Big Data Hadoop and Spark Developer

NoSQL Databases: HBase

Learning Objectives

By the end of this lesson, you will be able to:

- Illustrate NoSQL databases and understand their various types
- Analyze the HBase architecture and components
- Explain the applications and real-life connections of HBase
- Explain the HBase Storage Model

NoSQL Introduction

What Is NoSQL Database?

NoSQL databases are non-tabular databases that store data differently from relational tables.

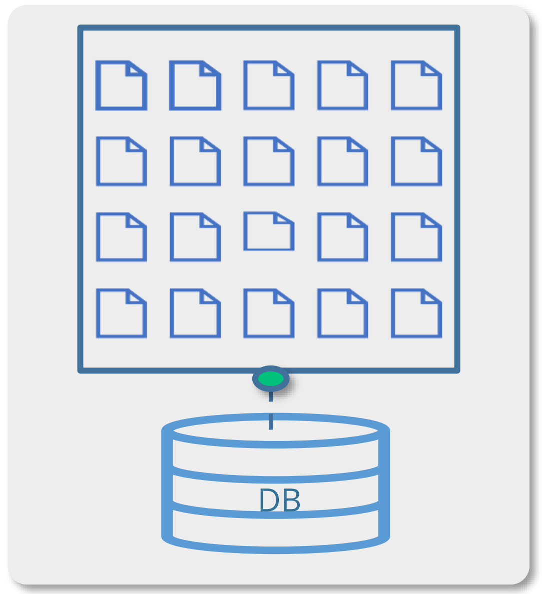
NoSQL is an entirely different database framework that enables high performance and massively scalable dynamic data processing.

It is a database system that has been specifically designed to meet the high demands of big data.

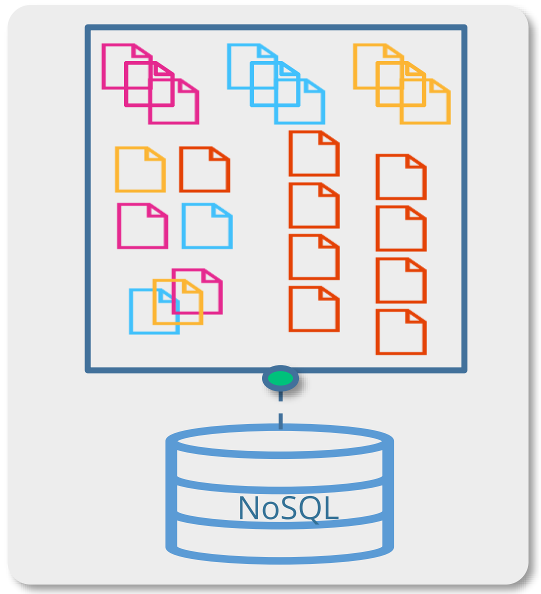
Note

NoSQL is also known as “Not Only SQL.”

Uses of NoSQL Database



Structured data



Unstructured data

- NoSQL is a form of unstructured storage.
- Big data and real-time web apps use NoSQL.
- Twitter, Facebook, and Google are some of the companies that use gigabytes of user data using NoSQL.

Why Is NoSQL Required?

The explosion of social media sites like Facebook, Instagram, Twitter, and others has increased the demand for big data management.

They enable users to store data in a flexible and fluid data architecture. Hence, they can scale up the database tier's capabilities while maintaining 100 percent uptime through replication.

NoSQL is the ideal database solution for handling huge amounts of data.

Types of NoSQL

Following are the four types of NoSQL:

01 Document databases

02 Key-value stores

03 Column-oriented databases

04 Graph databases

Types of NoSQL

Document databases

A document-based database stores data as JSON, BSON, or XML documents.

It is possible to nest documents. To speed up querying, certain elements can be indexed.

E-commerce platforms, trade platforms, and mobile app development are some of its use cases.

Key-value stores

A key-value store is like a relational database with only two columns.

Each data element is recorded as a key-value pair with an attribute name (or "key") and a value in a key-value store.

Shopping carts, user preferences, and user profiles are some of its use cases.

Types of NoSQL

Column-oriented databases

A column store is a collection of columns.

Users can read the columns immediately to analyze them without wasting RAM on irrelevant data.

The data warehouse makes use of a column-based database.

Graph databases

The focus is on relationships between data components.

Each element is saved as a node, such as a person in a social media graph.

Social networking and fraud detection are some of the use cases of the graph-based database.

