Big Data Hadoop and Spark Developer

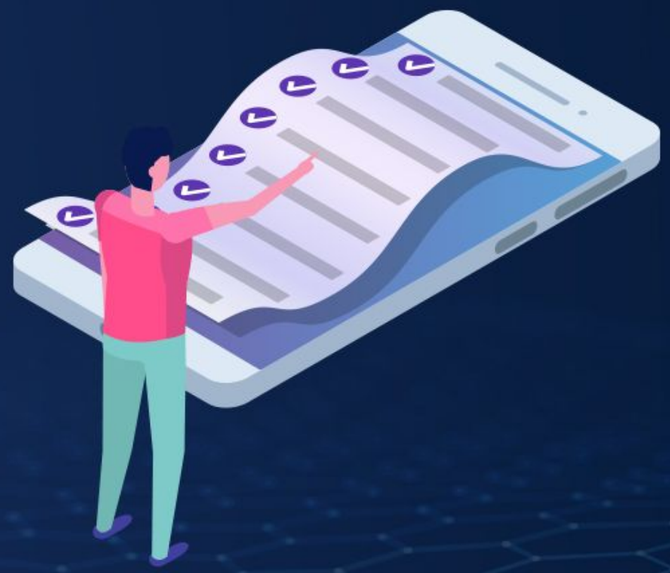
simplilearn

NoSQL Databases: HBase

Learning Objectives

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

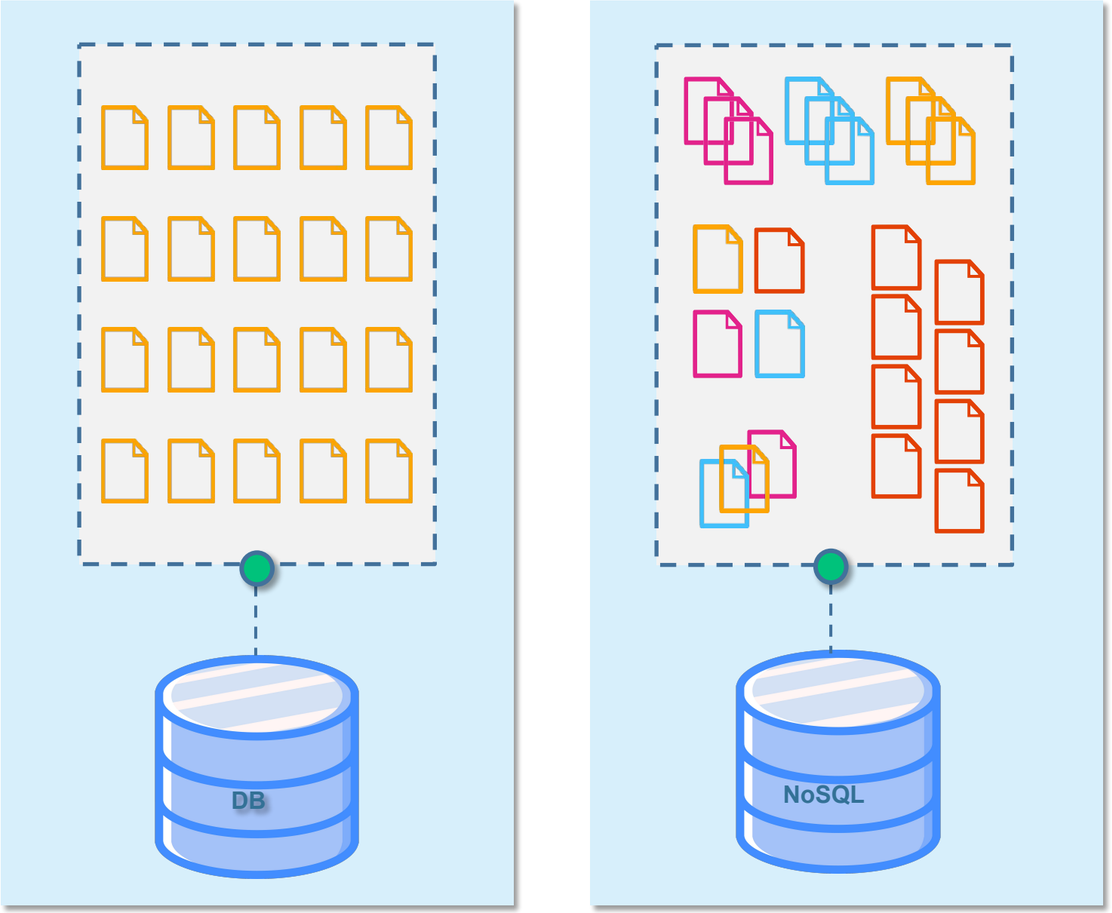
- Understand the need for NoSQL databases
- Analyze the HBase architecture and components
- Distinguish HBase from RDBMS



NoSQL Introduction

NoSQL Database

NoSQL is a form of unstructured storage.



