upGrad

NoSQL

Databases and Apache HBase

Course : Data Engineering – I

Lecture On : NoSQL Databases & Apache HBase

Session 1

Introduction to NoSQL Databases and Apache HBase

Segment 1

Module Introduction

MODULE INTRODUCTION

Session 1

- Drawbacks of RDBMS
- What is NoSQL Database?
- CAP Theorem
- How is NoSQL Database Designed?
- NoSQL Use Cases
- Inception of HBase
- HBase Data Model
- HBase Shell Commands

Session 2

- Programming with HBase
- HBase Python API: HappyBase
  - Manipulating HBase Tables
  - Creating, Updating, Deleting HBase Tables
  - Integration with MapReduce.

Session 3

- HBase Architecture
- Read/Write Operations on HBase
- HBase schema design
- Use Cases, Advantages and Disadvantages

Segment 3

Why NoSQL Databases?

LEARNING OBJECTIVES

Why industries are shifting from relational databases to distributed databases?

What are NoSQL Databases?

How SQL and NoSQL Databases are different from each other?

RDBMS

1 Data stored in form of tables

2 Schema- oriented

3 Run queries across multiple tables at once

4 No flexibility in data model

5 Processes and stores only structured data

STRUCTURED DATA

01 **Fixed record lengths**

02 **Defined data-types**

03 **Easily searchable**

04 **Customer data like name, address, bank details, etc.**

UNSTRUCTURED DATA

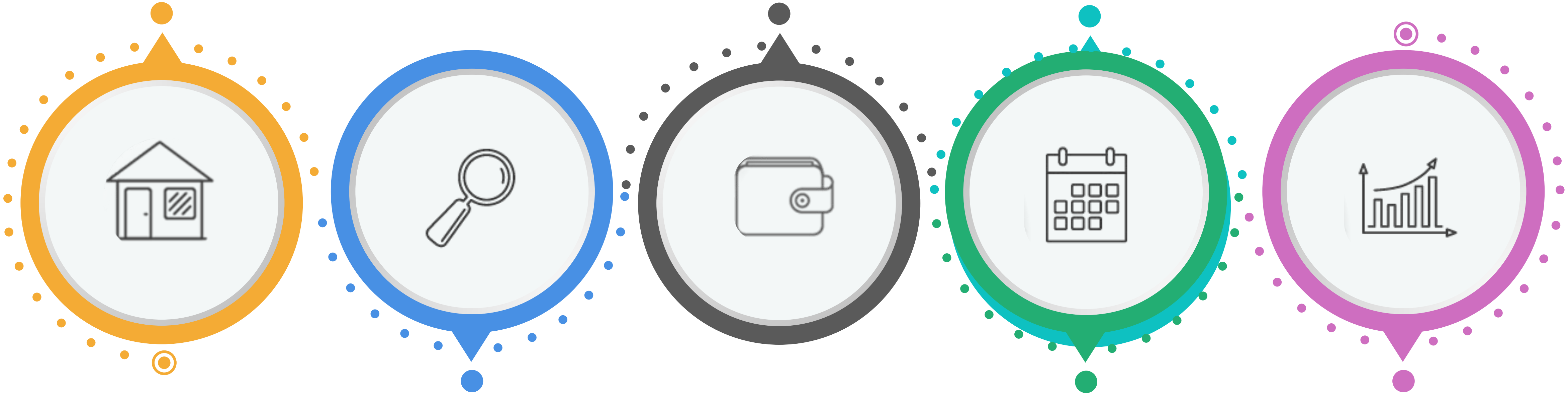
01 Any form or shape

02 No predefined schema

03 Human-generated: text files, audio files, etc.

04 Machine-generated: sensor data, satellite imagery

CHALLENGES WITH TRADITIONAL DATABASES



No horizontal scalability

Not equipped to handle data fields like graph database, time series and geospatial data

Unable to store massive amounts of data sets

Strict schema

Cannot handle semi-structured or unstructured data

Not Suitable for high-velocity data ingestion

WHAT IS A NOSQL DATABASE?

01 **Distributed database**

02 **Stores high volumes of semi- and unstructured data**

03 **Horizontally scalable**

04 **Cost-effective**

No strict schema

NoSQL VS RDBMS

NoSQL	RDBMS
Distributed database	Relational database
Horizontally scalable	Vertically scalable
Not good for complex queries	Good for complex queries
Dynamic structure	Fixed structure
Uses commodity hardware	Costly storage

KEY TAKEAWAYS

- Relational databases have a strict schema and can only store structured data.
- 80% of today’s data is unstructured.
- NoSQL databases can store massive amounts of unstructured data.
- NoSQL Databases are distributed data stores and follow dynamic structure.

Thank You!

Segment 4

How are NoSQL Databases Designed?

LEARNING OBJECTIVES

What is CAP Theorem?

What characteristics do NoSQL databases have?

Proof of CAP Theorem

CHARACTERISTICS OF DISTRIBUTED DATABASES

Consistency

- All servers in a system contain the same data.
- Users will get the same copy of data regardless of which server answers the request.

Availability

- The system will always be available, i.e., it will always respond to the users' requests.

