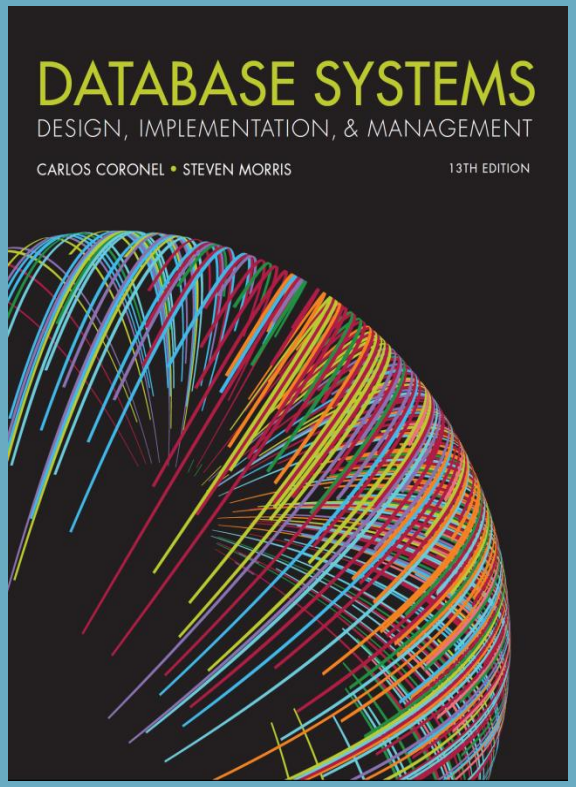


Chapter 8

NoSQL Database Management System

Learning Objectives

- Explain the role of Big Data in modern business
- Describe the primary characteristics of Big Data and how these go beyond the traditional “3 Vs”
- Explain how the core components of the Hadoop framework operate
- Identify the major components of the Hadoop ecosystem
- Summarize the four major approaches of the NoSQL data model and how they differ from the relational model
- Understand how to work with document databases using MongoDB

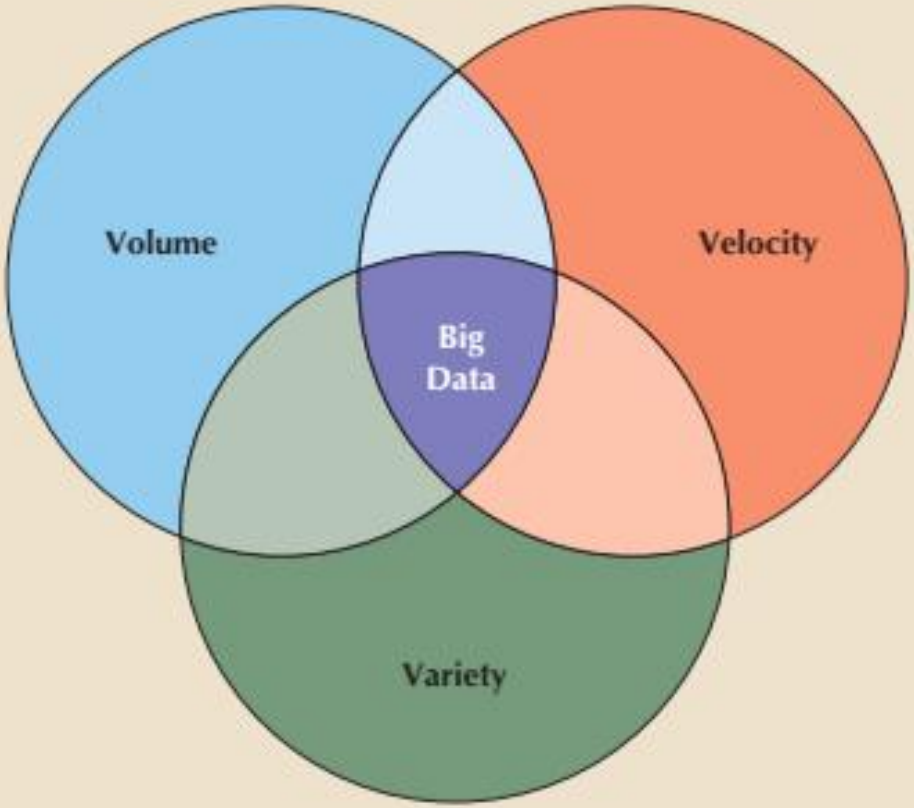
8.1 Big Data

- a set of data that displays the characteristics of **volume**, **velocity**, and **variety** (**3 Vs**) to an extent that makes the data unsuitable for management by a RDBMs.

Volume → the **quantity** 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

8.1 Big Data

FIGURE 14.1 ORIGINAL VIEW OF BIG DATA



8.1 Big Data

- Web data, a combination of text, graphics, video, and audio sources combined into complex structures created new challenges for data management that involve all three characteristics.
- After the dot-com bubble burst in the 1990s, many start-up web-based companies failed, but the companies that survived experienced significant growth as web commerce consolidated into a smaller set of businesses.
  - As a result, companies like Google and Amazon experienced significant growth and were among the first to feel the pressure of managing Big Data.
  - The success of social media giant Facebook became pioneers in creating new technologies to address Big Data problems.
  - Google created the BigTable data store, Amazon created Dynamo, and Facebook created Cassandra, technologies to deal with the growing need to store and manage large sets of data that had the characteristics of the 3 Vs.

