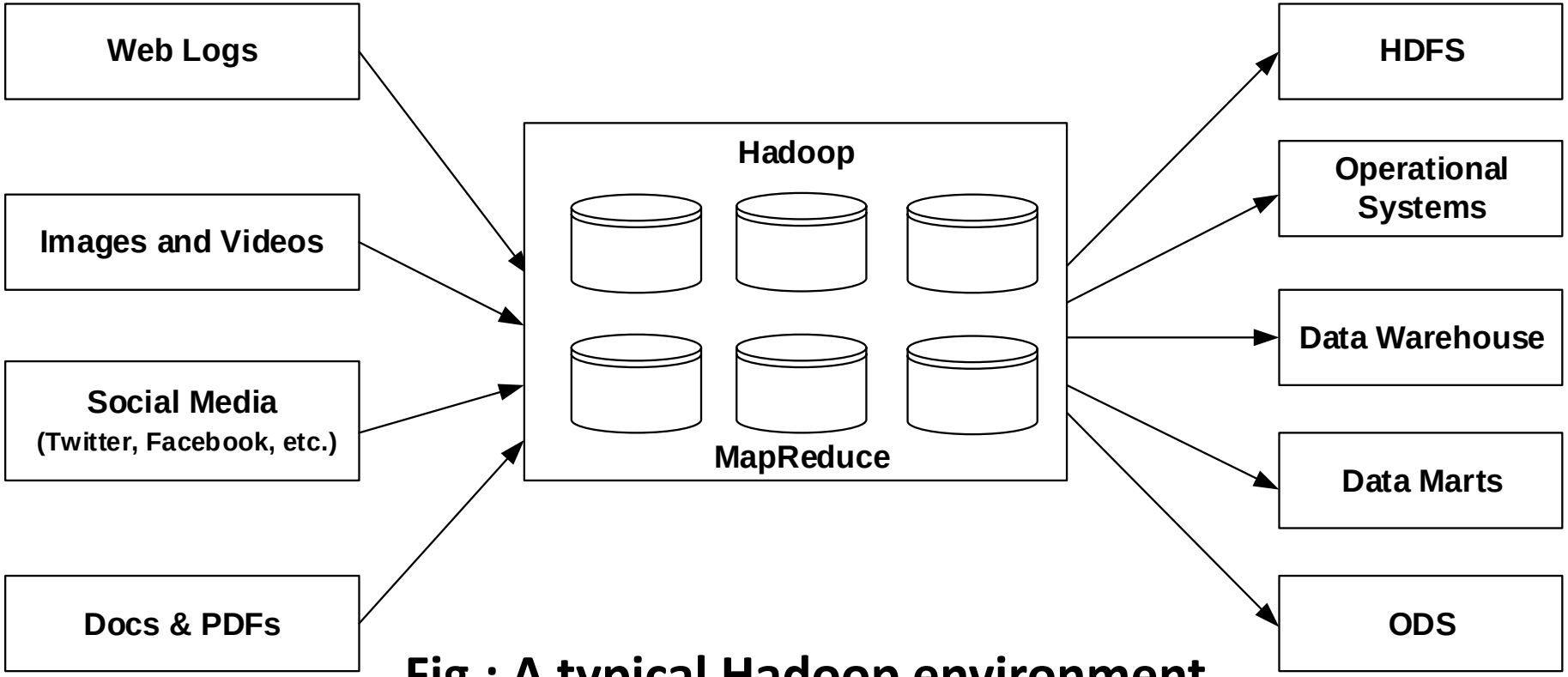


Fig.: A typical Hadoop environment

- Hadoop takes care of storage and processing using the following:

a) HDFS (Hadoop Distributed File System) (distributed storage)

b) MapReduce (distributed processing)

3.2 What is Big Data Analytics?

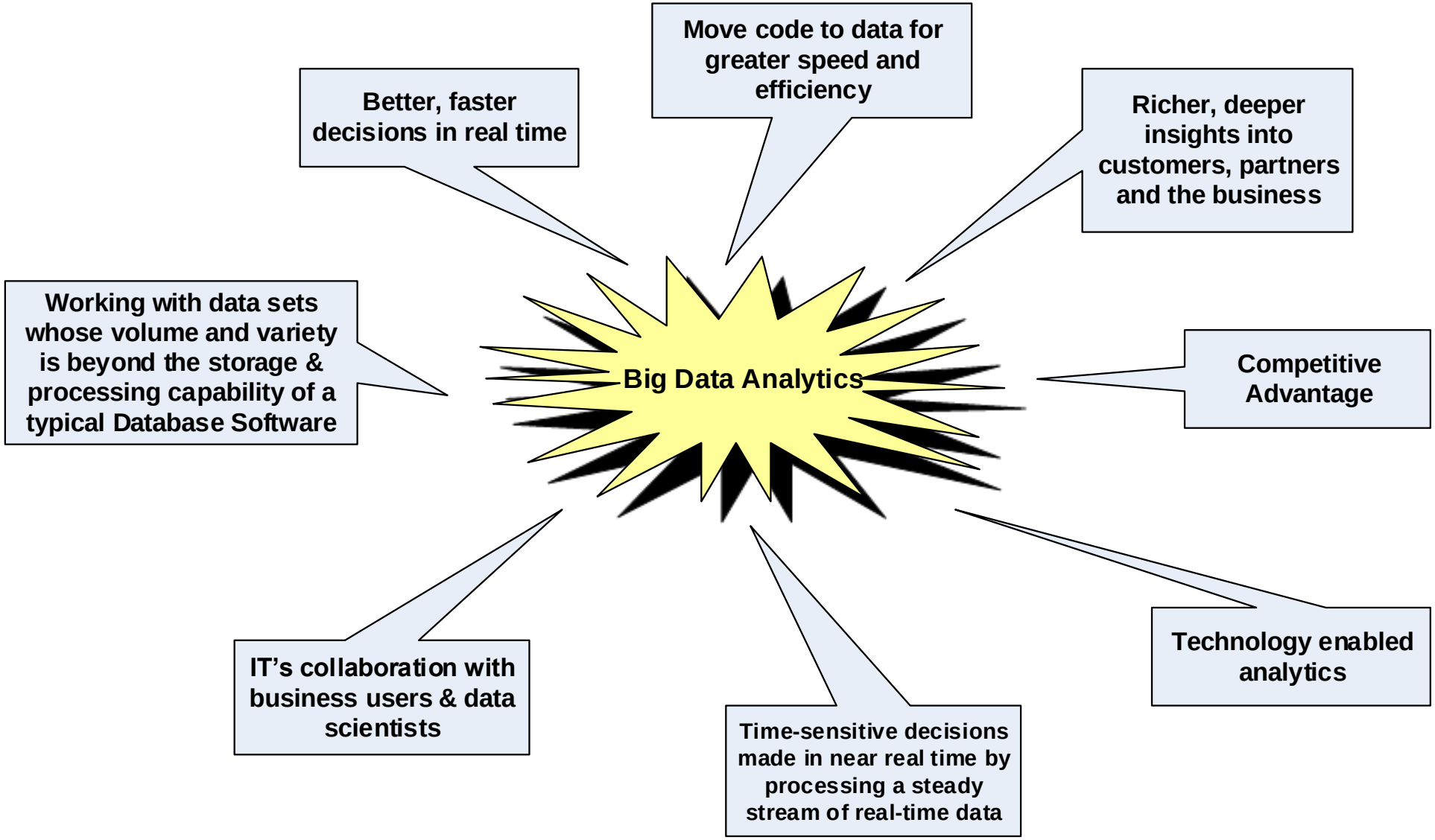


Fig.: What is Big Data Analytics?

3.2 What is Big Data Analytics?

- Big Data Analytics is:

1. Technology-enabled analytics: Leading vendors such as IBM, Tableau, R Analytics, etc. to help process and analyze the big data.
2. About gaining a meaningful, deeper, and richer insight into your business to steer it in the right direction, understanding the customer's demographics to cross-sell and up-sell to them, better leveraging the services of your vendors and suppliers, etc.
3. Stage for better and faster decision making.
4. A tight handshake between the communities of business users, IT and data scientists.
5. Working with datasets whose volume and variety exceed the current storage and processing capabilities and infrastructure of the enterprise.
6. About moving code to data. This makes perfect sense as the program for distributed processing is tiny compared to the data.