Types Of NoSQL: Example

Document databases

Example:

MongoDB, CouchDB, Orient DB, Crate DB

Key-value stores

Example:

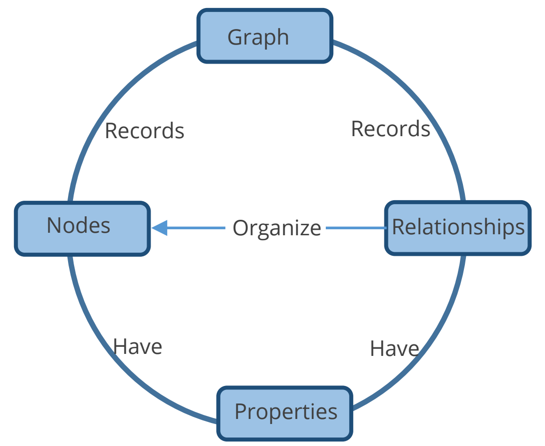
Oracle NoSQL, Redis server, Amazon dynamo

Column-oriented databases

Example:

Bigtable, Cassandra, HBase, Hyper table

Graph databases



Example:

Neo4J, Info Grid, Infinite Graph, Flock DB

RDBMS vs. NoSQL

The following are the differences between RDBMS and NoSQL databases:

Feature	RDBMS	NoSQL Databases
Data storage	Tabular	Variable storage model
Schema	Fixed	Dynamic
Performance	Low	High
Scalability	Vertical	Horizontal
Reliability	Good	Poor

HBase Overview

What Is HBase?

- HBase is a column-oriented, non-relational database management system.
- HBase runs on top of the Hadoop Distributed File System (HDFS).
- HBase provides fault tolerance for storing the database.
- It is the best application for real-time data processing or random reading or writing to huge datasets.

Why HBase?

- HDFS stores, processes, and manages large amounts of data efficiently.
- However, it performs only batch processing, and the data can be accessed sequentially.
- Therefore, a solution is required to access, read, or write data anytime regardless of its sequence in the clusters of data.

Advantages of HBase

Following are the advantages of HBase:

- HBase is consistent.
- It can be scalable to very large datasets (up to petabytes.)
- HBase supports auto sharding.
- It has scalability with hardware resources.

Advantages of HBase

Following are the advantages of HBase:

- It is cost-effective for gigabytes to petabytes of data.
- It is highly available in the case of failure and replication.
- HBase executes queries parallelly across clusters.

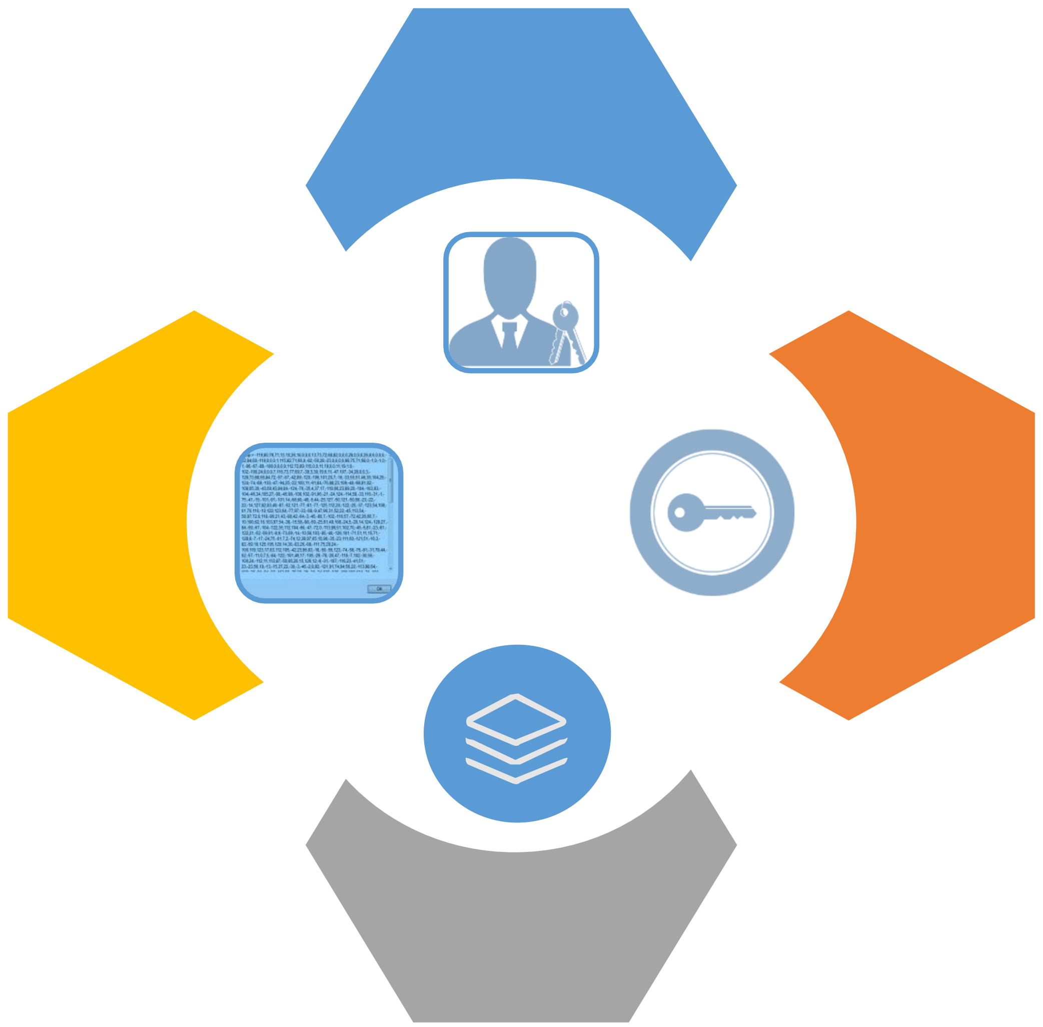
Characteristics of HBase

- HBase is a type of NoSQL database and is classified as a key-value store.
- HBase is a database in which tables have no schema.
- At the time of table creation, column families are defined, not columns.