Partition Tolerance

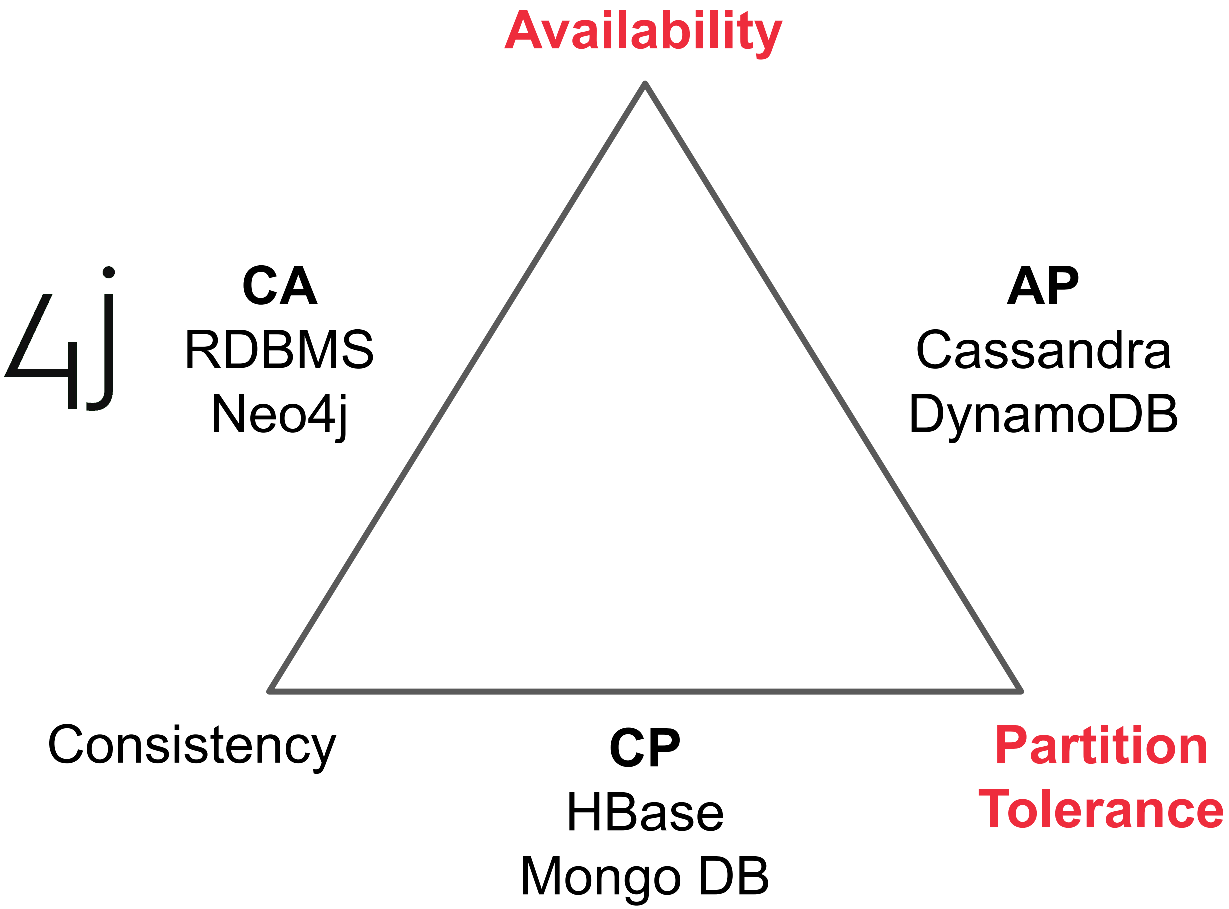
- The system continues to work even if one of the server fails or cannot be reached.

CAP THEOREM

‘A distributed database can achieve **at most two** out of three guarantees: Consistency, **A**vailability and **P**artition **T**olerance.’