3.5 Classification of Analytics

- Two types of thoughts:

1. Classification of analytics- Basic, Operationalized, Advanced, and Monetized.
2. Classification of analytics- Analytics 1.0, Analytics 2.0, and Analytics 3.0.

- 3.5.1 First School of Thought:

1. Basic Analytics: This primarily is slicing and dicing of data to help with basic business insights. This is about reporting on historical data, basic visualization, etc.
2. Operationalized Analytics: It is operationalized analytics if it gets woven into the enterprise's business processes.
3. Advanced Analytics: This largely is about forecasting for the future by way of predictive and prescriptive modeling.
4. Monetized Analytics: This is analytics in use to derive direct business revenue.

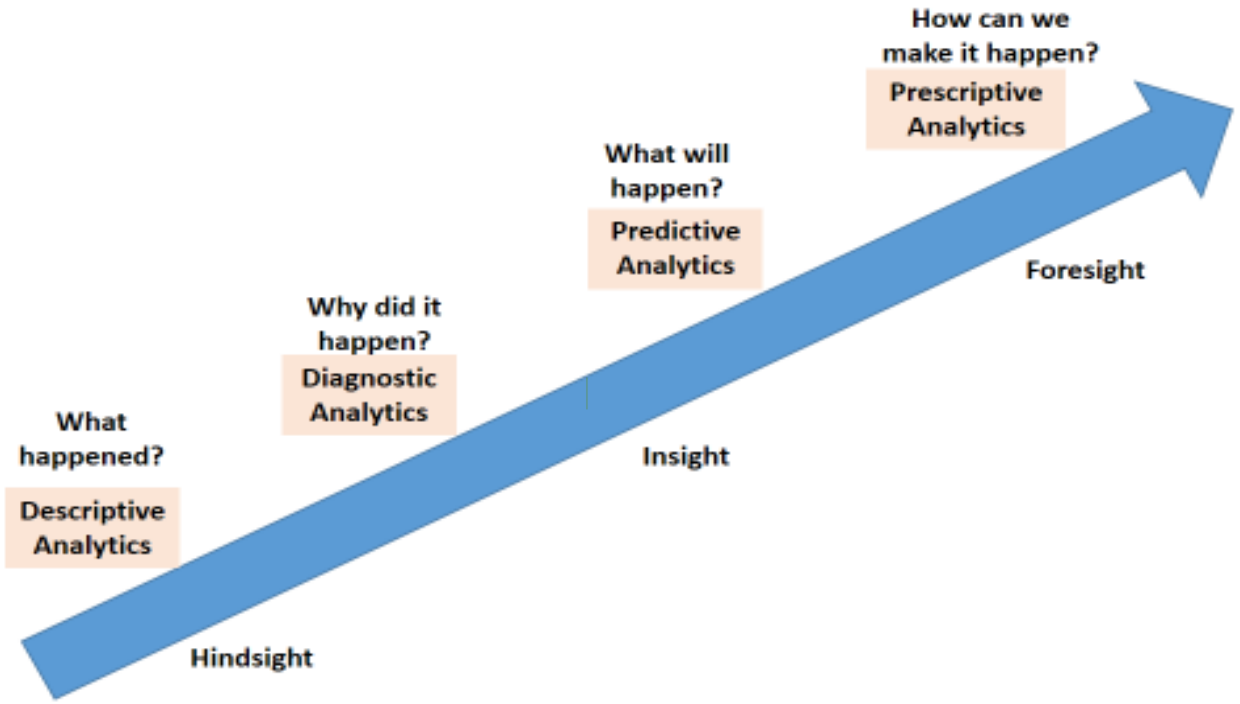
3.5.2 Second School of Thought:

Table 3.1 Analytics 1.0, 2.0, and 3.0

Analytics 1.0	Analytics 2.0	Analytics 3.0
Era: mid 1950s to 2009	**2005 to 2012**	**2012 to present**
Descriptive statistics (report on events, occurrences, etc. of the past)	Descriptive statistics + predictive statistics (use data from the past to make predictions for the future)	Descriptive + predictive + prescriptive statistics (use data from the past to make prophecies for the future and at the same time make recommendations to leverage the situation to one's advantage)
Key questions asked: What happened? Why did it happen?	Key questions asked: What will happen? Why will it happen?	Key questions asked: What will happen? When will it happen? Why will it happen? What should be the action taken to take advantage of what will happen?
Data from legacy systems, ERP, CRM, and 3rd party applications.	Big data	A blend of big data and data from legacy systems, ERP, CRM, and 3rd party applications.
Small and structured data sources. Data stored in enterprise data warehouses or data marts.	Big data is being taken up seriously. Data is mainly unstructured, arriving at a much higher pace. This fast flow of data entailed that the influx of big volume data had to be stored and processed rapidly, often on massive parallel servers running Hadoop.	A blend of big data and traditional analytics to yield insights and offerings with speed and impact.
Data was internally sourced.	Data was often externally sourced.	Data is both being internally and externally sourced.
Relational databases	Database appliances, Hadoop clusters, SQL to Hadoop environments, etc.	In memory analytics, in database processing, agile analytical methods, machine learning techniques, etc.

3.5 Classification of Analytics

- 3.5.2 Second School of Thought:



- Analytics 1.0 → deals with historical data to report on events, occurrences of the past.
- Analytics 2.0 → helps to predict what will happen in future
- Analytics 3.0 → is about predicting what will happen and to best leverage the situation when it happens.

3.8 Why is Big Data Analytics Important

- Various approaches to analyze the data are as follows:

1. Reactive – Business Intelligence: BI allows the business to make faster and better decisions by providing the right information to the right person at the right time in the right format. It is about analysis of the past or historical data and then displaying the findings of the analysis or reports in the form of enterprise dashboards, alerts, notifications, etc.
2. Reactive – Big Data Analytics: Here the analysis is done on huge datasets but the approach is still reactive as it is still based on static data.
3. Proactive – Analytics: This is to support futuristic decision making by the use of data mining, predictive modeling, text mining, and statistical analysis. Analysis will be on traditional database.
4. Proactive – Big Data Analytics: This is sieving through terabytes, petabytes, exabytes of information to filter out the relevant data to analyze. This also includes high performance analytics to gain rapid insights from big data and the ability to solve complex problems using more data.

3.12 Terminologies used in Big Data Environment

1. In-Memory Analytics.
2. In-Database Processing.
3. Symmetric Multiprocessor System (SMP).
4. Massively Parallel Processing.
5. Difference between parallel and distributed systems.
6. Shared nothing architecture.
7. CAP theorem (Consistency, Availability, and Partition tolerance).

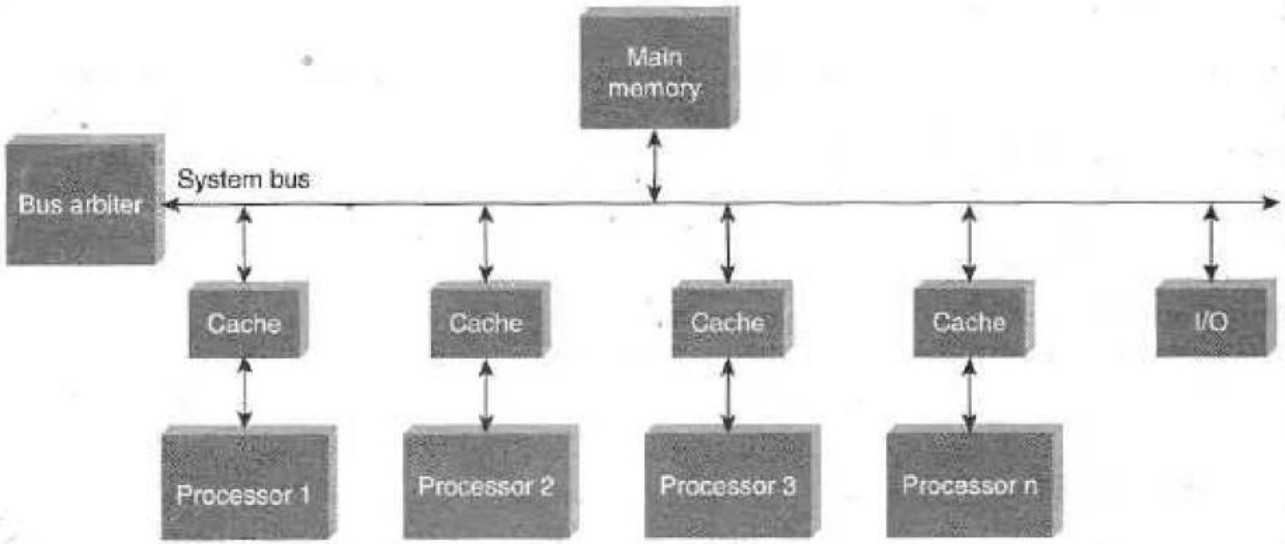


Figure 3.9 Symmetric Multiprocessor System.