Characteristics of HBase

The values are quickly accessed through value keys.

The key and value are byte arrays.

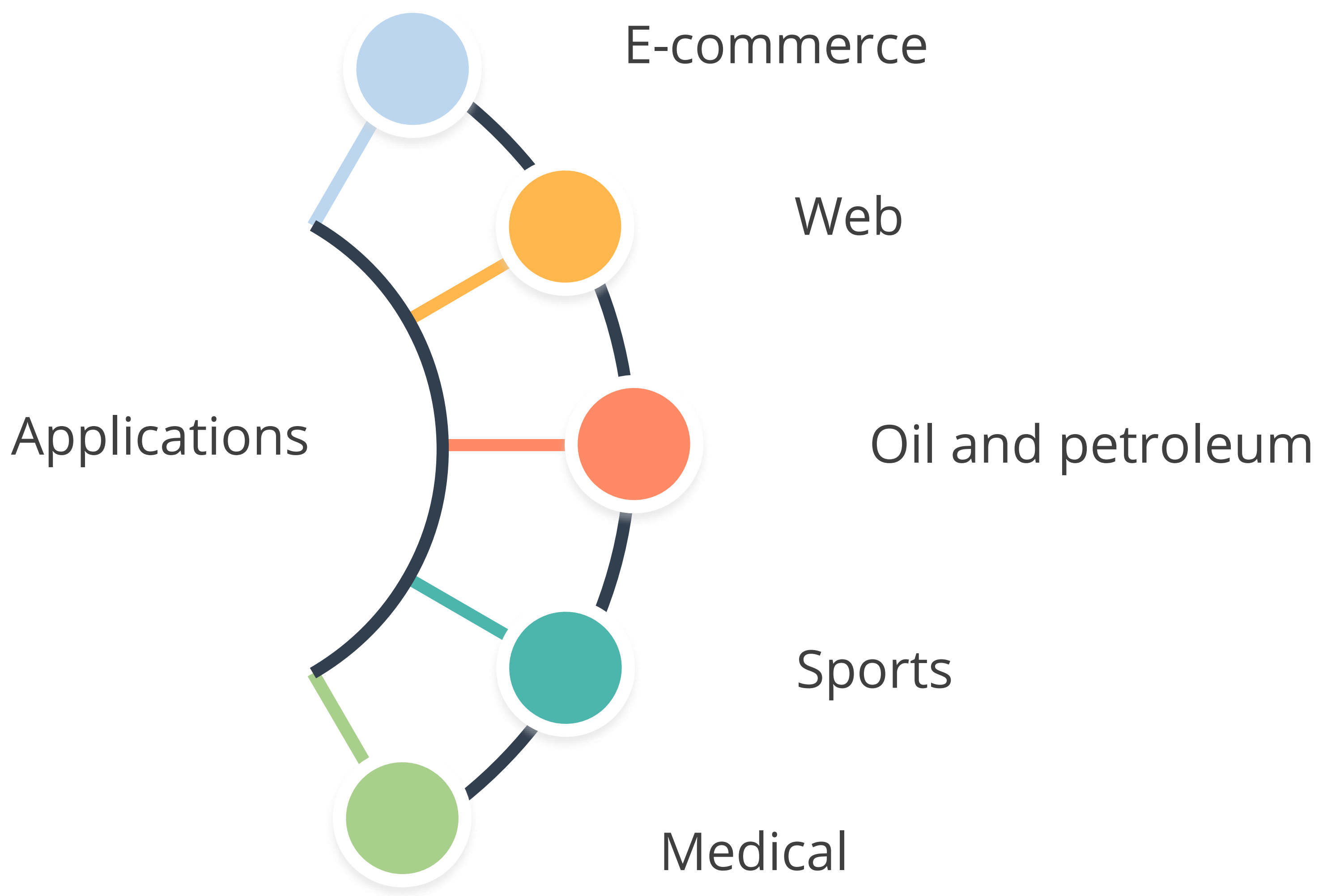


The values are identified with a key.

The values are stored in key-orders.

Applications of HBase

HBase can be applied to the following industries:



HBase: Real-life Connect

The following are examples of some of the companies that are using HBase:

Mozilla

Mozilla stores all crash data in HBase.

Facebook

Facebook stores all the real-time messages in HBase.

HBase: Real-life Connect

The following are examples of some of the companies that are using HBase:

infolinks

Infolinks

Infolink uses HBase to process user advertisement selection.

Twitter

Twitter runs on HBase in its Hadoop cluster.

yahoo!

Yahoo!

Yahoo! uses HBase to store document fingerprints to detect near-duplicates.

HBase Architecture

HBase Architecture

The HBase architecture has the following 3 major components

HBase region server

HBase master server

ZookeepeR

HBase Region Server

Regions are the fundamental building blocks of an HBase cluster, consisting of a distribution of tables and column families.

The region server runs on an HDFS data node in the Hadoop cluster.

Region server regions are responsible for a variety of tasks, including handling, administering, and performing HBase operations on that set of regions.