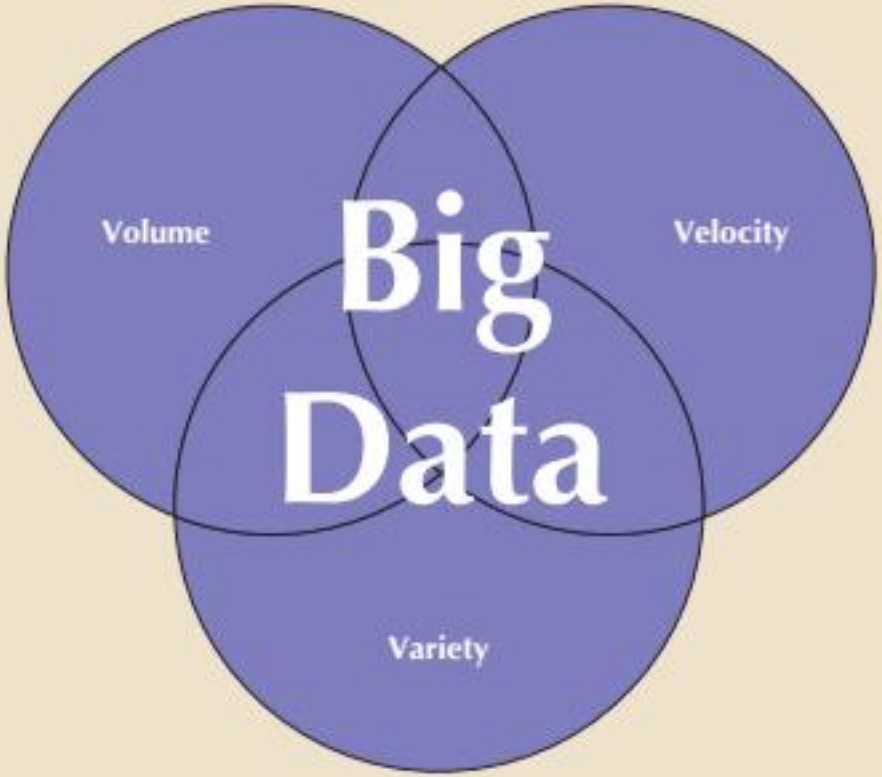
8.1 Big Data

- Changes in technology have increased the opportunities for businesses to generate and track data so that Big Data has been redefined as involving any, but not necessarily all, of the 3 Vs, as shown in Figure 14.2.

- Advances in technology have led to a vast array of user-generated data and machine-generated data that can spur growth in specific areas. I.e. Magic Band by Disneyland
  - Radio Frequency Identification (RFID)
  - Near-field Communication (NFC)

8.1 Big Data

FIGURE 14.2 CURRENT VIEW OF BIG DATA



8.1 Big Data: Volume

- The quantity of data to be stored

STORAGE CAPACITY UNITS		
TERM	**CAPACITY**	**ABBREVIATION**
Bit	0 or 1 value	b
Byte	8 bits	B
Kilobyte	1024* bytes	KB
Megabyte	1024 KB	MB
Gigabyte	1024 MB	GB
Terabyte	1024 GB	TB
Petabyte	1024 TB	PB
Exabyte	1024 PB	EB
Zettabyte	1024 EB	ZB
Yottabyte	1024 ZB	YB

8.1 Big Data: Volume

- As the quantity of data needing to be stored increases, the need for larger storage devices increases as well. When this occurs, systems can either scale up or scale out.

- **Scaling up** --> keeping the same number of systems, but migrating each system to a larger system

- **Scaling out** --> when the workload exceeds the capacity of a server, the workload is spread out across a number of servers.
  - This is also referred to as ***clustering***—creating a cluster of low-cost servers to share a workload.
  - To reduce the overall cost of the computing resources since it is cheaper to buy ten 100 TB storage systems than it is to buy a single 1 PB storage system.

8.1 Big Data: Volume

There are significant limits associated with the ability to distribute the DBMS due to the increased performance costs of communication and coordination as the number of nodes grows. This limits the degree to which a relational database to be scaled out as data volume grows, and it makes RDBMSs ill-suited for clusters.

8.1 Big Data: Velocity

- The rate at which new data enters the system as well as the rate at which the data must be processed.

I.e. In the past, a retail store might capture only the data about the final transaction of a customer making a purchase. Today, a retailer like Amazon captures not only the final transaction but also every click of the mouse in the ***searching, browsing, comparing***, and ***purchase*** process.

- Instead of capturing one event (the final sale) in a 20-minute shopping experience, it might capture data on 30 events during that 20-minute time frame—a 30× increase in the velocity of the data.