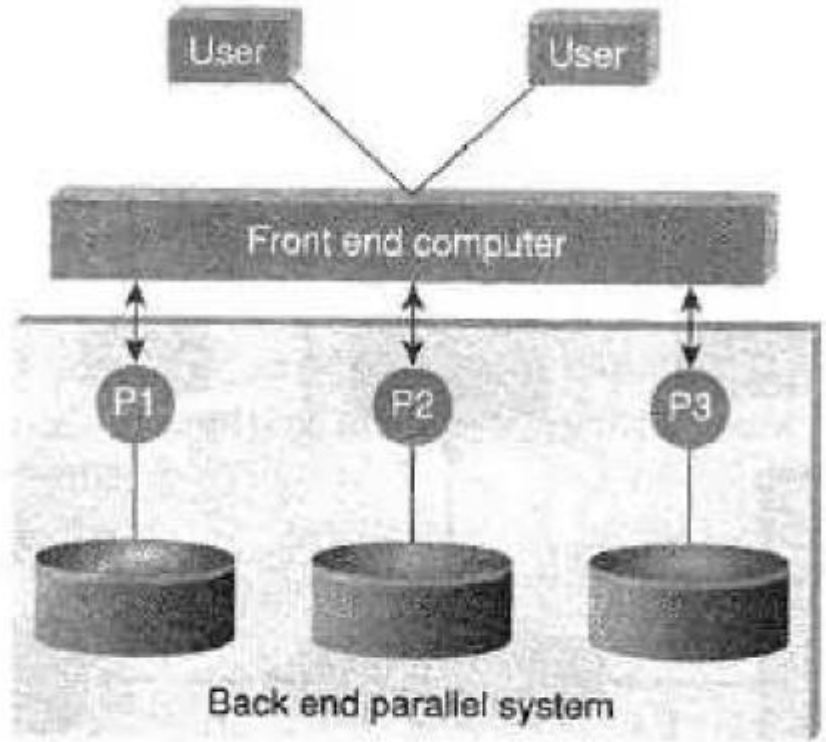


Figure 3.10 Parallel system.

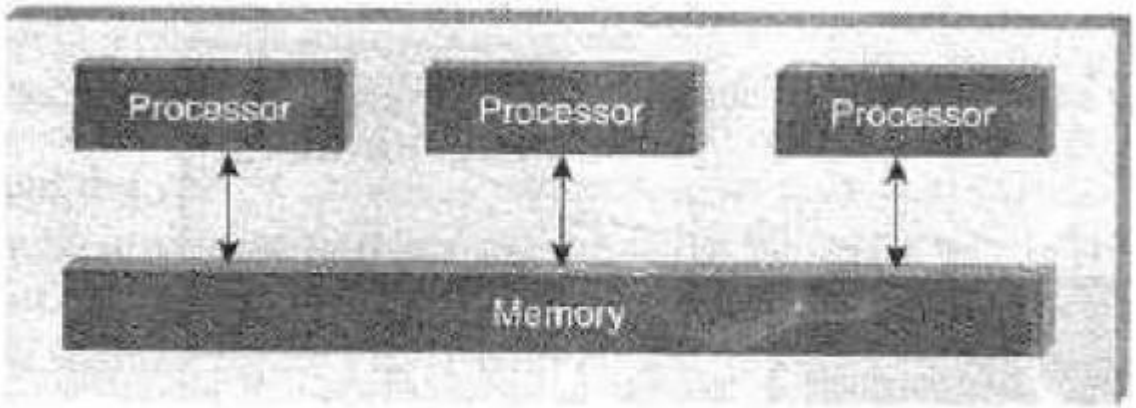


Figure 3.11 Parallel system.

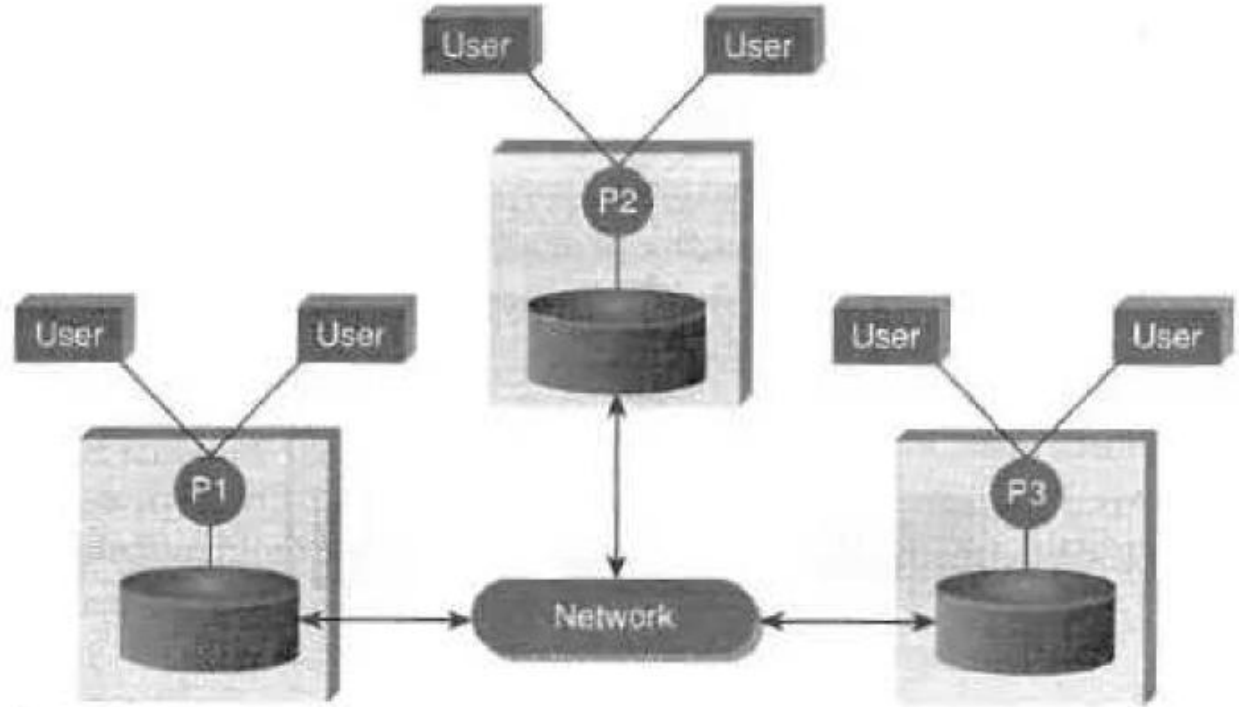


Figure 3.12 Distributed system.

4.1 NoSQL (Not Only SQL)

- Few features of NoSQL databases are as follows:
  - They are open source.
  - They are non-relational.
  - They are distributed.
  - They are schema-less.
  - They are cluster friendly.
- **4.1.1 Where is it Used?**
- NoSQL databases are widely used in big data and other real-time web applications.
- NoSQL databases is used to stock log data which can then be pulled for analysis. Likewise it is used to store social media data and all such data which cannot be stored and analyzed comfortably in RDBMS.