CAP THEOREM

neo4j



Availability

CA
RDBMS
Neo4j

AP
Cassandra
DynamoDB

Consistency

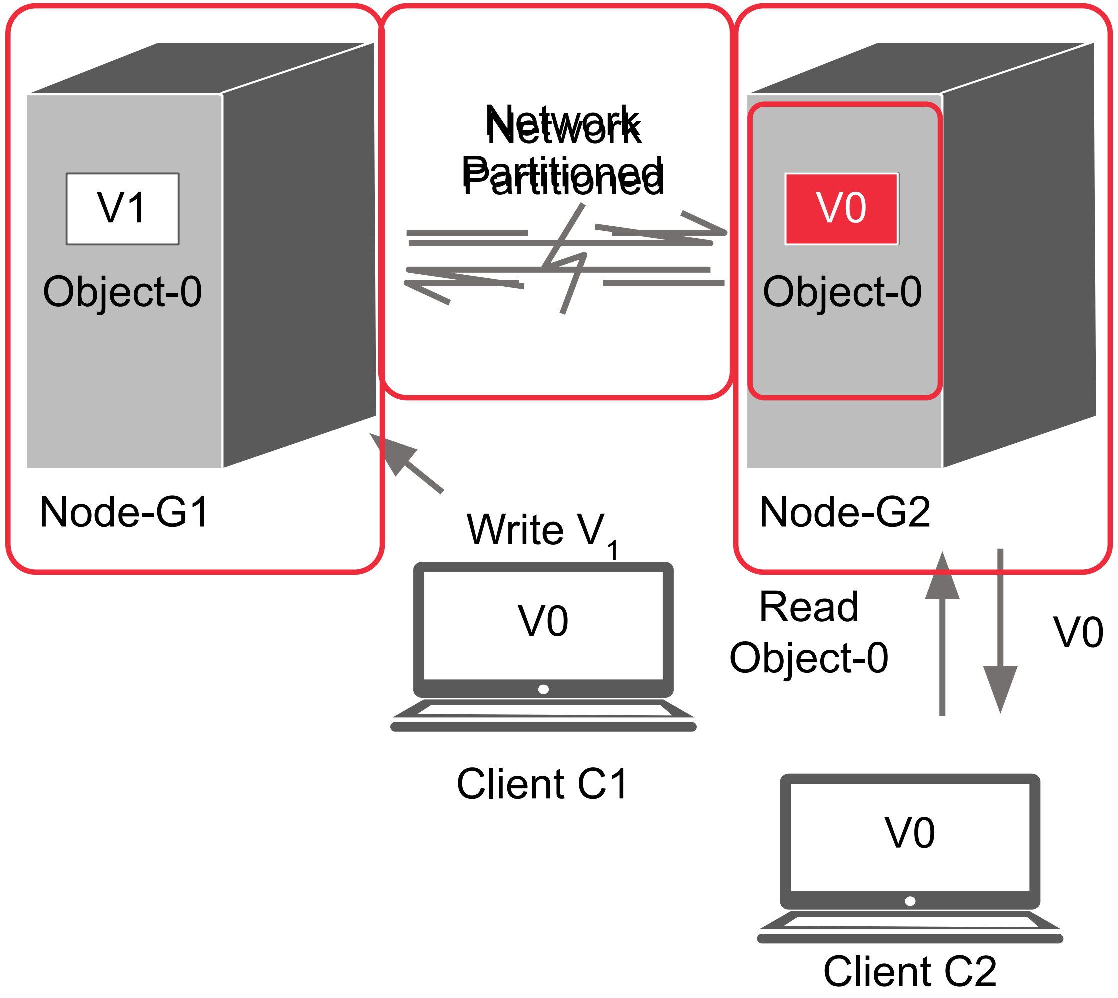
CP
HBase
Mongo DB

Partition Tolerance

cassandra

APACHE
HBASE

PROOF OF CAP THEOREM



EXAMPLE OF CA, CP & AP APPLICATIONS

- **CA Systems-** Transactional data of Bank ATM, Employee data, Data related to hosted website etc.
- **CP Systems-** Messaging Applications like Whatsapp or Banking Websites of various banks etc.
- **AP Systems-** Travel Portals like Make My Trip or Shopping Websites like Amazon, FlipKart etc.

KEY TAKEAWAYS

- CAP theorem states that all the three basic characteristics of a distributed database i.e. consistency, availability and partition tolerance cannot be achieved together at a same time.
- The relational databases provide consistency and availability.
- The NoSQL databases can be either AP or CP in case of a network partition.

Thank You!

Segment 5

Types of NoSQL Databases and Use Cases

LEARNING OBJECTIVES

What are some of the types of NoSQL Databases available?

What are use-cases of NoSQL Databases?

Examples of NoSQL Databases.

TYPES OF NOSQL DATABASES

01

Key-Value Store

Cassandra, DynamoDB and Redis

02

cassandra DynamoDB redis

03

CouchDB relax mongoDB

04

Graph-Based Store

Neo4j

neo4j

NOSQL: INDUSTRY USE CASE

1

Real-time Big Data

1. Massive amounts of data produced by analytics, logging and financial information

2

Semi-Structured Data

1. The data model with tags or other semantic models like XML or JSON

3

Internet of Things

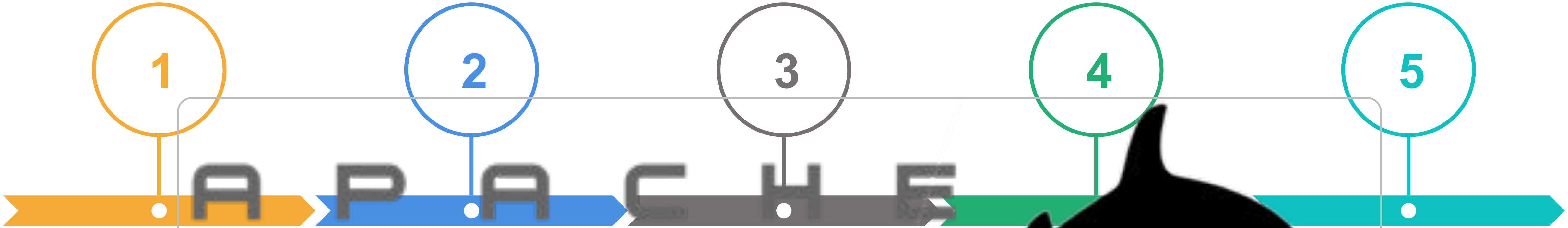
1. Data generated by millions of connected devices and systems