8.1 Big Data: Velocity

- Data must be processed at a very rapid pace.
- Two categories of velocity of processing:

1) Stream processing

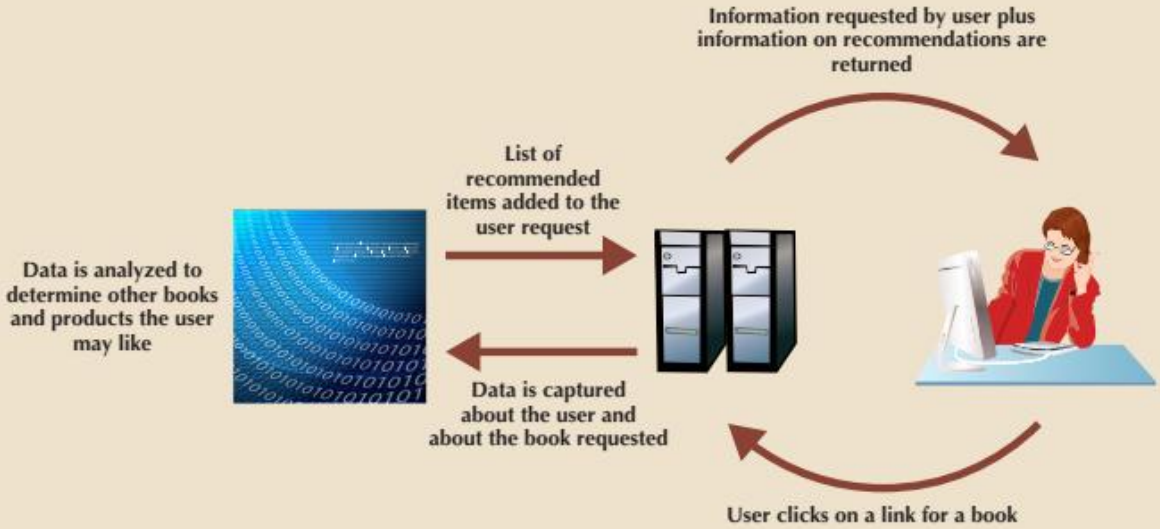
- Focuses on input processing
- Requires analysis of the data stream as it enters the system.
- The data must be processed and filtered as it enters the system to determine which data to keep and which data to discard.
- Scientists use algorithms to decide which data to keep

8.1 Big Data: Velocity

1) Feedback loop processing

- The analysis of the data to produce actionable results.
- Focusing on the outputs.
- The process of capturing the data, processing it into usable information, and then acting on that information.

FIGURE 14.3 FEEDBACK LOOP PROCESSING



8.1 Big Data: Variety

- The vast array of formats and structures in which the data may be captured.

- Data can be considered to be:

  - **Structured** - organized to fit a predefined data model
  - **Unstructured** - not organized to fit into a predefined data model
  - **Semistructured** - combines elements of both

8.1 Big Data: Variety

- Big Data requires that the data be captured in whatever format it naturally exists, without any attempt to impose a data model or structure to the data.
  - One of the key differences between processing data in a relational database and Big Data processing.
  - Relational databases impose a structure on the data when the data is captured and stored.
  - Big Data processing imposes a structure on the data as needed for applications as a part of retrieval and processing.

8.1 Big Data: Other Characteristics

As the industry matures, other characteristics have been put forward as being equally important.

TABLE 14.2

ADDITIONAL Vs OF BIG DATA

CHARACTERISTIC	DESCRIPTION
Variability	Data meaning changes based on context.
Veracity	Data is correct.
Value (Viability)	Data can provide meaningful information.
Visualization	Data can be presented in such a way as to make it understandable.

Variability is especially relevant in areas such as *sentiment analysis* that attempt to understand the meanings of words.

8.2 Hadoop

Big Data requires a different approach to distributed data storage that is designed for large-scale clusters.

- Hadoop the de facto standard for most Big Data storage and processing.
- A **Java-based framework** for distributing and processing very large data sets across clusters of computers.
- The Hadoop system uses **batch processing**.
- 2 important parts:-
  - **Hadoop Distributed File System (HDFS)**
  - **MapReduce**

8.2 Hadoop: HDFS

Key assumptions:

- **High Volume**
  - Terabytes, petabytes or larger
  - Hadoop assumes files in HDFS will be extremely large
  - Data in the HDFS is organized into physical blocks
- **Write-once, read-many**
  - Simplifies concurrency issues and improves overall data throughput.
  - A file is created, written to the file system, and then closed. Changes cannot be made to its contents.
- **Streaming access**
  - Hadoop is optimized for batch processing of entire files as a continuous stream of data.
- **Fault tolerance**
  - HDFS is designed to replicate data across many different devices so that when one device fails, the data is still available from another device.