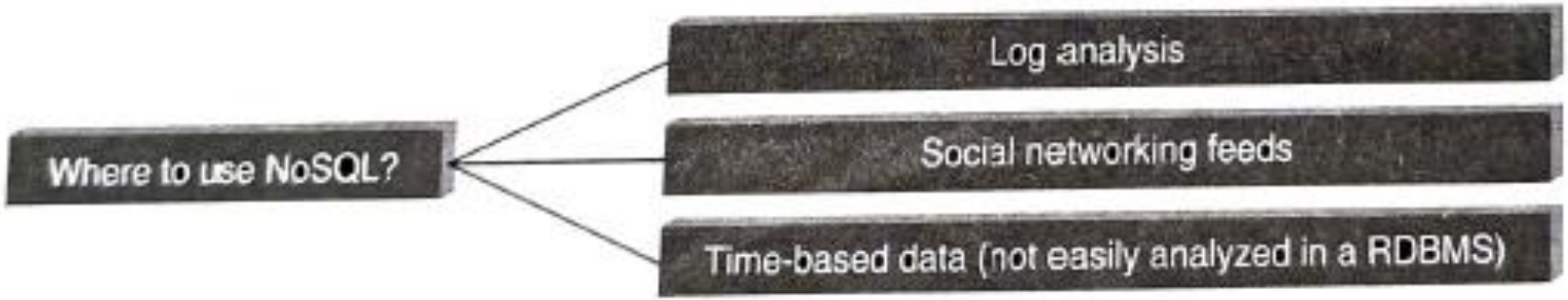


Figure 4.1 Where to use NoSQL?

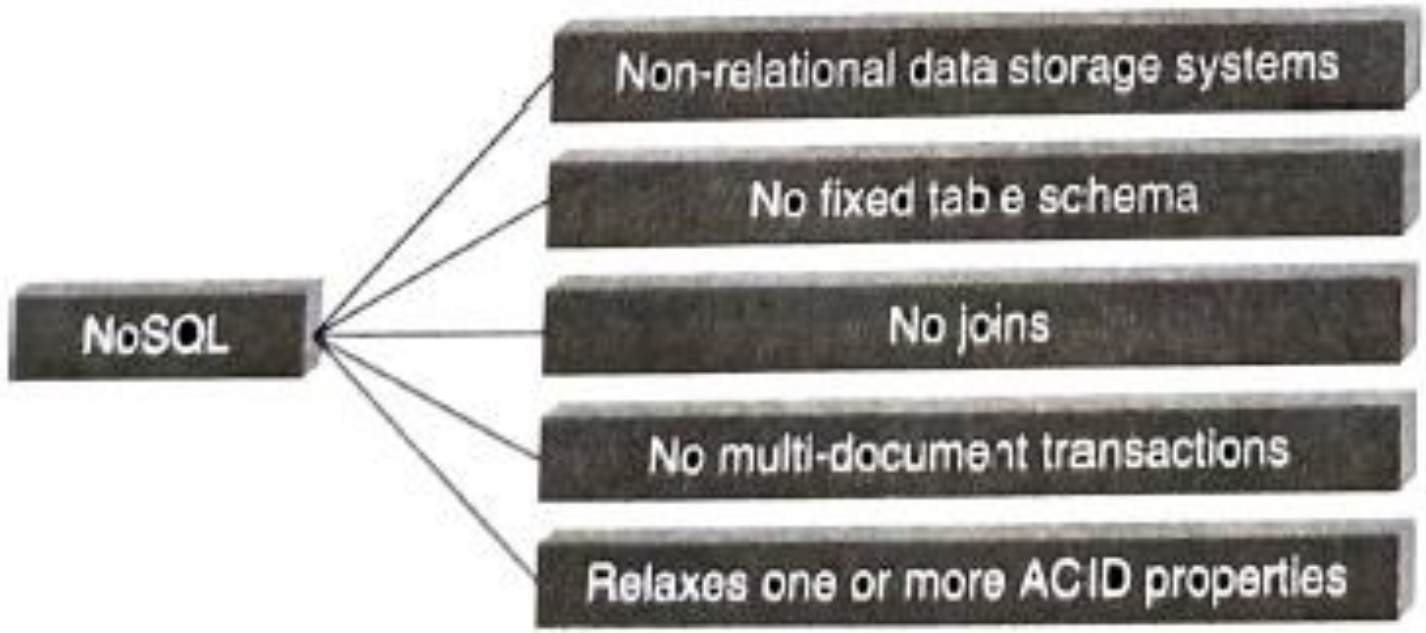


Figure 4.2 What is NoSQL?

4.1 NoSQL (Not Only SQL)

- **4.1.2 What is it?**
- NoSQL stands for Not Only SQL. These are non-relational, open source, distributed databases. They are hugely popular today owing to their ability to scale out or scale horizontally and the adeptness at dealing with a rich variety of data: structured, semi-structured and unstructured data.
  1. Non-relational.
  2. Distributed.
  3. Offer no support for ACID properties.
  4. Provide no fixed table schema.
- **4.1.3 Types of NoSQL Databases?**
- Classified into 2 types:

1. Key-value or the big hash table.
2. Schema-less

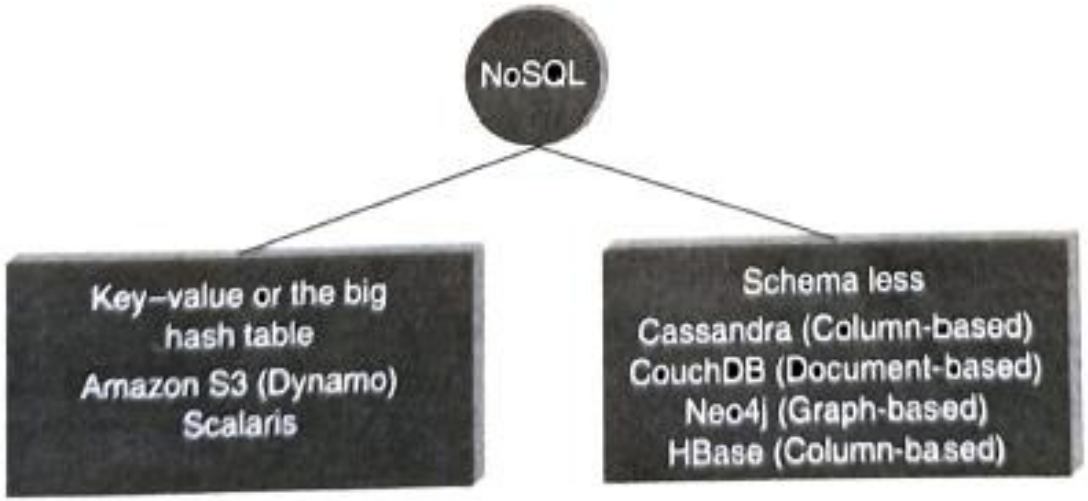


Figure 4.3 Types of NoSQL databases.

4.1 NoSQL (Not Only SQL)

- Key-value: It maintains a big hash table of keys and values. For example, Dynamo, Redis, Riak, etc.

- Document It maintains data in collections constituted of documents. For example, MongoDB, Apache CouchDB, Couchbase, MarkLogic, etc.

- Column: Each storage block has data from only one column. For example: Cassandra, HBase, etc.

- Graph: They are also called nerwork database. A graph stores data in nodes. For cxample, Neod HyperGraphDB, etc.

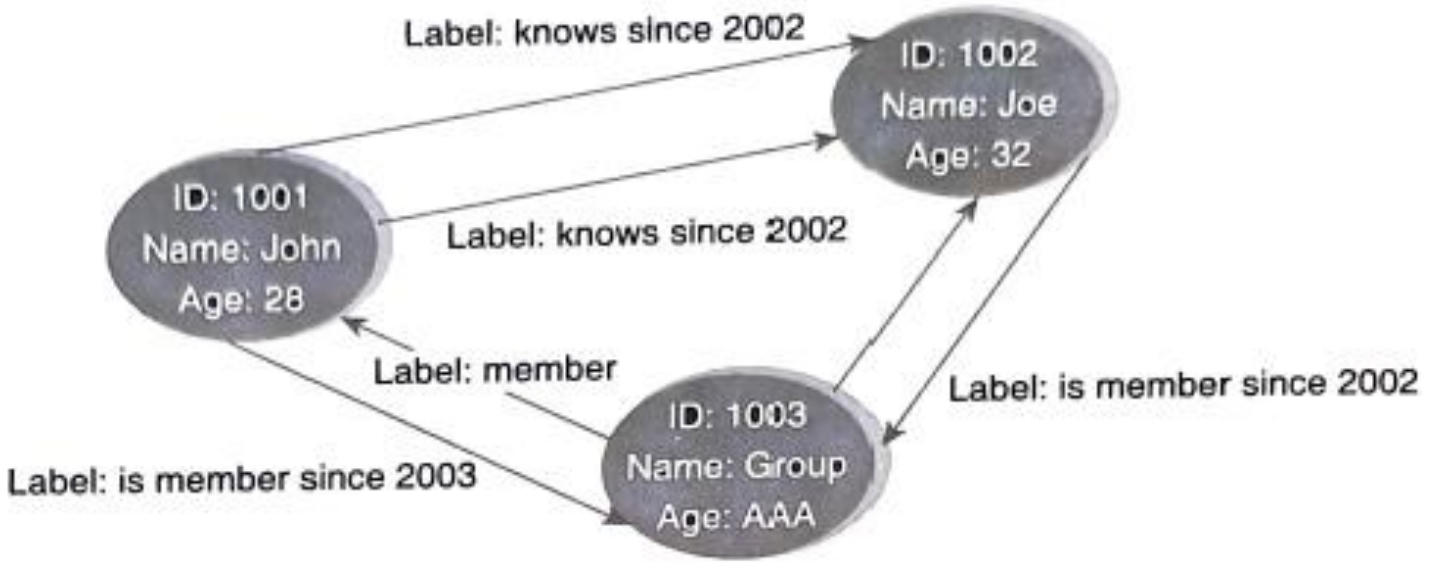
Sample Key-Value Pair in Key-Value Database