A region's default size is 256 MB.

HBase Region Server

A table can be divided into several regions.

A region is a sorted set of rows that stores data between two keys, start and finish.

A region server serves a collection of regions to clients. A client can receive about 1000 regions from a region server.

HBase Master Server

- HMaster (HBase master server) is the HBase implementation of the master server.
- It is a process in which regions and DDL (create and delete table) operations are assigned to a region server.
- It keeps track of all-region server instances in the cluster.

HBase Master Server

● The master server is in control of schema modifications and other metadata actions, such as table and column family creation.

● It has a user interface that allows one to create, delete, and update tables.

● This process is accomplished with the help of Apache zookeeper, which assigns regions to region servers.

Zookeeper

- The zookeeper serves as a coordinator within the HBase distributed system.
- Communicating over sessions aids in the maintenance of the server state within the cluster.

Zookeeper

Every region server, in combination with the HMaster server, sends a continuous heartbeat to the zookeeper at regular intervals, checking which servers are alive and available.

The inactive HMaster listens for the notice sent by the active HMaster, while the active HMaster sends heartbeats to the zookeeper. The session is destroyed if the active HMaster fails to send a heartbeat, and the inactive HMaster becomes active.

Zookeeper

If a region server fails to send a heartbeat, the session is terminated, and all listeners are informed. Then, HMaster takes the necessary steps to recover.

Zookeeper also monitors the path of the .META Server, which aids any client in finding any region.

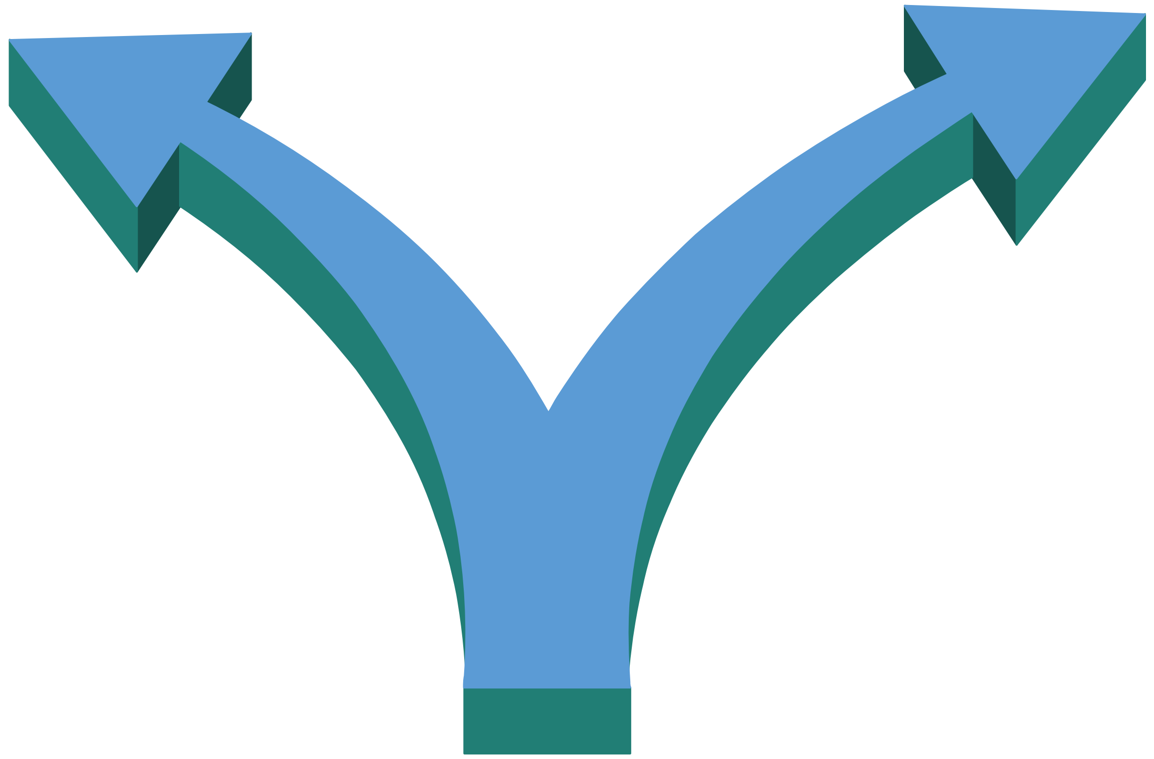
What Is META?

- The META table is a unique HBase catalog table.
- It keeps track of all the HBase storage system’s region servers.
- The table is kept in the form of keys and values in the .META file.
- The key represents the region's start key and IDs, whereas the value represents the path of the region server.

Storage Model of HBase

The two major components of the storage model are as follows:

Partitioning



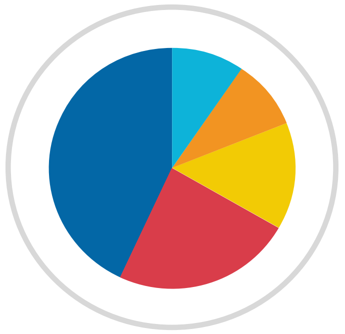
Persistence and data availability

Storage Model of HBase

The two major components of the storage model are as follows:

Partitioning

- A table is horizontally partitioned into regions.
- Each region is managed by a region server.
- A region server may hold multiple regions.