8.2 Hadoop: HDFS

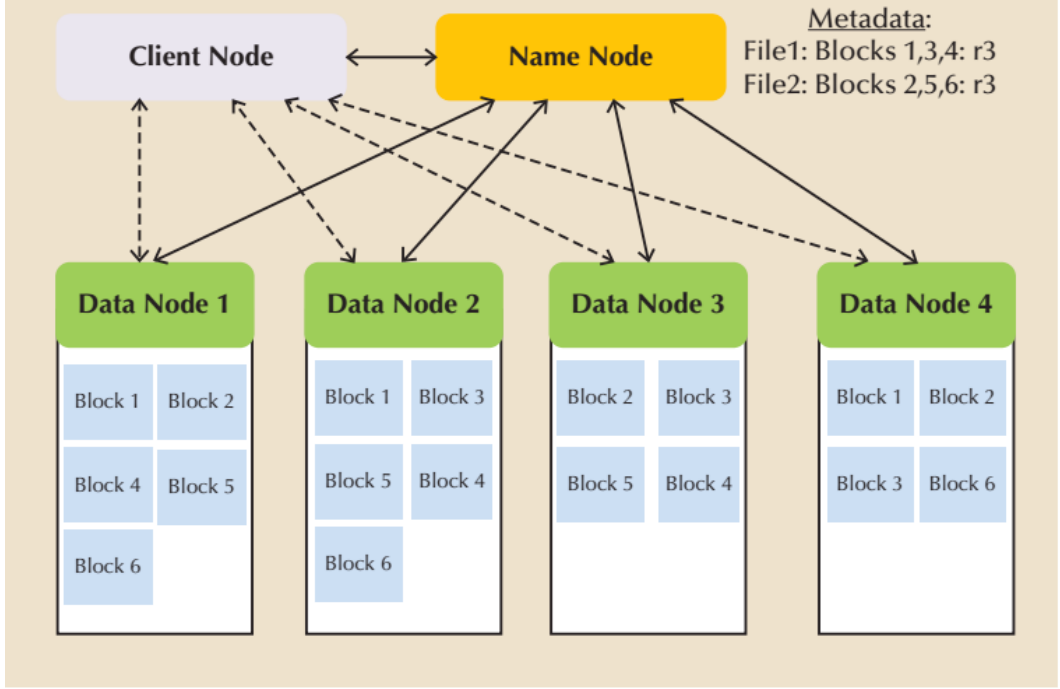
- Hadoop uses several types of nodes

When a **client node** needs to create a new file, it communicates with the **name node**.

The **name node**:

- Adds the new file name to the metadata.
- Determines a new block number for the file.
- Determines a list of which **data nodes** the block will be stored.
- Passes that information back to the **client node**.

FIGURE 14.4 HADOOP DISTRIBUTED FILE SYSTEM (HDFS)



8.2 Hadoop: HDFS

- The **client node** contacts the first **data node** specified by the **name node** and begins writing the file on that **data node**.
- At the same time, the **client node** sends the **data node** the list of other **data nodes** that will be replicating the block.
- As the data is received from the **client node**, the **data node** contacts the next **data node** in the list and begins sending the data to this node for replication.
- This second **data node** then contacts the next **data node** in the list and the process continues with the data being streamed across all of the **data nodes** that are storing the block.
- Once the first block is written, the **client node** can get another block number and list of **data nodes** from the **name node** for the next block.
- When the entire file has been written, the informs the **name node** that the file is closed. **client node**

8.2 Hadoop: HDFS

The data nodes send **block reports** and **heartbeats.**

- A **block report** is sent every **6 hours** and informs the **name node** of which blocks are on that **data node**.

- **Heartbeats** are sent every **3 seconds**.
  - To let the **name node** know that the **data node** is still available. If a **data node** experiences a fault, due to hardware failure, power outage, and so on, then the **name node** will not receive a heartbeat from that **data node**.
  - As a result, the **name node** knows not to include that **data node** in lists to **client nodes** for reading or writing files.
  - If the lack of a heartbeat from a **data node** causes a block to have fewer than the desired number of replicas, the **name node** can have a "**live**" **data node** initiate replicating the block on another **data node**.

8.2 Hadoop: MapReduce

- Computing framework used to process large data sets across clusters.
- ***Divide*** and ***conquer*** principles
- Breaks down a complex task into a collection of smaller subtasks, performs the subtasks all at the same time, and then combines the result of each subtask to produce a final result for the original task.