Key	Value
First Name	Simmonds
Last Name	David

Sample Document in Document Database

```
{
"Book Name":    "Fundamentals of Business Analytics",
"Publisher":    "Wiley India",
"Year of Publication":    "2011"
}
```

Sample Graph in Graph Database



4.1 NoSQL (Not Only SQL)

- **4.1.4 Why NoSQL?**

1. It has scale out architecture instead of the monolithic architecture of relational databases.
2. It can house large volumes of structured, semi-structured, and unstructured data.
3. Dynamic schema: NoSQL database allows insertion of data without a pre-defined schema. In other words, it facilitates application changes in real time, which thus supports faster development, easy code integration, and requires less database administration.
4. Auto-sharding: It automatically spreads data across an arbitrary number of servers. The application in question is more often not even aware of the composition of the server pool. It balances the load of data and query on the available servers; and if and when a server goes down, it is quickly replaced without any major activity disruptions.
5. Replication: It offers good support for replication which in turn guarantees high availability, fault tolerance, and disaster recovery.

4.1 NoSQL (Not Only SQL)

- **4.1.5 Advantages of NoSQL?**

1. Can easily scale up and down: NoSQL database supports scaling rapidly and elastically and even allows to scale to the cloud.

   (a) Cluster scale: It allows distribution of database across 100+ nodes often in multiple data centers.

   (b) Performance scale: It sustains over 100,000+ database reads and writes per second.

   (c) Data scale: It supports housing of 1 billion+ documents in the database.

2. Doesn't require a pre-defined schema: NoSQL does not require any adherence to pre-defined schema. It is pretty flexible. For example, if we look at MongoDB, the documents (equivalent of records in RDBMS) in a collection (equivalent of table in RDBMS) can have different sets of key-value pairs.

   {id:101, ‘BookName’:"Fundamentals of Business Analytics', "AuthorName": "Seema Acharya", "Publisher": "Wiley India"}

4.1 NoSQL (Not Only SQL)

- **4.1.5 Advantages of NoSQL?**

3. Cheap, easy to implement: Deploying NoSQL properly allows for all of the benefits of scale, high availability, fault tolerance, etc. while also lowering operational costs.
4. Relaxes the data consistency requirement: NoSQL databases have adherence to CAP theorem (Consistency, Availability, and Partition tolerance). Most of the NoSQL databases compromise on consistency in favor of availability and partition tolerance. However, they do go for eventual consistency.
5. Data can be replicated to multiple nodes and can be partitioned: There are two terms that we will discuss here:

   (a) Sharding: Sharding is when different pieces of data are distributed across multiple servers. NoSQL databases support auto-sharding; this means that they can natively and automatically spread data across an arbitrary number of servers, without requiring the application to even be aware of the composition of the server pool. Servers can be added or removed from the data layer without application downtime. This would mean that data and query load are automatically balanced across servers, and when a server goes down, it can be quickly and transparently replaced with no application disruption.

   (b) Replication: Replication is when multiple copies of data are stored across the cluster and even across data centers. This promises high availability and fault tolerance.

4.1 NoSQL (Not Only SQL)

- **4.1.6 What we miss with NoSQL?**
- **4.1.7 Use of NoSQL in Industry**

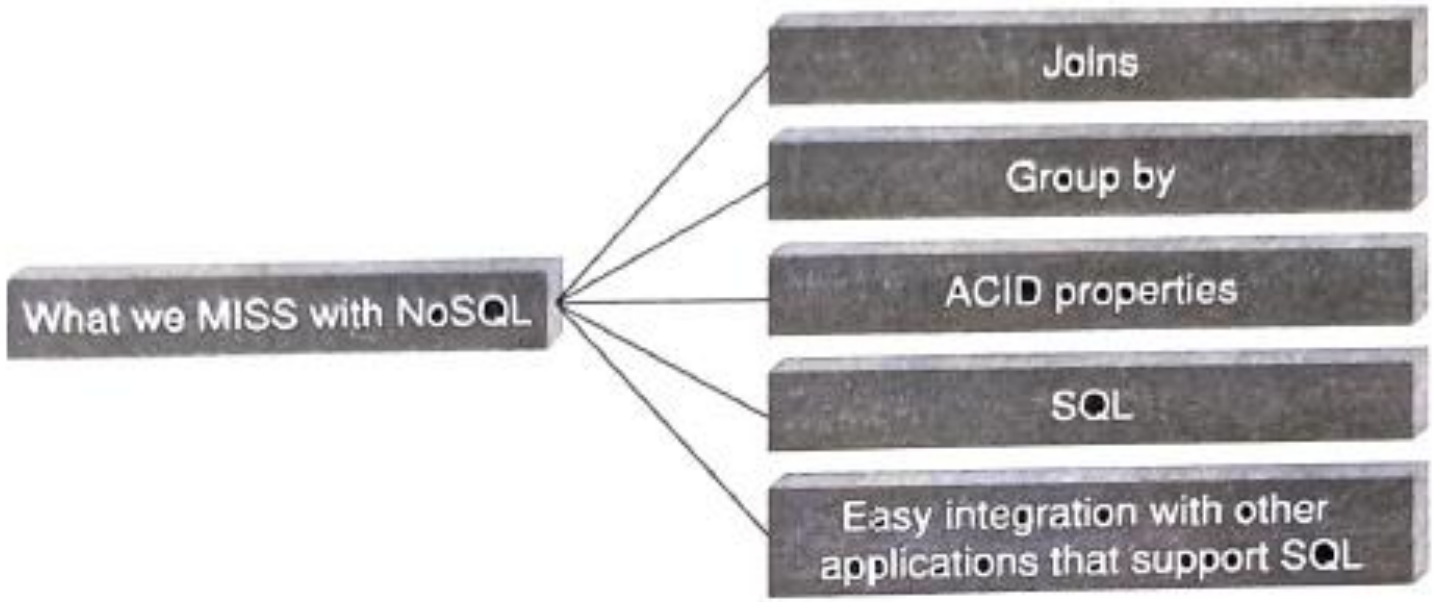


Figure 4.5 What we miss with NoSQL?