Storage Model of HBase

The two major components of the storage model are as follows:

Persistence and data availability

- HBase stores its data in HDFS, does not replicate region servers and relies on HDFS replication for data availability.
- Memory cache called MemStore serves the updates and reads.

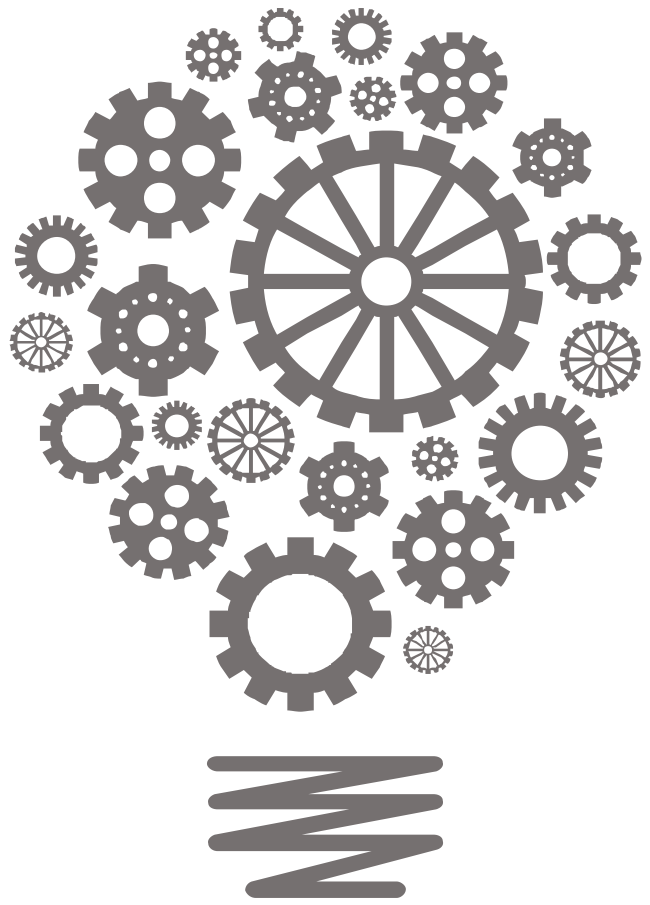
Data Storage in HBase

Data is saved as HFiles or StoreFiles, which are often saved in HDFS.

- HFile is a key-value map.
- When data is added, it is written to a log called the **Write Ahead Log**, and it is stored in MemStore memory.
- HFiles are immutable because HDFS does not support updates to an existing file.
- HBase periodically performs data compactions to control the number of HFiles and keep the cluster well-balanced.

HBase Data Model

HBase Data Model



The HBase Data Model is designed to manage semi-structured data with varying field sizes and data and columns. The layout of the data model divides the data into smaller components and distributes them across the cluster.

Logical Components of HBase Data Model

There are six logical components of the HBase Data Model:

01 Table

02 Row key

03 Column

04 Column family

05 Cell

06 Timestamp

Logical Components of HBase

01

Table: There are various columns inside an HBase table. The tables in HBase are defined during the schema specification process.

02

Row key: A row key identifies one instance of data in a table. Row keys are unique, and they are always considered as a byte[].

03

Column: In HBase, a column is made up of a family and a qualifier of columns, which is identified by a character : (colon).