Map

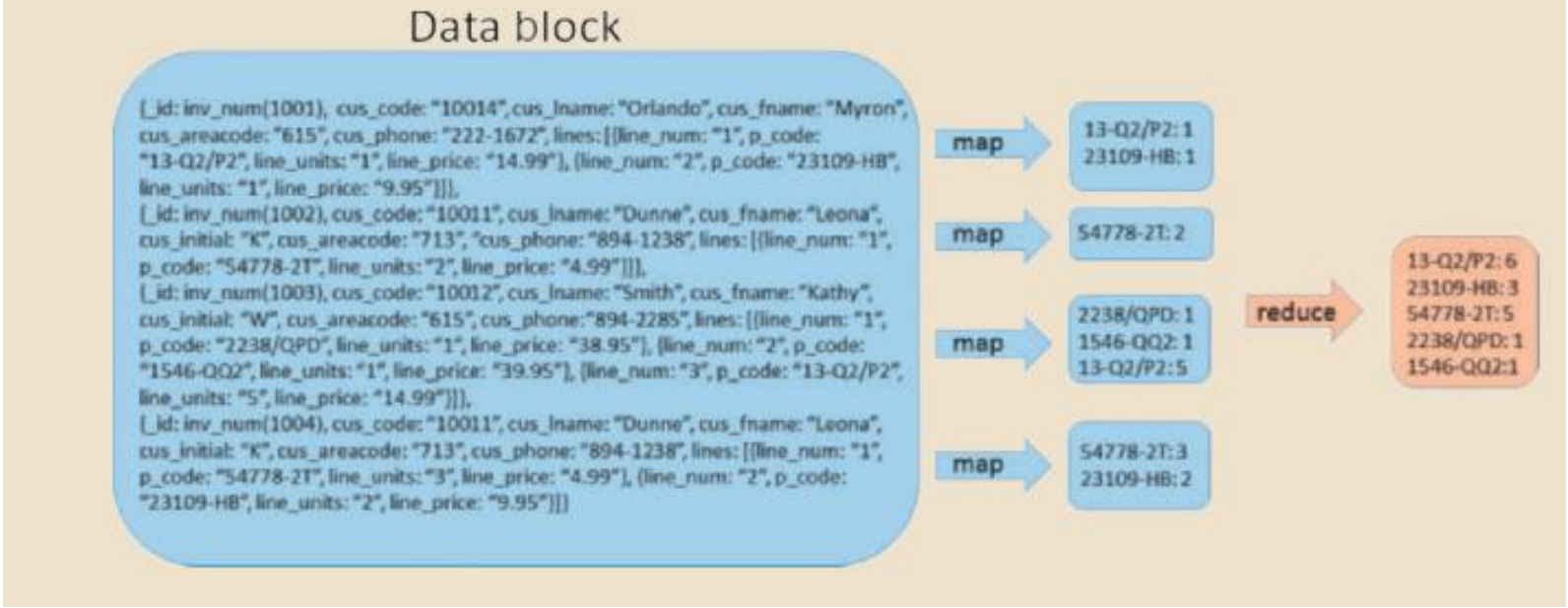
- Takes a collection of data and sorts and filters the data into a set of key-value pairs.
- Performed by a mapper.

Reduce

- Takes a collection of key-value pairs, all with the same key value, and summarizes them into a single result.
- Performed by a program called a reducer.

8.2 Hadoop: MapReduce

FIGURE 14.5 MAPREDUCE



- Map functions parse each invoice to find data
- The result is a new list of key-value pairs; product code (key) and the line units (value).
- The reduce function then takes that list of key-value pairs and combines them by summing the values associated with each key (product code) to produce the summary result.

8.2 Hadoop: MapReduce

- The Hadoop framework distributes a mapper for each block on each **data node** that must be processed.

  → Lead to a very large number of mappers. I.e. If 1 TB of data is to be processed and the HDFS is using 64 MB blocks, that yields over 15,000 mapper programs.

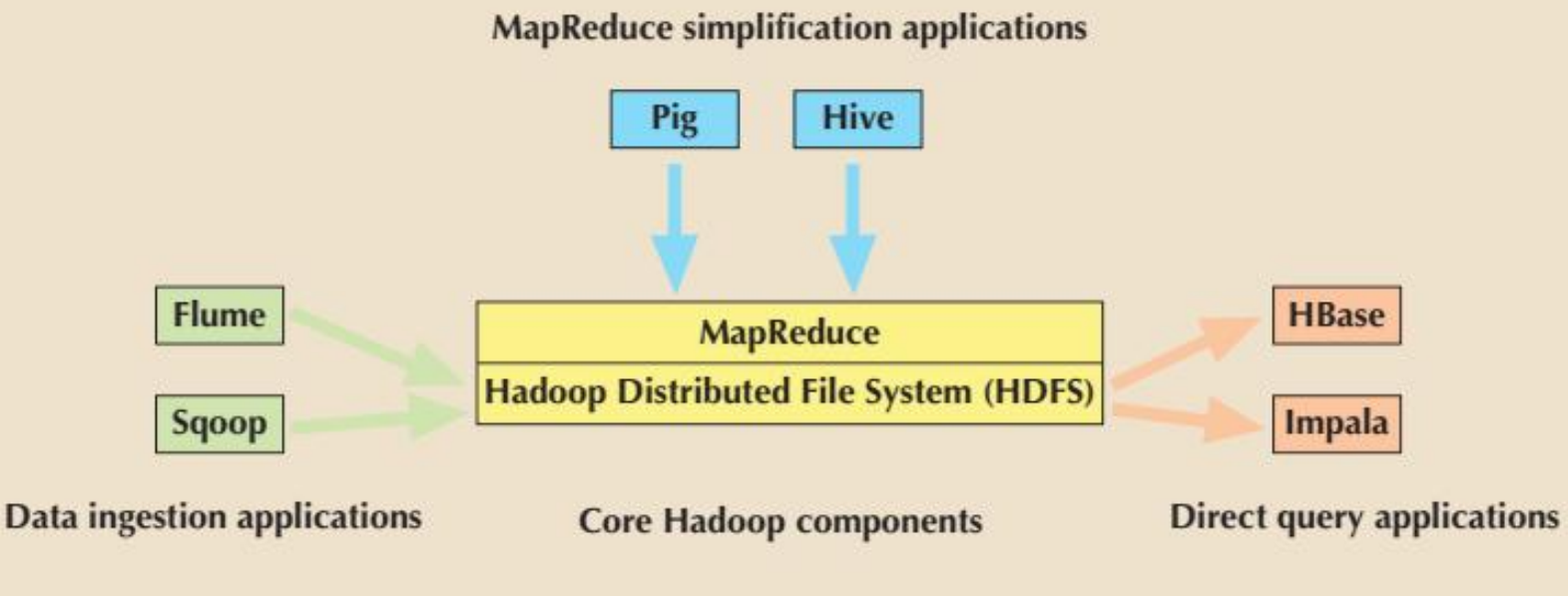
  → The number of reducers is configurable by the user, but best practices suggest about one reducer per **data node**.