4

Customer 360° View

1. Same customer data shared by multiple applications

APACHE HBASE



- Open-source
- Column-oriented
- Distributed database

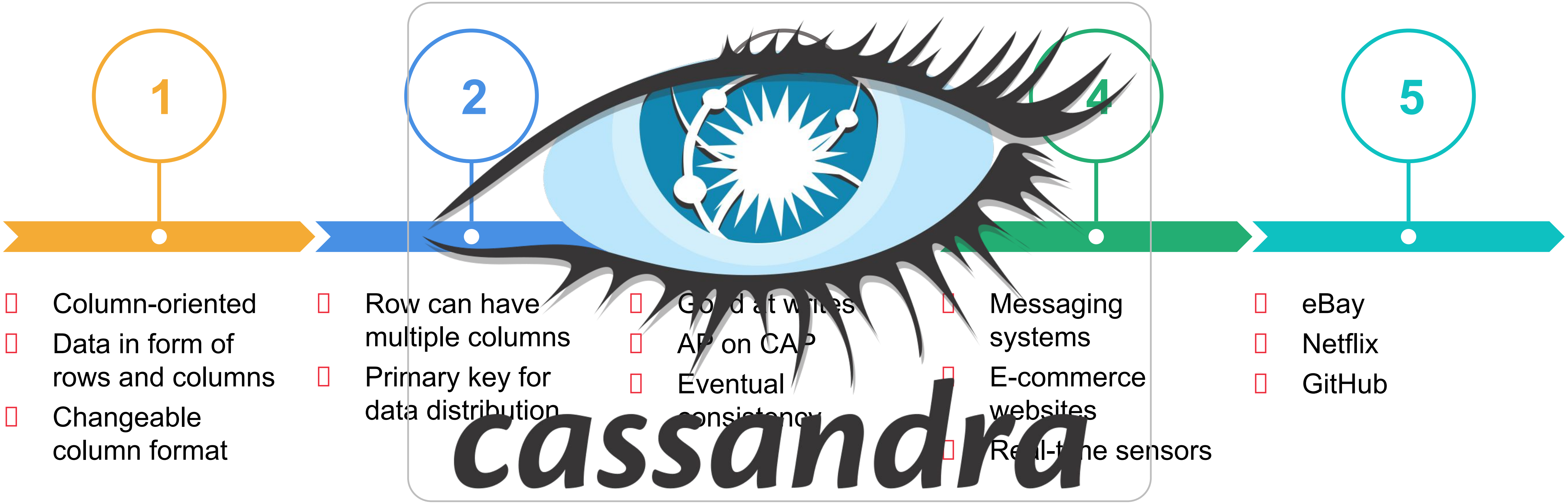
- Storage of large data sets
- Built on top of HDFS
- Distributed file system

- Log analytics
- Write-heavy applications

- C[illegible]
- Optimized for reads
- Immediate consistency

- Yahoo
- Xiaomi
- Salesforce

CASSANDRA



- Column-oriented
- Data in form of rows and columns
- Changeable column format

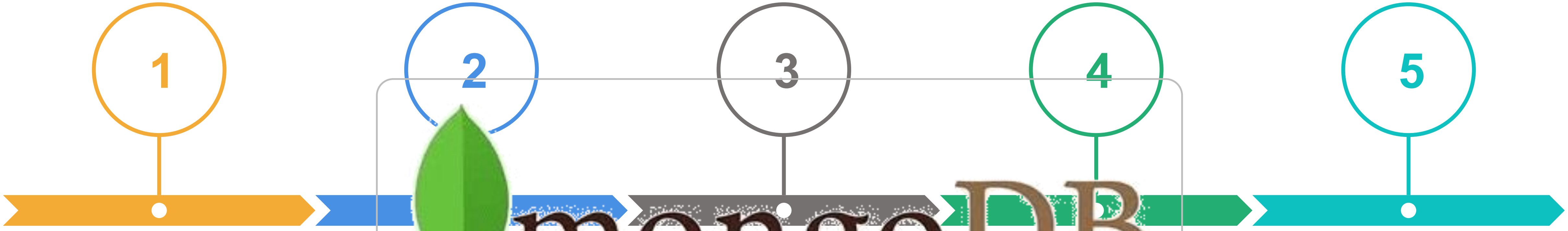
- Row can have multiple columns
- Primary key for data distribution

- Good at writes
- AP on CAP
- Eventual consistency

- Messaging systems
- E-commerce websites
- Real-time sensors

- eBay
- Netflix
- GitHub

MONGODB



- Document store
- Enterprise & community version
- Schema-less database

- JSON like documents for storage
- Faster queries through indexes
- CP on CAP

- High availability
- Horizontal scalability
- Immediate consistency

- Mobile applications
- IOT applications
- Content management systems

- CISCO
- Adobe
- Google

KEY TAKEAWAYS

- Four main types of NoSQL Databases: key-value, document-based, column-based and graph-based.
- NoSQL provides many industry use cases like IoT, big data, etc.
- HBase and Cassandra are column-based NoSQL datastores.
- MongoDB is a document-based datastore.

Thank You!

Segment 6

Introduction to HBase

LEARNING OBJECTIVES

What is Apache HBase?

What features are provided by Apache HBase

How HBase and RDBMS are different from each other?

APACHE HBASE

Distributed data store

- Built on top of HDFS
- Part of the Hadoop Ecosystem

Google's Bigtable

- Open-source implementation of Google's Bigtable

INCEPTION OF HBASE

- In 2007, Mike Cafarella released the code for open-source BigTable implementation.
- It was named HBase.
- Later, in May 2010, HBase became a top-level Apache project.
- **Google** BigTable is a distributed storage system for managing data that is designed to scale to a very large size.
- Many projects at Google store data in BigTable, including web indexing, Google Earth and Google Finance.