Logical Components of HBase

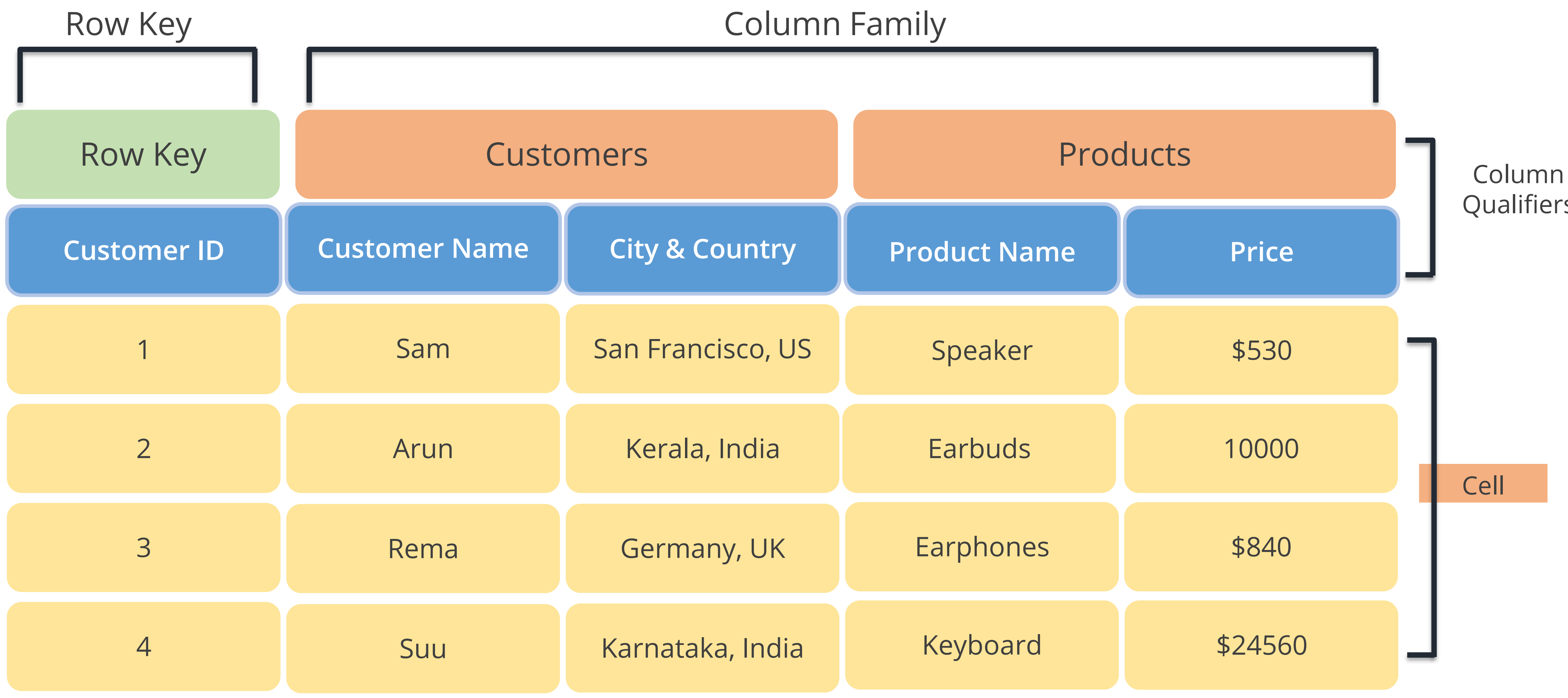
04 **Column family:** Column families are groups of data in a row. Each column family has one or more columns, which are stored together in a low-level storage file called HFile.

05 **Cell:** A cell is a unique combination of row key, column family, and column that stores data.

06 **Timestamp:** The timestamp shows when the data was written to the region server. However, we can assign a different timestamp value to the cell when we enter data.

Logical Components of HBase: Visualization

The following diagram shows the visualization of logical components:



Row Key	Customers		Products	
Customer ID	Customer Name	City & Country	Product Name	Price
1	Sam	San Francisco, US	Speaker	$530
2	Arun	Kerala, India	Earbuds	10000
3	Rema	Germany, UK	Earphones	$840
4	Suu	Karnataka, India	Keyboard	$24560

Storing Multiple Versions of Data

By default, HBase can store 3 different versions of the column, that a timestamp identifies.

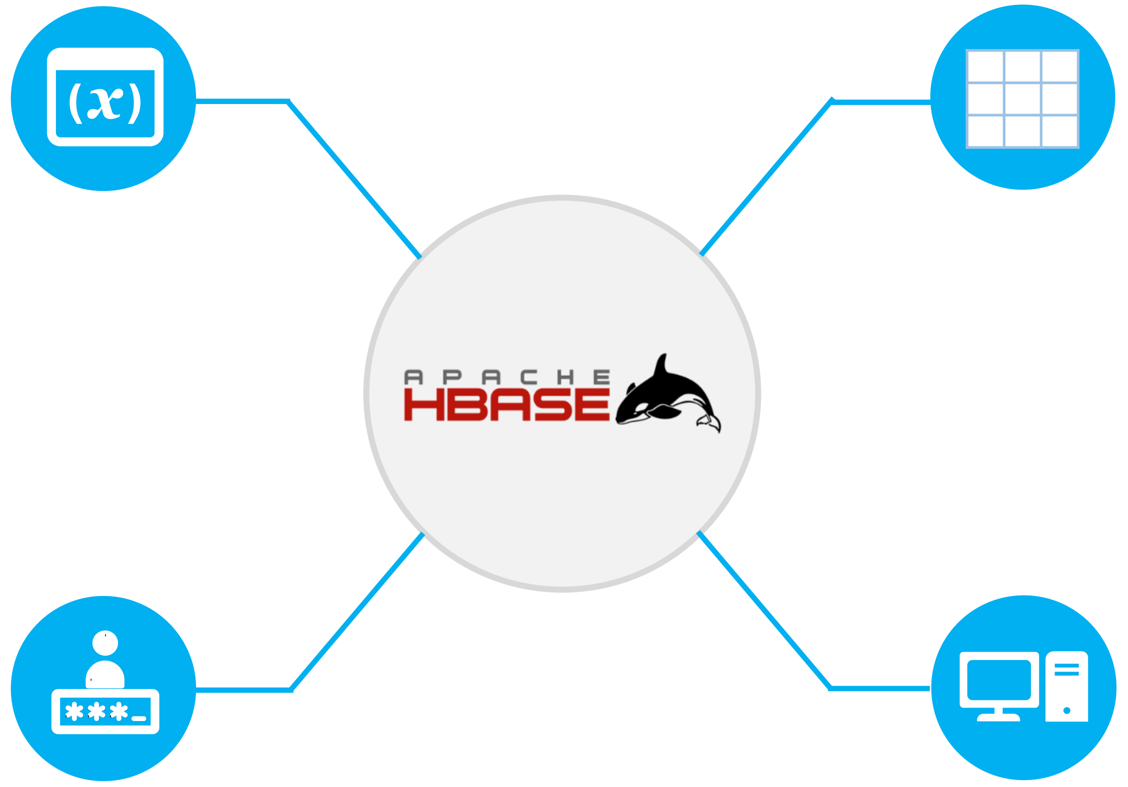
For example, in a table with age as a column, one can insert 13 first, then update to 15, and finally with 23 as the final age.
One needs to enable this through the below-mentioned cmd in the HBase shell:

Command Prompt

```
alter 'your_table_name', age => multi_version_cf', VERSIONS => 3
```

When to Use HBase?