8.2 Hadoop: MapReduce

- The implementation of MapReduce complements the structure of the HDFS. MapReduce uses:
  - **JobTracker** – A central control for MapReduce processing, and it normally exists on the same server that is acting as the **name node**.
    - Locate data
    - Determine which nodes to use
    - Divide job into tasks for the nodes
    - Manage failed nodes
  - **TaskTrackers** – reside on the **data nodes**.
    - Handle a set no of tasks
    - Creates JVM to run the map and reduce functions
    - Send heartbeat messages to the job tracker

8.3 Hadoop Ecosystem

FIGURE 14.6 A SAMPLE OF THE HADOOP ECOSYSTEM

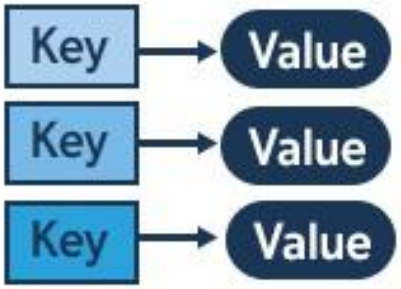


8.4 NoSQL

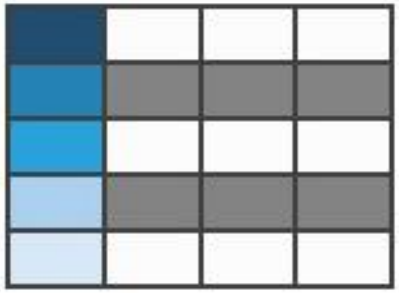
- Not only SQL
- Non-relational databases that store and retrieve data in flexible, non-tabular formats
- Dynamic schema to handle large volumes of unstructured and semi-structured data
- Horizontal scalability
- Distributed and High availability
- Performance

NoSQL

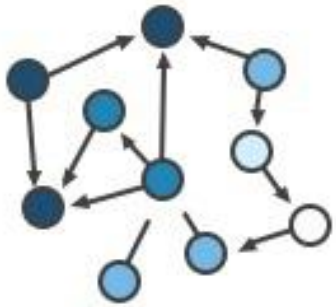
Key-Value



Column-Family



Graph



Document

8.4 NoSQL Databases

TABLE 14.3

NoSQL DATABASES

NoSQL CATEGORY	EXAMPLE DATABASES	DEVELOPER
Key-value database	Dynamo Riak Redis Voldemort	Amazon Basho Redis Labs LinkedIn
Document databases	MongoDB CouchDB OrientDB RavenDB	MongoDB, Inc. Apache OrientDB Ltd. Hibernating Rhinos
Column-oriented databases	HBase Cassandra Hypertable	Apache Apache (originally Facebook) Hypertable, Inc.
Graph databases	Neo4J ArangoDB GraphBase	Neo4j ArangoDB, LLC FactNexus

8.4 Types of NoSQL Databases

Key-Value Stores	Document-Based	Column-Oriented	Graph-Based
Every data element in the database is stored in key-value pairs.	Uses documents to store data in the database	Stores the data in columns instead of rows	focus on the relationship between the elements.
The data can be retrieved by using a unique key allotted to each element in the database	Stores data in JSON, BSON, or XML documents.	To run analytics on a small number of columns, you can read those columns directly without consuming memory with the unwanted data.	It stores the data in the form of nodes in the database
The values can be simple data types like strings and numbers or complex objects.	The particular elements can be accessed by using the index value that is assigned for faster querying.	Columnar databases are designed to read data more efficiently and retrieve the data with greater speed.	The connections between the nodes are called links or relationships.
Key Features: • Simplicity • Scalability • Speed	**Key Features:** • Flexible schema • Faster creation and maintenance • No foreign keys • Open formats	**Key Features:** • Scalability • Compression • Very responsive	**Key Features:** • Easy to identify the relationship between the data by using the links • The Query's output is real-time results • The speed depends upon the number of relationships among the database elements • Easy data update

8.4 NoSQL Databases: Key-Value

FIGURE 14.7 KEY-VALUE DATABASE STORAGE

Bucket = Customer

Key	Value
10010	"LName Ramas FName Alfred Initial A Areacode 615 Phone 844-2573 Balance 0"
10011	"LName Dunne FName Leona Initial K Areacode 713 Phone 894-1238 Balance 0"
10014	"LName Orlando FName Myron Areacode 615 Phone 222-1672 Balance 0"

- Key-value pairs are typically organized into "**buckets**."
- A bucket is a logical grouping of keys.
- Key values must be unique within a bucket, but they can be duplicated across buckets.
- KV DB Operations: *get/fetch, store* & *delete*

8.4 NoSQL Databases: Document

FIGURE 14.8 DOCUMENT DATABASE TAGGED FORMAT

Collection = Customer

Key	Document
10010	{LName: "Ramas", FName: "Alfred", Initial: "A", Areacode: "615", Phone: "844-2573", Balance: "0"}
10011	{LName: "Dunne", FName: "Leona", Initial: "K", Areacode: "713", Phone: "894-1238", Balance: "0"}
10014	{LName: "Orlando", FName: "Myron", Areacode: "615", Phone: "222-1672", Balance: "0"}

- Group documents into logical groups called "**collections**".
- Always stores a document in the value component.
- The document can be in any encoded format, such as XML, JSON or BSON
- Attempt to understand the content of the value component.