APACHE HBASE

Built on top of HDFS

1. Part of the Hadoop Ecosystem
2. Can leverage all the benefits provided by HDFS and Hadoop

Google’s Bigtable

1. Open-source implementation of Google’s Bigtable

Persistent

1. The data will be spread across the multiple commodity hardware

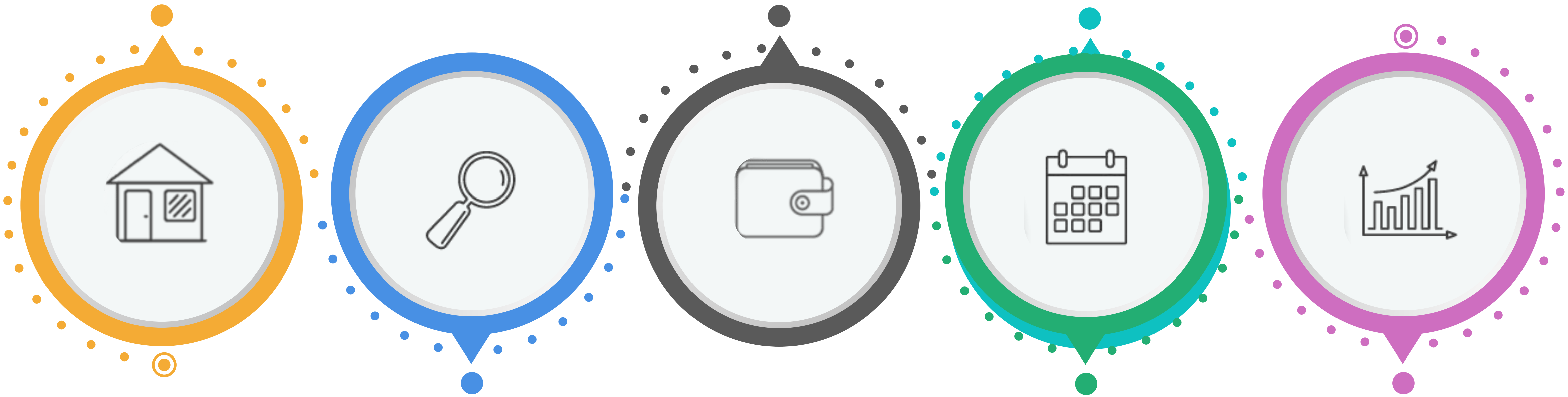
1. Data storage saved after the process it generated ends.

Multi-Dimensional

1. The data is scattered.
2. Sorted map

1. A particular cell can have multiple versions.
2. The fields in rows of HBase tables can be NULL.

HBASE: FEATURES



Horizontally Scalable

Faster Lookups

Distributed Column-Oriented

Sorted by Row-key

Schema-Less

Data Replication

HBASE VS RDBMS

HBase	RDBMS
Column-based data stores	Row-oriented data stores
Flexible schema	Fixed schema
Not optimised for joins	Optimised for joins
Horizontally scalable	Vertically scalable
Good for structured, semi-structured and unstructured data	Good for structured data

KEY TAKEAWAYS

- HBase (Hadoop Database) is a distributed database built on top of HDFS.
- It is an open-source implementation of Google's Bigtable.
- It provides horizontal scalability, faster and random lookups.

Thank You!

Segment 7

Data Model of HBase

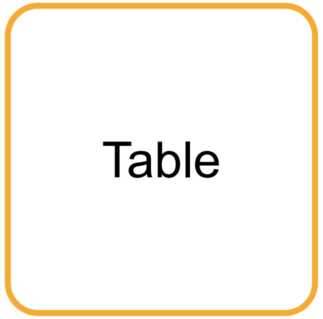
LEARNING OBJECTIVES

What is the structure of HBase table?

How HBase tables are different from relational tables?

What is HBase Columnar data table?