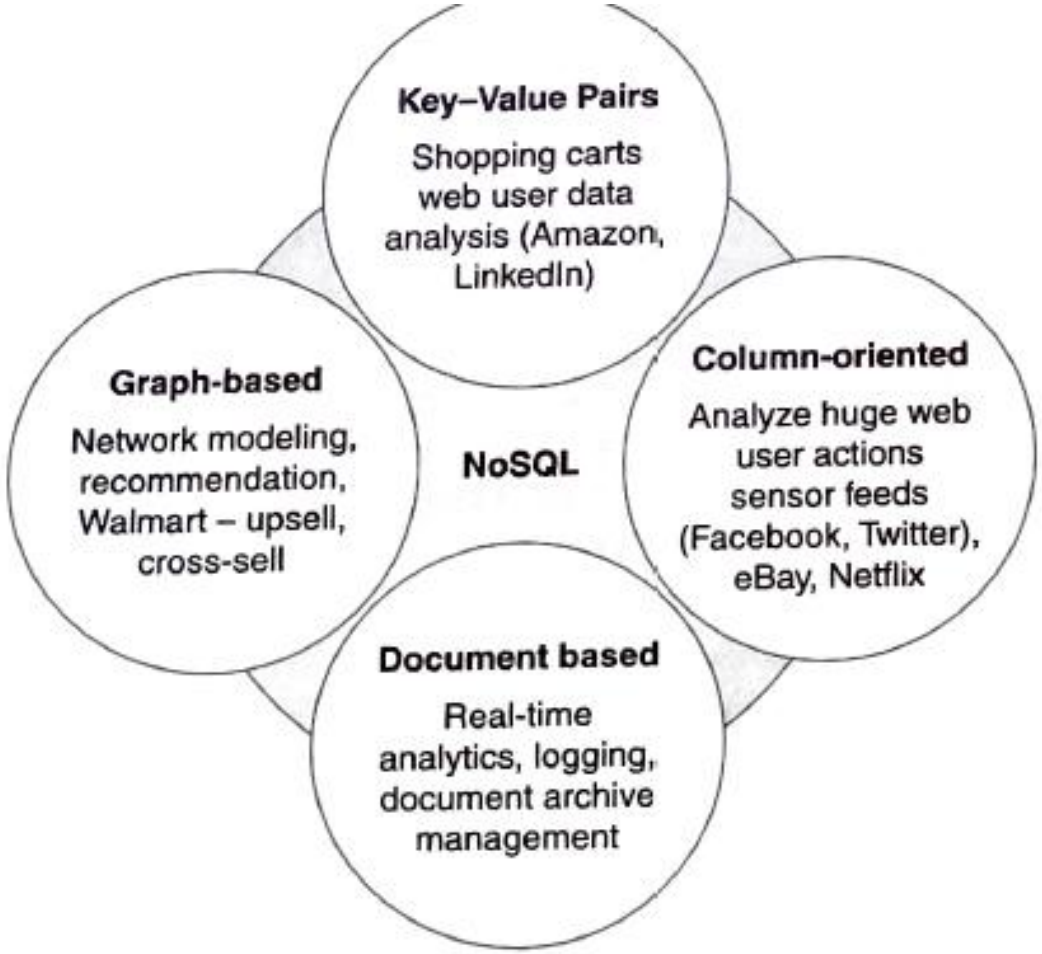


Figure 4.6 Use of NoSQL in industry.

- **4.1.9 SQL versus NoSQL**

Table 4.3 SQL versus NoSQL

SQL	NoSQL
Relational database	Non-relational, distributed database
Relational model	Model-less approach
Pre-defined schema	Dynamic schema for unstructured data
Table based databases	Document-based or graph-based or wide column store or key–value pairs databases
Vertically scalable (by increasing system resources)	Horizontally scalable (by creating a cluster of commodity machines)
Uses SQL	Uses UnQL (Unstructured Query Language)
Not preferred for large datasets	Largely preferred for large datasets
Not a best fit for hierarchical data	Best fit for hierarchical storage as it follows the key–value pair of storing data similar to JSON (Java Script Object Notation)
Emphasis on ACID properties	Follows Brewer's CAP theorem
Excellent support from vendors	Relies heavily on community support
Supports complex querying and data keeping needs	Does not have good support for complex querying
Can be configured for strong consistency	Few support strong consistency (e.g., MongoDB), some others can be configured for eventual consistency (e.g., Cassandra)
Examples: Oracle, DB2, MySQL, MS SQL, PostgreSQL, etc.	Examples: MongoDB, HBase, Cassandra, Redis, Neo4j, CouchDB, Couchbase, Riak, etc.

4.2 HADOOP

- Hadoop is an open-source project of the Apache foundation. It is a framework written in Java, originally developed by Doug Cutting in 2005 who named it after his son's toy elephant.
- Hadoop uses Google's MapReduce and Google File System technologies as its foundation. Hadoop is now a core part of the computing infrastructure for companies such as Yahoo, Facebook, LinkedIn, Twitter, etc. Refer Figure 4.8.

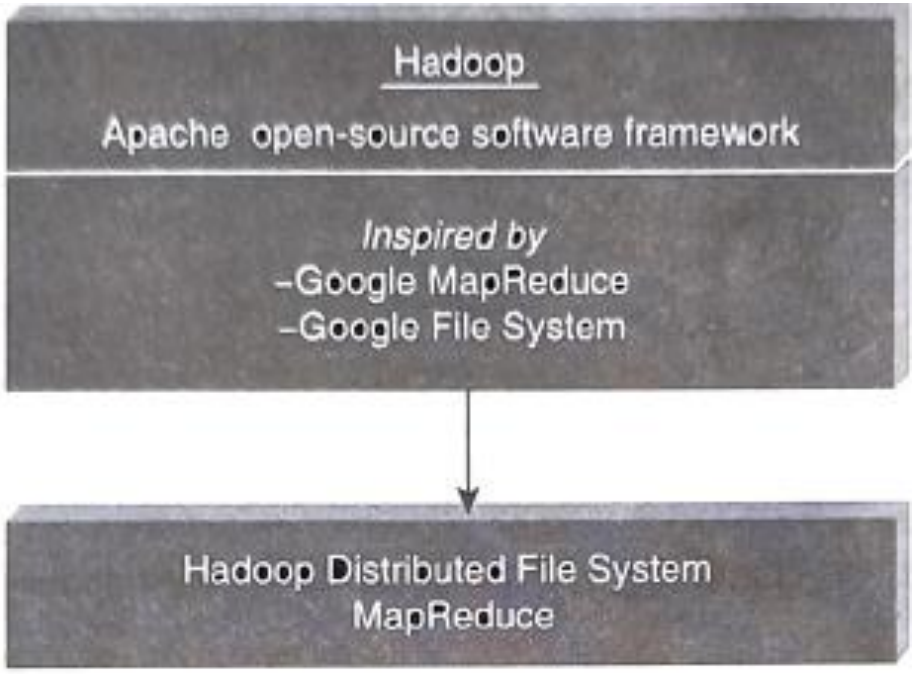


Figure 4.8 Hadoop.

4.2 HADOOP

- **4.2.1 Features of Hadoop**

1. It is optimized to handle massive quantities of structured, semi-structured, and unstructured data, using commodity hardware, that is, relatively inexpensive computers.
2. Hadoop has a shared nothing architecture.
3. It replicates its data across multiple computers so that if one goes down, the data can still be processed from another machine that stores its replica.
4. Hadoop is for high throughput rather than low latency. It is a batch operation handling massive quantities of data; therefore the response time is not immediate.
5. It complements On-Line Transaction Processing (OLTP) and On-Line Analytical Processing (OLAP). However, it is not a replacement for a relational database management system.
6. It is NOT good when work cannot be parallelized or when there are dependencies within the data.
7. It is NOT good for processing small files. It works best with huge data files and datasets.