Following are the scenarios to use HBase:



- To utilize HBase invariable schema
- When sufficient data is available in millions or billions of rows
- For random selections and range scans by key
- When commodity hardware is available with at least five nodes

HBase vs. RDBMS

The following table shows a comparison between HBase and a Relational Database Management System (RDBMS):

HBase	RDBMS
It is automatic partitioning.	It is usually manual and admin-driven partitioning.
It scales linearly and automatically with new nodes.	It usually scales vertically by adding more hardware resources.
It uses commodity hardware.	It relies on expensive servers.
It has fault tolerance.	It may or may not have fault tolerance.
It leverages batch processing with MapReduce distributed processing.	It relies on multiple threads or processes rather than MapReduce distributed processing.

HBase vs. HDFS

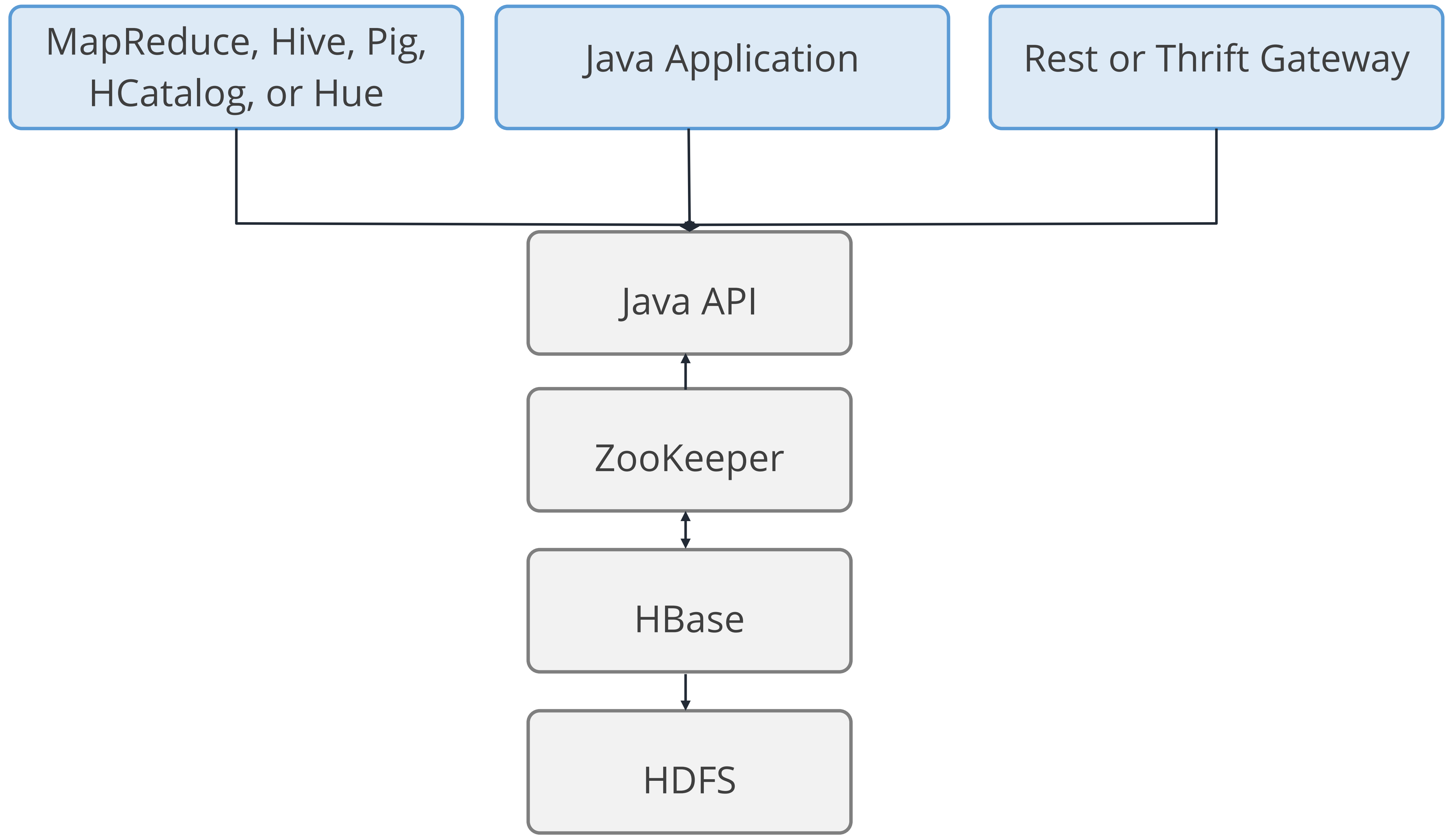
The following table shows a comparison between HBase and a Relational Database Management System (RDBMS):

HBase	HDFS
It provides random reads and writes.	It provides only sequential reads and writes.
It supports variable and flexible architecture.	It supports only rigid architecture.
It is used for real-time processing.	It is generally used for offline batch processing.
It has fault tolerance.	It may or may not have fault tolerance.
It has the lowest latency even for millions of records.	It has a high latency for reading and writing operations.