HBASE: DATA MODEL

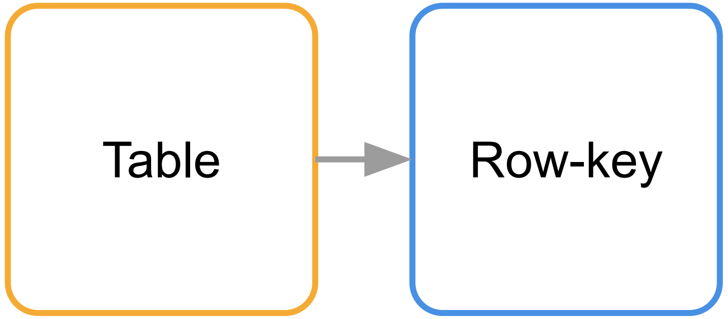


Table

EXAMPLE: HBASE DATA MODEL

Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

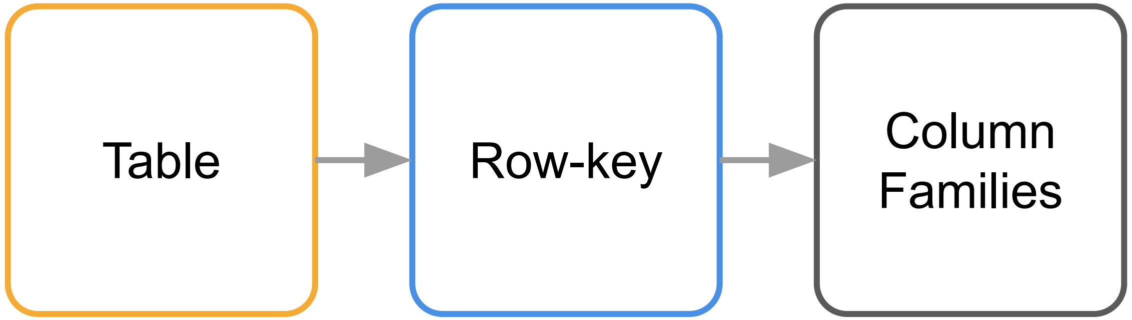
HBASE: DATA MODEL



EXAMPLE: HBASE DATA MODEL

Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

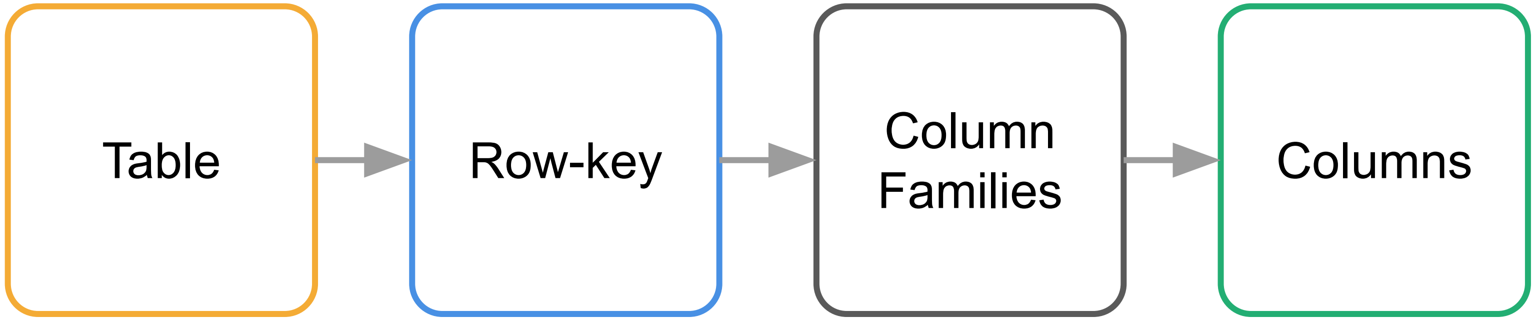
HBASE: DATA MODEL



EXAMPLE: HBASE DATA MODEL

Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

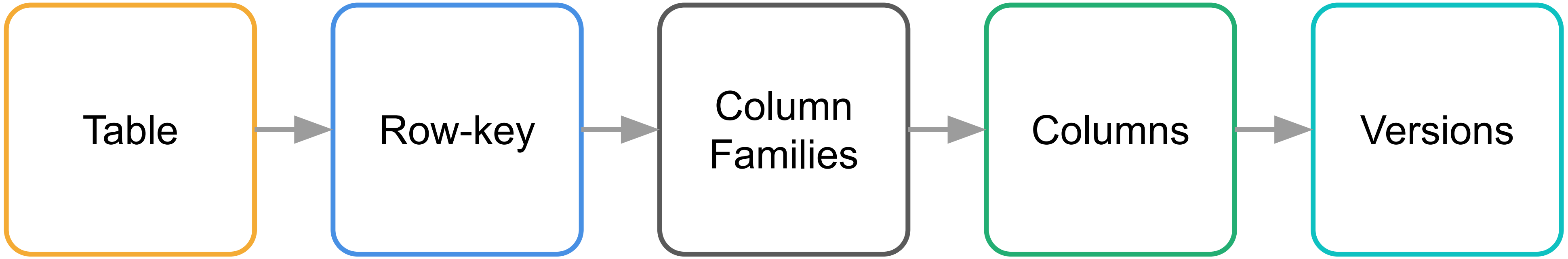
HBASE: DATA MODEL



EXAMPLE: HBASE DATA MODEL

Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

HBASE: DATA MODEL



Table

Row-key

Column Families

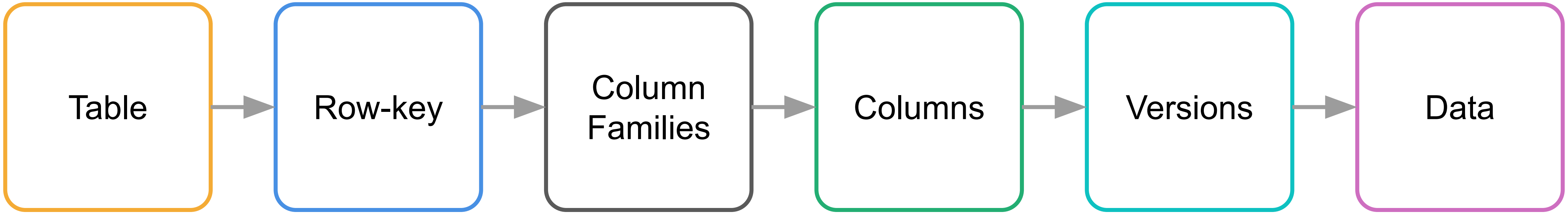
Columns

Versions

EXAMPLE: HBASE DATA MODEL

Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

HBASE: DATA MODEL



EXAMPLE: HBASE DATA MODEL

Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

EXAMPLE: HBASE DATA MODEL

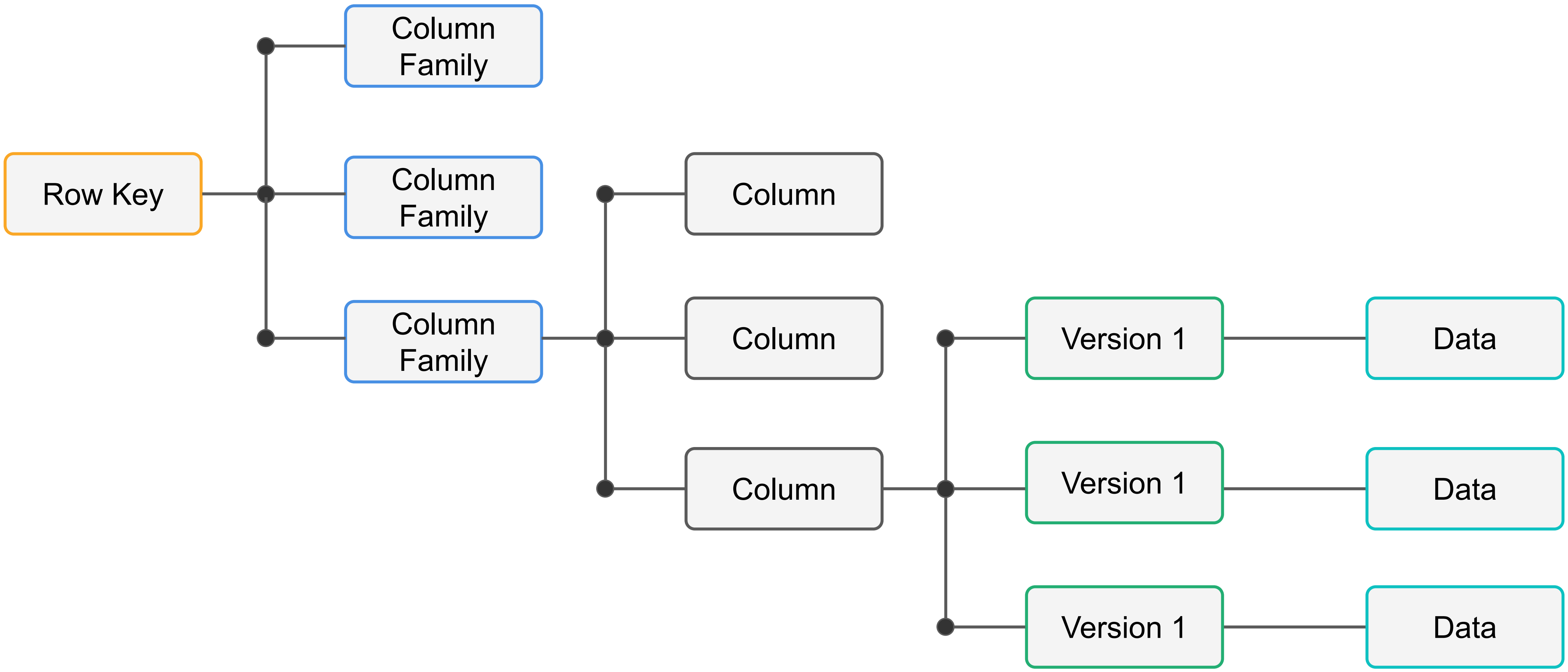
Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

EXAMPLE: HBASE DATA MODEL

Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

DATA MODEL IN HBASE

HBase Four-dimension Data Model



RDBMS MODEL

- Only the latest versions of values available
- Wastage of memory used by NULL value cell

Row Key	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@yahoo.com	9876512345
Row2	Sam	26	Male	sam123@gmail.com	**NULL**
Row3	Mary	30	Female	mary@yahoo.com	8891234561

HBASE COLUMNAR DATA TABLE

Row key	Column Family	Column	Timestamp	Value
Row1	Contact Details	Phone	200	9876512345
Row1	Contact Details	E-mail	200	john@gmail.com
Row1	Contact Details	E-mail	100	john@yahoo.com
Row1	Personal Data	Name	100	John
Row1	Personal Data	Gender	100	Male
Row1	Personal Data	Age	100	25

HBASE COLUMNAR DATA MODEL

- **Timestamp:** The time when a particular value was written in HBase.
- Multiple versions are distinguished by their timestamps.

HBASE COLUMNAR DATA TABLE

Row key	Column Family	Column	Timestamp	Value
Row1	Contact Details	Phone	200	9876512345
Row1	Contact Details	E-mail	200	john@gmail.com
Row1	Contact Details	E-mail	100	john@yahoo.com