8.4 NoSQL Document database: MongoDB

- Product of **MongoDB Inc.** (open source)
- One of the most successful in database market. Design for
  - High availability
  - High scalability
  - High performance
- Schema-less and aggregate aware
- MongoDB databases are comprised of collections of documents (JSON files).
- Use ***MongoDB Query Language*** (JavaScript-like Language)

8.4 NoSQL Document database: MongoDB

- Each MongoDB server can host many databases.
- When connected to the MongoDB server, the first task is to specify with which database object you want to work.
- Objects are enclosed in curly brackets {} that contain key-value pairs.
- A single JSON object can contain many key:value pairs separated by commas.

8.4 NoSQL Document database: MongoDB

- I.e. A simple JSON to store data on a book

```
{_id: 101, title: 'Database Systems'}
```

- The value component may have multiple values that would be appropriate for a given key.
- When there are multiple values for a single key, an array is used.
- Arrays in JSON are placed inside square brackets [].

```
{_id: 101, title: 'Database Systems', author: ['Coronel', 'Morris']}
```

- To retrieve a list of available databases on server: **`Show dbs`**
- To create a new DB in MongoDB: **`Use fact`**
- Method to retrieve objects from a collection that match restrictions provided: **`find()`**

8.4NoSQL Databases: Column-oriented /Column family

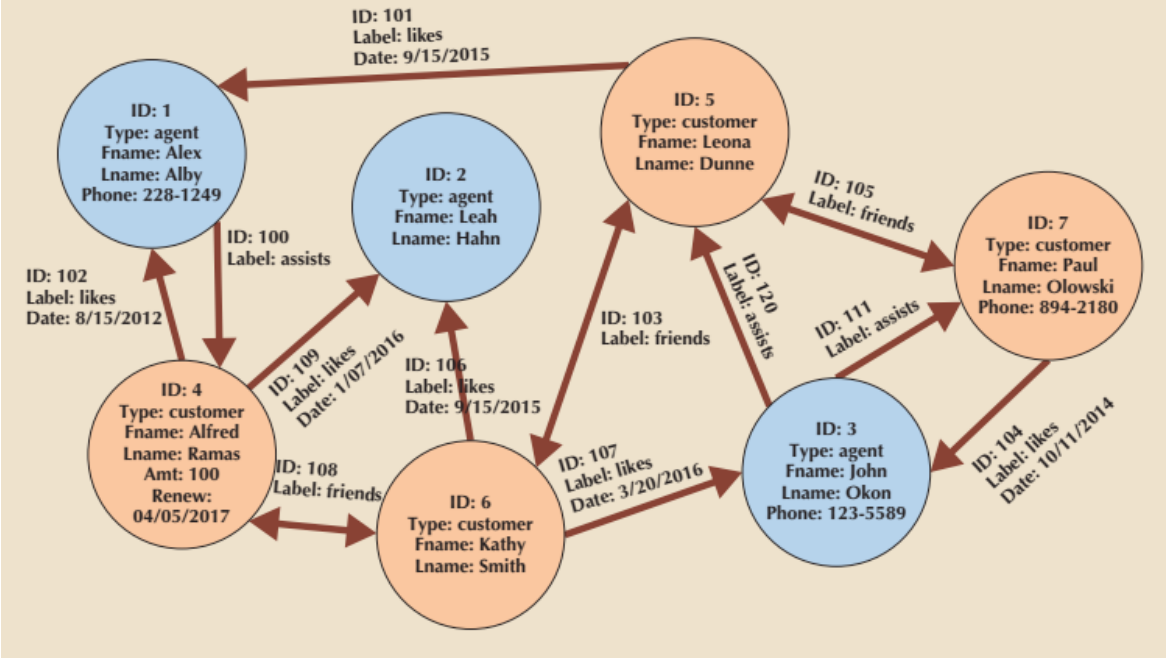
FIGURE 14.10 COLUMN FAMILY DATABASE

Column Family Name	CUSTOMERS	
Key	Rowkey 1	
Columns	City	Nashville
	Fname	Alfred
	Lname	Ramas
	State	TN
Key	Rowkey 2	
Columns	Balance	345.86
	Fname	Kathy
	Lname	Smith
Key	Rowkey 3	
Columns	Company	Local Markets, Inc.
	Lname	Dunne

- Organizes data in key value pairs with keys mapped to a set of columns in the value component.
- The key is the name of the column, and the value component is the data that is stored in that column.
- Some columns form natural groups (logically related) - **super column**
- A column family can be composed of columns or super columns, but it cannot contain both.