Connecting to HBase

Connecting to HBase

HBase can be connected through the following media:



HBase Shell Commands

The following are the common shell commands in HBase:

Example

```
HBase> create ‘t1’, {NAME => ‘f1’}, {NAME =>
‘f2’}, {NAME => ‘f3’}
HBase> #

The above in shorthand would be the
following:
HBase> create ‘t1’, ‘f1’, ‘f2’, ‘f3’

Output: It creates a table called t1.
```

This command creates a table.

Note

Pass the table name from a dictionary of specifications per column family and a dictionary of table configuration, if required

HBase Shell Commands

The following are the common shell commands in HBase:

```
Example

HBase> describe ‘t1’

Output: It describes the table which is
created.
```

It describes the created table.

```
Example

HBase> disable ‘t1’

Output: It will disable the table created.
```

Here, the table is disabled.

HBase Shell Commands

The following are the common shell commands in HBase:

Example

```
HBase> drop ‘t1’

Output: It drops the table created.
```

It disables the table and then its name is dropped.

Example

```
HBase> list

Output: It lists all the tables present.
```

It lists all the tables in HBase, and then regular expression parameters filter the output.

HBase Shell Commands

Few more common shell commands in HBase are:

Count
Counts the number of rows in a table

Delete
Deletes a cell value

Get
Gets the contents of a row or a cell

Put
Puts a cell value

Scan
Scans a table's value

Assisted Practice: Data Upload from HDFS to HBase

Duration: 15 Minutes

Problem Scenario: Create a table using namespace in the HBase shell to upload bulk data from HDFS

Objective: To initiate the process of creating a table in HBase for storing bulk data stored on HDFS

Dataset to be Used: data.csv

Steps Overview:
Step 1: Load the data from **data.csv** into HDFS
Step 2: Create a table to store data using the namespace in the HBase shell
Step 3: Execute the command to fetch data from HDFS into the table
Step 4: Verify whether the data is in the table or not by using the scan command

Note: The solution to this assisted practice is provided under the Reference Materials section.

Key Takeaways

- NoSQL databases are non-tabular databases that store data differently from relational databases.
- Document databases, key-value stores, column-oriented databases, and graph databases are the types of NoSQL.
- HBase is a column-oriented, non-relational database management system.
- The three major components of HBase architecture are the HBase region server, HBase master server, and Zookeeper.
- The HBase Data Model manages semi-structured data with varying field sizes, as well as data and columns

Knowledge Check

Knowledge Check

1

Which of the following are the nodes of HBase?

A. Spooldir and Master

B. Syslog and RegionalServer

C. Master and RegionalServer

D. None of the above

Knowledge Check

1

Which of the following are the nodes of HBase?

A. Spooldir and Master

B. Syslog and RegionalServer

C. Master and RegionalServer

D. None of the above

The correct answer is **C**

Master and RegionalServer are the nodes of HBase, whereas the other options are parts of Flume.

Knowledge Check

2

HBase can be used in which of the following scenarios?

A. For random selects and range scans by key

B. For sufficient commodity hardware with at least five nodes

C. In variable schema

D. All the above

Knowledge Check

2

HBase can be used in which of the following scenarios?

A. For random selects and range scans by key

B. For sufficient commodity hardware with at least five nodes

C. In variable schema

D. All the above

The correct answer is **D**

HBase can be used for random selects and range scans by key, for sufficient commodity hardware with at least five nodes, and in invariable schema.

Lesson-End Project: Uploading Data from HDFS to HBase

Problem Scenario:

Angelo was working as a DB admin for Cassandra at DBS Bank. The bank decided to migrate all its data to Hadoop. However, they encountered problems while transferring the data from Cassandra to HBase. After some research, Angelo suggested that both databases are columnar so they can migrate data table by table. However, there is an issue with HBase column families, which must be built before data transfer. As suggested, the database team must begin the migration using the user table data, which includes information about the personal and professional lives of employees.

Objective: To create a table in HBase to store bulk data that is stored on HDFS

Dataset to be Used: data.csv

Lesson-End Project: Uploading Data from HDFS to HBase

Steps Overview:

1. Download the dataset **data.csv** from the **Reference Materials section**
2. Load the dataset into HDFS
3. Design the schema with personal and professional columns and create a table to store data using the namespace in the HBase shell
4. Display the metadata information table
5. Insert the data into the table
6. Ensure that the data is present
7. Delete the data
8. Execute the command to fetch bulk data from HDFS into the table
9. Verify whether the data is present by using the scan command

Thank You