Row1	Personal Data	Name	100	John
Row1	Personal Data	Gender	100	Male
Row1	Personal Data	Age	100	25

HBASE COLUMNAR DATA MODEL

- **Timestamp:** The time when a particular value was written in HBase.
- Multiple versions are distinguished by their timestamps.
- All the empty fields from the table are not stored in the Columnar table.

EXAMPLE: HBASE DATA MODEL

Row Key	Personal Data			Contact Details	
	Name	Age	Gender	E-mail	Phone
Row1	John	25	Male	john@gmail.com	9876512345
				john@yahoo.com	🟥
Row2	Sam	26	Male	sam123@gmail.com	
Row3	Mary	30	Female	mary1@gmail.com	7654309876
				mary@yahoo.com	8891234561

HBASE COLUMNAR DATA TABLE

Row key	Column Family	Column	Timestamp	Value
Row1	Contact Details	Phone	200	9876512345
Row1	Contact Details	E-mail	200	john@gmail.com
Row1	Contact Details	E-mail	100	john@yahoo.com
Row1	Personal Data	Name	100	John
Row1	Personal Data	Gender	100	Male
Row1	Personal Data	Age	100	25

HBASE COLUMNAR DATA MODEL

- **Timestamp:** The time when a particular value was written in HBase.
- Multiple versions are distinguished by their timestamps.
- All the empty fields from the table are not stored in the Columnar table.
- **A multi-dimensional map:** The unique key in this view for a value stored in an HBase table is**: <Row key, Column Family:Column, Timestamp>**

HBASE COLUMNAR DATA TABLE

Row key	Column Family	Column	Timestamp	Value
Row1	Contact Details	Phone	200	9876512345
Row1	Contact Details	E-mail	200	john@gmail.com
Row1	Contact Details	E-mail	100	john@yahoo.com
Row1	Personal Data	Name	100	John
Row1	Personal Data	Gender	100	Male
Row1	Personal Data	Age	100	25

HBASE COLUMNAR DATA MODEL

- **Timestamp:** The time when a particular value was written in HBase.
- Multiple versions are distinguished by their timestamps.
- All the empty fields from the table are not stored in the Columnar table.
- **A multi-dimensional map:** The unique key in this view for a value stored in an HBase table is**: <Row key, Column Family:Column, Timestamp>**
- All values are sorted (lexicographical order) **w.r.t. Row Key** for faster lookups.