8.4 NoSQL Databases: Graph

FIGURE 14.11 GRAPH DATABASE REPRESENTATION



- A NoSQL database based on graph theory to store data about relationship-rich environments.
- Graph theory is a mathematical and computer science field that models relationships, or edges, between objects called nodes.
- A query in a graph database is called a **traversal**.

8.4 NoSQL Database Use Cases

Key-Value Stores	• **E-commerce** platforms where large amounts of customer-related data like orders, user profiles, and product catalogues are generated daily. • Due to its low latency and quick processing, it is suitable for real-time inventory management and handling high traffic.
Document-based	• **Content Management Systems**. Example: a blogging website. • Data in the form of articles, comments, categories, and tags can be stored and retrieved quickly. • Suitable for storing unstructured data like texts, images, links, etc.
Column-oriented	• **Data warehousing applications**. • These applications require analysing large amounts of data for business intelligence with a high write throughput.
Graph-based	• Creating **recommendation engines**. Example: Youtube, which recommends videos to users based on their viewing history. • Graph databases can store and process interconnected data and quickly deliver relevant content.

8.4 SQL vs NoSQL Technologies: Main Features

Main Features	SQL	NoSQL
Data Storage	• Tables: Rows & Columns • Constraints: e.g. datatype • Relations: Foreign keys	• Documents (JSON) • Column oriented • Graphs • Key Value
Data Structure & Model	Works better with structured data	Make sense for less structured data (unstructured & semi structured)
Scale	• Vertical (scale-up with a larger server)	Horizontal (scale-out across commodity servers)
Access	• Raw SQL • Direct database connection • Object Relational Mappers	• REST APIs • CRUD (Vendor specific language)

8.4 NoSQL Vs SQL

Feature	Relational DB (SQL)	NoSQL
Data Model	Structured, Tabular	Flexible (Documents, Key-value, Graph)
Scalability	Vertical Scaling	Horizontal Scaling
Schema	Predefined	Dynamic and Schema-less
ACID Support	Strong	Limited or Eventual Consistency
Best For	Transactional Applications	Big Data, real-time analytics
Examples	MySQL, PostgreSQL, ORACLE	MongoDB, Cassandra, Redis

8.4 NoSQL Applications

Big Data Applications: Efficiently stores and processes massive amounts of unstructured and semi-structured data.

Real-Time Analytics: Supports fast queries and analysis for use cases like recommendation engines or fraud detection.

Scalable Web Applications: Handles high traffic and large user bases by scaling horizontally across servers.

Flexible Data Storage: Manages diverse data formats (JSON, key-value, documents, graphs) without rigid schemas.

8.4 Popular NoSQL Databases

MongoDB:

- A document-oriented database that stores JSON-like documents in dynamic schemas.

Apache CouchDB:

- CouchDB uses the JSON data exchange format to store its documents; JavaScript for indexing, combining, and transforming documents; and HTTP for its API.

Apache HBase:

- HBase is a column store database written in Java with capabilities similar to those that Google BigTable® provides.

Oracle NoSQL Database:

- A proprietary database that supports JSON table and key-value datatypes running on-premise or as a cloud service.

Apache Cassandra DB:

- A distributed database that excels at handling extremely large amounts of structured data. Cassandra DB is also highly scalable. Facebook® created Cassandra DB.

Riak:

- An open-source, key-value store database written in Erlang. Riak has built-in fault-tolerance replication and automatic data distribution that enable it to offer excellent performance.

Objectivity InfiniteGraph:

- A highly specialized graph database that focuses on graph data structures. InfiniteGraph, implemented in Java, is useful for finding hidden relationships in big data.

8.4 NoSQL: Challenges

- Lack of standardization
- Lack of ACID compliance
- Narrow focus
- Absence of complex query support
- Lack of maturity
- Management complexity
- Limited GUI tools

Summary

- Big Data is characterized by data of such volume, velocity, and/or variety that the relational model struggles to adapt to it.
- The Hadoop framework has quickly emerged as a standard for the physical storage of Big Data.
- The primary components of the Hadoop framework include the Hadoop Distributed File System (HDFS) and MapReduce.
- The Hadoop framework also supports an entire ecosystem of additional tools and technologies, such as Hive, Pig, and Flume, that work together to produce a complex system of Big Data processing.
- NoSQL is a broad term to refer to any of several nonrelational database approaches to data management.
- Most NoSQL databases fall into one of four categories: key-value databases, document databases, column-oriented databases, and graph databases.

Summary

- Key-value databases store data in key-value pairs, where the value of the key must be known to the DBMS, but the data in the value component can be of any type.
- Document databases also store data in key-value pairs, but the data in the value component is an encoded document. The document must be encoded using tags, such as in XML or JSON.
- Column-oriented databases organize data into key-value pairs in which the value component is composed of a series of columns, which are themselves key-value pairs.
- Graph databases are based on graph theory and represent data through nodes, edges, and properties.
- MongoDB is a document database that stores documents in JSON format. The documents can be created, updated, deleted, and queried using a JavaScript-like language (MongoDB Query Language). Data retrieval is done primarily through the find() method