4.2 HADOOP

- **4.2.2 Key Advantages of Hadoop**

1. Store data in its native format.
2. Scalable.
3. Cost-effective.
4. Resilient to failure.
5. Flexibility.
6. Fast.

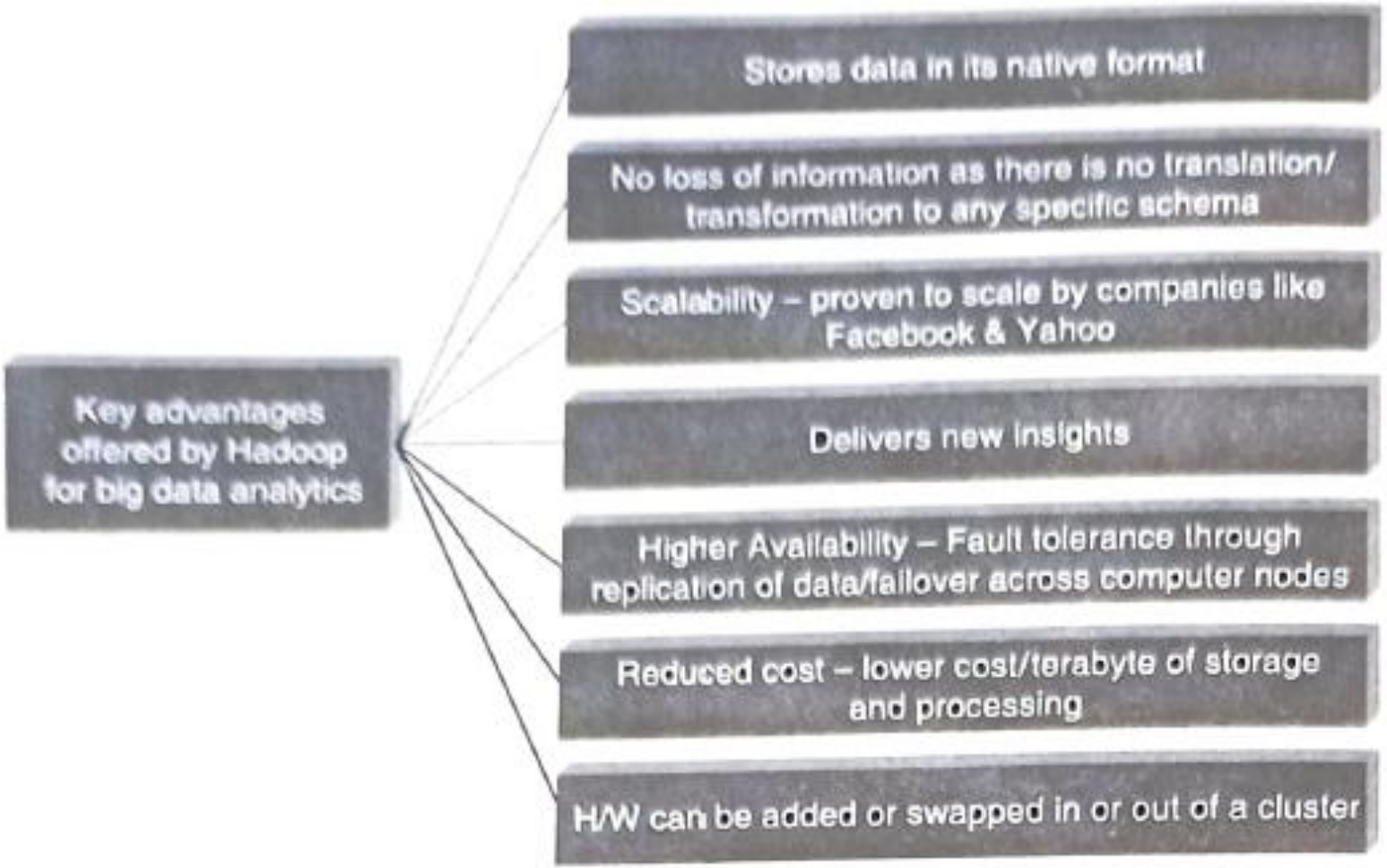
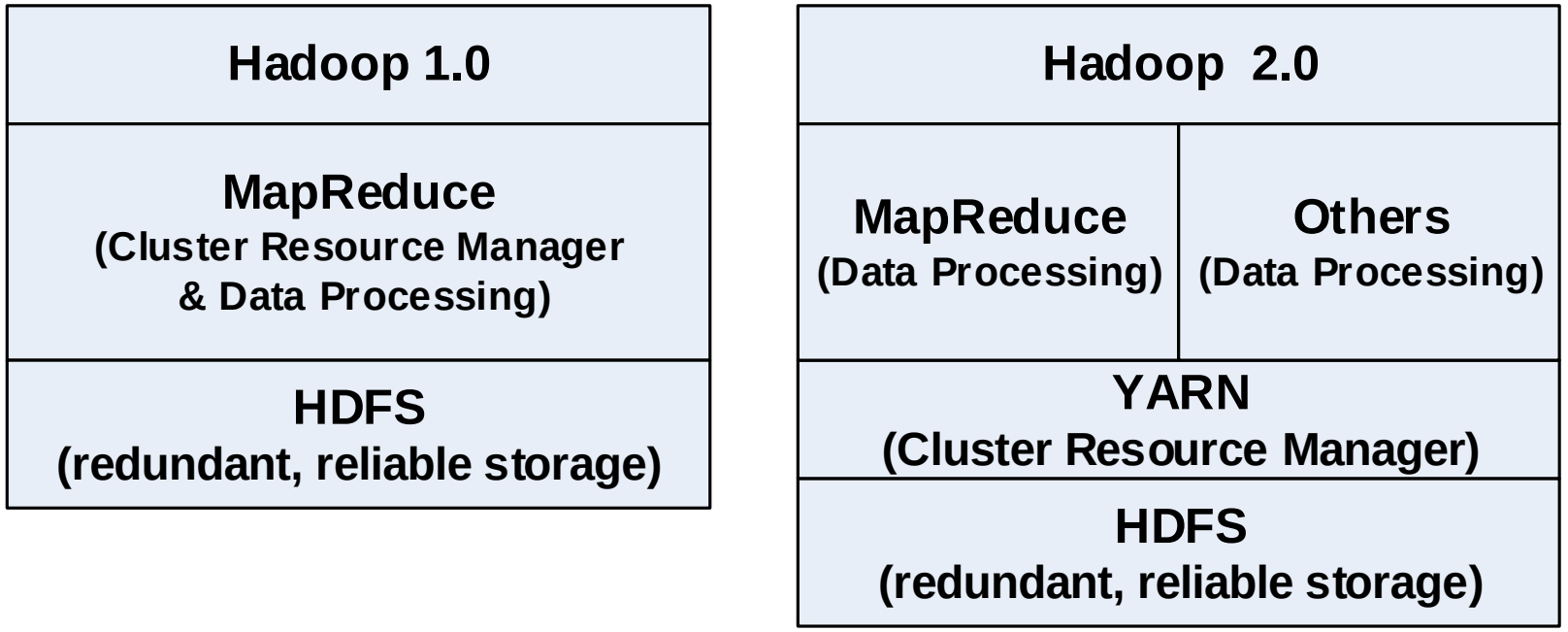


Figure 4.9 Key advantages of Hadoop.

- **4.2.3 Versions of Hadoop**



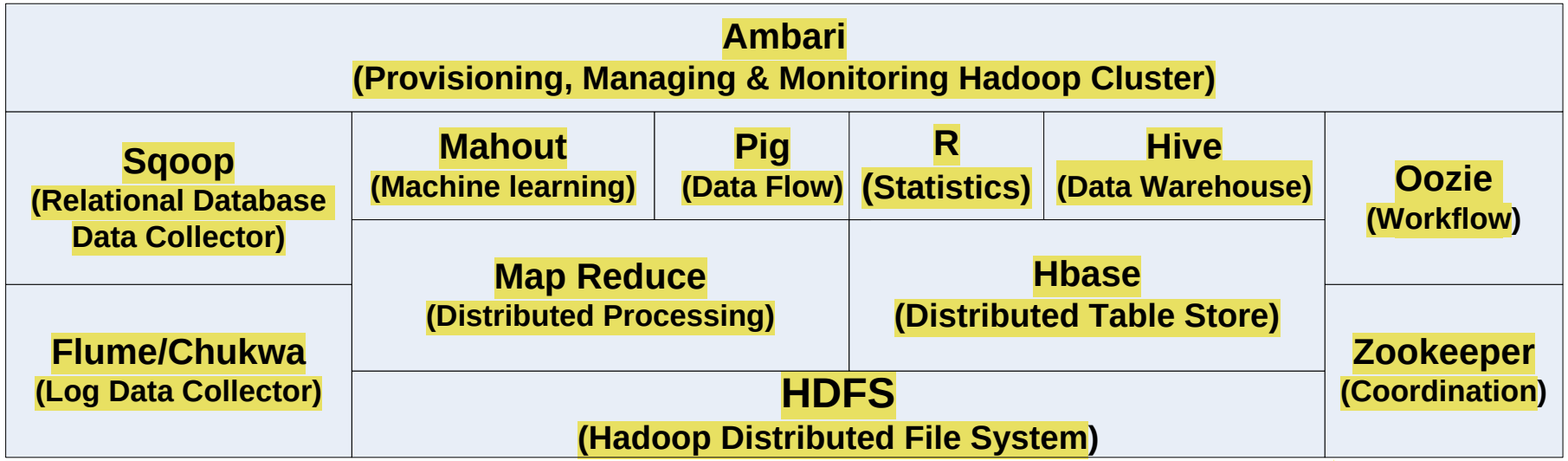


Fig.: Hadoop Eco System

4.2 HADOOP

- **4.2.4 Overview of Hadoop Eco System**
- Components that help with Data Ingestion are:
  - Sqoop.
  - Flume.
- Components that help with Data Processing are:
  - MapReduce.
  - Spark.
- Components that help with Data Analysis are:
  - Pig.
  - Hive.
  - Impala.
- **HDFS:** It is the distributed storage unit of Hadoop. It provides streaming access to file system data as well as file permissions and authentication. It is based on GFS (Google File System). It is used to scale a single cluster node to hundreds and thousands of nodes. It handles large datasets running on commodity hardware. HDFS is highly fault-tolerant. It stores files across multiple machines. These files are stored in redundant fashion to allow for data recovery in case of failure.