KEY TAKEAWAYS

- HBase stores data in the form of tables having rows and columns.
- Components of HBase data model-
  - HBase Tables are collection of rows.
  - A row is a collection of column families.
  - A column family comprises of multiple columns.
  - There can be multiple versions of a data value.
  - HBase tables are sorted according to rowkey.
- HBase tables are stored in columnar format.
- A unique key to access data value in columnar table is-
  <Row key, Column Family:Column, Timestamp>

Thank You!

Segment 8

HBase Shell Commands

LEARNING OBJECTIVES

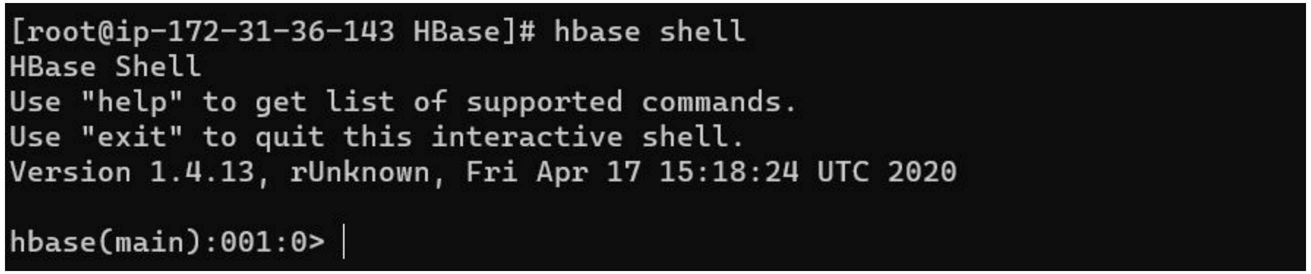
How HBase Shell can be accessed in EMR instance?

How CRUD operations can be performed in HBase Shell?

How filters can be applied while reading data from HBase tables?

DEMO: HBASE SHELL COMMANDS

- **Reference guide**
- **How to start**
  - Login to your EMR instance.
  - Open the HBase shell by running the following command: **hbase shell**



```
[root@ip-172-31-36-143 HBase]# hbase shell
HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
Version 1.4.13, rUnknown, Fri Apr 17 15:18:24 UTC 2020

hbase(main):001:0> |
```

GENERAL COMMANDS

- Status:
  - Provides Status of Cluster
  - Status can be ‘simple’, ‘summary’ or ‘detailed’
  - Syntax: **status**
- Version:
  - Display currently used version
  - Syntax: **version**
- Table Help:
  - Guide for using table referenced commands
  - Syntax: **table_help**

TABLE MANAGEMENT COMMANDS

- Create:
    - Creates a new table in HBase
    - Syntax: **create '<table_name>', '<column_family_name>'**
- List:
    - Displays all the tables present
    - Syntax: **list**
- Describe:
    - Gives information about the mentioned table
    - Syntax: **describe '<table name>'**

TABLE MANAGEMENT COMMANDS

- Disable:
  - Disables the mentioned table
  - Syntax: **disable '<tablename>'**
- Enable:
  - Enables the mentioned table
  - Syntax: **enable '<tablename>'**
- Exists:
  - Verifies the existence of mentioned table
  - Syntax: **exists '<table_name>'**

TABLE MANAGEMENT COMMANDS

- Alter:
  - Used to alter column family schema
  - Used for adding or deleting column families, updating version numbers of column families
- Drop:
  - Drops the mentioned table
  - Syntax: **drop '<tablename>**

DATA MANIPULATION COMMANDS

- Put:
  - Adds cell value to the mentioned table.
  - Syntax: **put '<table_name>', '<row_key>', '<column_value>', '<value>'**
- Get:
  - Fetches data from the table
  - Syntax: **get '<table_name>', '<row_key>', {'<Additional Parameters>'}**

DATA MANIPULATION COMMANDS

- Count:
  - Provides the number of rows present in a table
  - Syntax: **count '<table_name>'**
- Delete:
  - Deletes the cell value in a table
  - Syntax: **delete '<table_name>', '<row_key>', '<column_value>', <timestamp_value>**

DATA MANIPULATION COMMANDS

Get data based on filters: Two input parameters that are a logical operator and a comparator

- **Value Filter:** It compares each **value** with the comparator using the comparison operator.
  - Syntax: **"ValueFilter(<compareOp>, '<value_comparator>')"**
- **Qualifier Filter:** Each **qualifier name** is compared with the comparator using the compare operator.
  - Syntax: **"QualifierFilter(<compareOp>, '<qualifier_comparator>')"**
- **Family Filter:** A FamilyFilter is used to fetch key-values for a specified column family.
  - Syntax: **"FamilyFilter(<compareOp>, '<family_comparator>')"**

DATA MANIPULATION COMMANDS

- Scan:
  - Views all contents of table created.
  - Syntax: **scan '<table_name>'**
- Truncate:
  - Deletes all the data from the table
  - Syntax: **truncate '<table_name>'**

SESSION SUMMARY

- The traditional relational databases cannot store massive amounts of unstructured data.
- NoSQL databases store data in a distributed manner and provide horizontal scalability.
- HBase is a column oriented data store.
- It is an open source implementation of Google’s Bigtable.
- HBase follows a dynamic schema.
- The data in HBase is stored in form of tables having rows and columns.
- Basic CRUD operations can be performed on HBase tables using shell commands.

Thank You!