4.2 HADOOP

- **4.2.4 Overview of Hadoop Eco System**
- **HBase:** It stores data in HDFS. It is the first non-batch component of the Hadoop Ecosystem. It is a database on top of HDFS.
- It provides a quick random access to the stored data. It has very low latency compared to HDFS.
- It is a NoSQL database, is non-relational and is a column-oriented database. A table can have thousands of columns.
- A table can have multiple rows. Each row can have several column families. Each column family can have several columns. Each column can have several key values. It is based on Google BigTable.
- This is widely used by Facebook, Twitter, Yahoo, etc.

4.2 HADOOP

- **4.2.4 Overview of Hadoop Eco System**
- Difference between HBase and Hadoop/HDFS:

1. HDES is the file system whereas HBase is a Hadoop database.
2. HDFS is WORM (Write once and read multiple times or many times). Latest versions support appending of data but this feature is rarely used. However, HBase supports real-time random read and write.
3. HDFS is based on Google File System (GFS) whereas HBase is based on Google Big Table.
4. HDFS supports only full table scan or partition table scan. HBase supports random small range scan or table scan.
5. Performance of Hive on HDFS is relatively very good but for HBase it becomes 45 times slower.
6. The access to data is via MapReduce job only in HDFS whereas in HBase the access is via Java APIs, Rest, Avro, Thrift APIs.
7. HDFS does not support dynamic storage owing to its rigid structure whereas HBase supports dynamic storage.
8. HDFS has high latency operations whereas HBase has low latency operations.
9. HDFS is most suitable for batch analytics whereas HBase is for real-time analytics.

4.2 HADOOP

- **4.2.4 Overview of Hadoop Eco System**
- **Hadoop Ecosystem Components for Data Ingestion**:

1. Sqoop: Sqoop stands for SQL to Hadoop. Its main functions are:

   a) Importing data from RDBMS such as MySQL, Oracle, DB2, etc. to Hadoop file system (HDFS, HBase, Hive).

   b) Exporting data from Hadoop File system (HDES, HBase, Hive) to RDBMS (MySQL, Oracle, DB2).

- Uses of Sqoop:

a) It has a connector-based architecture to allow plug-ins to connect to external systems such as MySQL, Oracle, DB2, etc.

b) It can provision the data from external system on to HDFS and populate tables in Hive and HBase.

c) It allows to schedule and automate import and export tasks.

4.2 HADOOP

- **4.2.4 Overview of Hadoop Eco System**

2. Flume: Flume is an important log aggregator (aggregates logs from different machines and places them in HDES) component in the Hadoop ecosystem. Flume has been developed by Cloudera. It is designed for high volume ingestion of event-based data into Hadoop. The default destination in Flume (called as sink in flume parlance) is HDFS. However it can also write to HBase or Solr.

- **Hadoop Ecosystem Components for Data Processing**:

1. MapReduce: It is a programing paradigm that allows distributed and parallel processing of huge datasets. It is based on Google MapReduce.

- The MapReduce framework gets the input data from HDFS. There are two main phases: Map phase and the Reduce phase. The map phase converts the input data into another set of data (key-value pairs). This new intermediate dataset then serves as the input to the reduce phase. The reduce phase acts on the dataset to combine (aggregate and consolidate) and reduce them to a smaller set of tuples. The result is then stored back in HDFS.

4.2 HADOOP

- **4.2.4 Overview of Hadoop Ecosystem**

2. Spark: It is both a programming model as well as a computing model. It is written in Scala. It provides in-memory computing for Hadoop.

- In Spark, workloads execute in memory rather than on disk owing to which it is much faster (10 to 100 times) than when the workload is executed on disk. However, if the datasets are too large to fit into the available system memory, it can perform conventional disk-based processing.
- It serves as a potentially faster and more flexible alternative to MapReduce. It accesses data from HDFS (Spark docs not have its own distributed file system) but bypasses the MapReduce processing.
- Spark libraries: Spark SQL, Spark streaming, Mlib, and GraphX.

4.2 HADOOP

- **Hadoop Ecosystem Components for Data Analysis:**

1. Pig: It is a high-level scripting language used with Hadoop. It serves as an alternative to MapReduce. It has two parts:
   a) Pig Latin: It is SQL-like scripting language. Pig Latin scripts are translated into MapReduce jobs which can then run on YARN and process data in the HDFS cluster. It was initially developed by Yahoo. It is immensely popular with developers who are not comfortable with MapReduce. However, SQL developers may have a preference for Hive. How it works? There is a "Load" command available to load the data from "HDES" into Pig. Then one can perform functions such as grouping, filtering, sorting, joining etc. The processed or computed data can then be either displayed on screen or placed back into HDFS. It gives you a platform for building data flow for ETL (Extract, Transform and Load), processing and analyzing huge data sets.
   b) Pig Runtime.
2. Hive: Hive is a data warehouse software project built on top of Hadoop. Three main tasks performed by Hive are summarization, querying and analysis. It supports queries written in a language called HQL or HiveQL which is a declarative SQL-like language. It converts the SQL-style queries into MapReduce jobs which are then executed on the Hadoop platform.

Table 4.5 Hive versus RDBMS

	Hadoop	RDBMS
Data Variety	Used for structured, semi-structured and unstructured data. Hadoop supports a variety of data formats in real time such as XML, JSON, and text-based flat file formats.	Used for structured data
Data Storage	Usually datasets of size terabytes, petabytes	Usually datasets of size gigabytes
Querying	HiveQL	SQL
Query Response	In Hadoop, there is latency due to batch processing.	In RDBMS, query response time is immediate.
Schema	Schema required on read	Schema required on write
Speed	Writes are faster compared to reads as there is no adherence to schema required at the time of inserting or writing data. Schema is enforced at read time	Reads are very fast (supported by building indexes on required columns).
	Hadoop is designed for write once read many times. It does not work for random reading and writing of a few records like RDBMS.	RDBMS is designed for read and write many times.

Cost	Apache Hadoop is open-source, large-scale, distributed, scalable, data intensive computing.	Available as proprietary RDBMS such as Oracle, MS SQL Server, IBM DB2, etc. Also open-source RDBMS are available such as MySQL, PostgreSQL, etc.
Use Cases	Analytics, data discovery	OLTP (Online Transaction Processing). Mainly used to store and process day-to-day business data.
Throughput	High	Low
Scalability	Horizontal (Hadoop scales by adding nodes to a Hadoop cluster of easily available commodity machines).	Vertical: RDBMS scales vertically by increasing the horsepower (CPU, Hard Disk Capacity, RAM, etc.) of the machine.
Hardware	Commodity/Utility Hardware	High End Servers
Integrity	Low	High. Obeys ACID properties A – Atomicity C – Consistency I – Integrity D – Durability

4.2 HADOOP

- **4.2.5. Hadoop Distributions:** Hadoop is an open-source Apache project. Anyone can freely download the core aspects of Hadoop. The core aspects of Hadoop include the following:

  1. Hadoop Common. 2. HDFS. 3. YARN. 4.MapReduce.

4.2.6 Hadoop versus SQL

Table 4.6 lists the differences between Hadoop and SQL.

Table 4.6 Hadoop versus SQL

Hadoop	SQL
Scale out	Scale up
Key-Value pairs	Relational table
Functional Programming	Declarative Queries
Offline batch processing	Online transaction processing