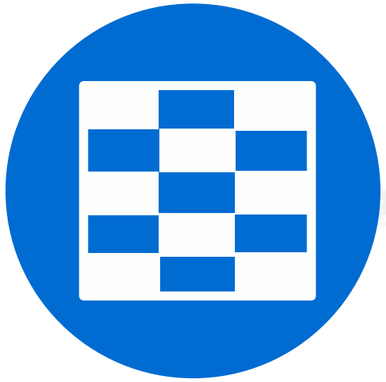
Structured **Unstructured**

Why NoSQL?

With the explosion of social media sites, such as Facebook and Twitter, the demand to manage large data has grown tremendously.

Key-Value Pair Databases

Document Databases



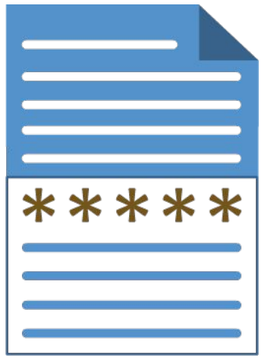
Column-Based Data Stores

Types of NoSQL

Key-Value

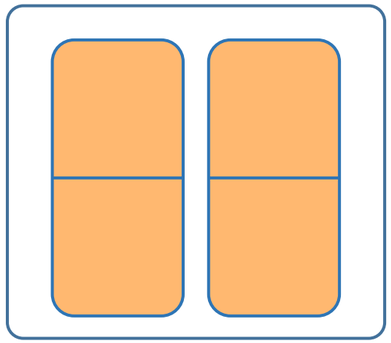
Example:
Oracle NoSQL, Redis Server, Scalaris

Document-Based



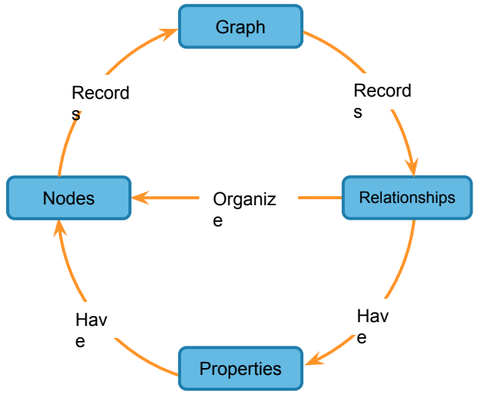
Example:
MongoDB, CouchDB, OrientDB, RavenDB

Column-Based



Example:
BigTable, Cassandra, HBase, Hypertable

Graph-Based



Example:
Neo4J, InfoGrid, Infinite Graph, FlockDB

RDBMS vs. NoSQL

The differences between RDBMS and NoSQL databases are as follows:

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

YARN Tuning

Duration: 15 mins

Problem Statement: In this demonstration, you will learn, how to tune YARN and allow HBase to run smoothly without being resource starved.

Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

HBase Overview

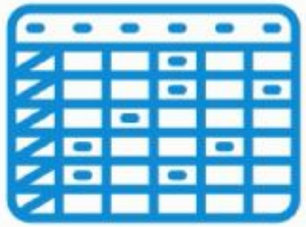
What Is HBase?

HBase is a database management system designed in 2007 by Powerset, a Microsoft company.

HBase rests on top of HDFS and enables real-time analysis of data.

What Is HBase?

It can store huge amount of data in tabular format for extremely fast reads and writes.



HBase is mostly used in a scenario that requires regular and consistent inserting and overwriting of data.

Why HBase?

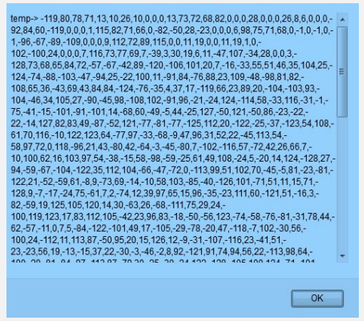
HDFS stores, processes, and manages large amounts of data efficiently.
However, it performs only batch processing and the data will be accessed in a sequential manner.

Therefore, a solution is required to access, read, or write data anytime regardless of its sequence in the clusters of data.

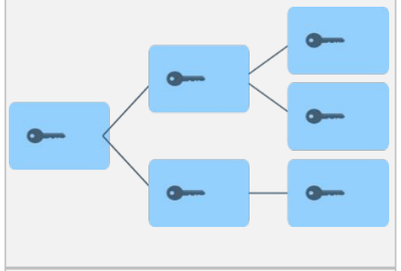
Characteristics of HBase

HBase is a type of NoSQL database and is classified as a key-value store. In HBase:

Value is identified with a key



Key and value are a ByteArray



Values are stored in key-orders

Quickly accessed by value keys

HBase is a database in which tables have no schema. At the time of table creation, column families are defined, not columns.

HBase: Real-Life Connect

Facebook's messenger platform needs to store over 135 trillion messages every month.

Where do they store such data?

Rarely Accessed Dataset

Highly Volatile Dataset

HBase Architecture

HBase Architecture

HBase has two types of nodes: Master and RegionServer. Their characteristics are as follows:

Master

- Single Master node running at a time
- Manages cluster operations
- Not a part of the read or write path

HBase Nodes

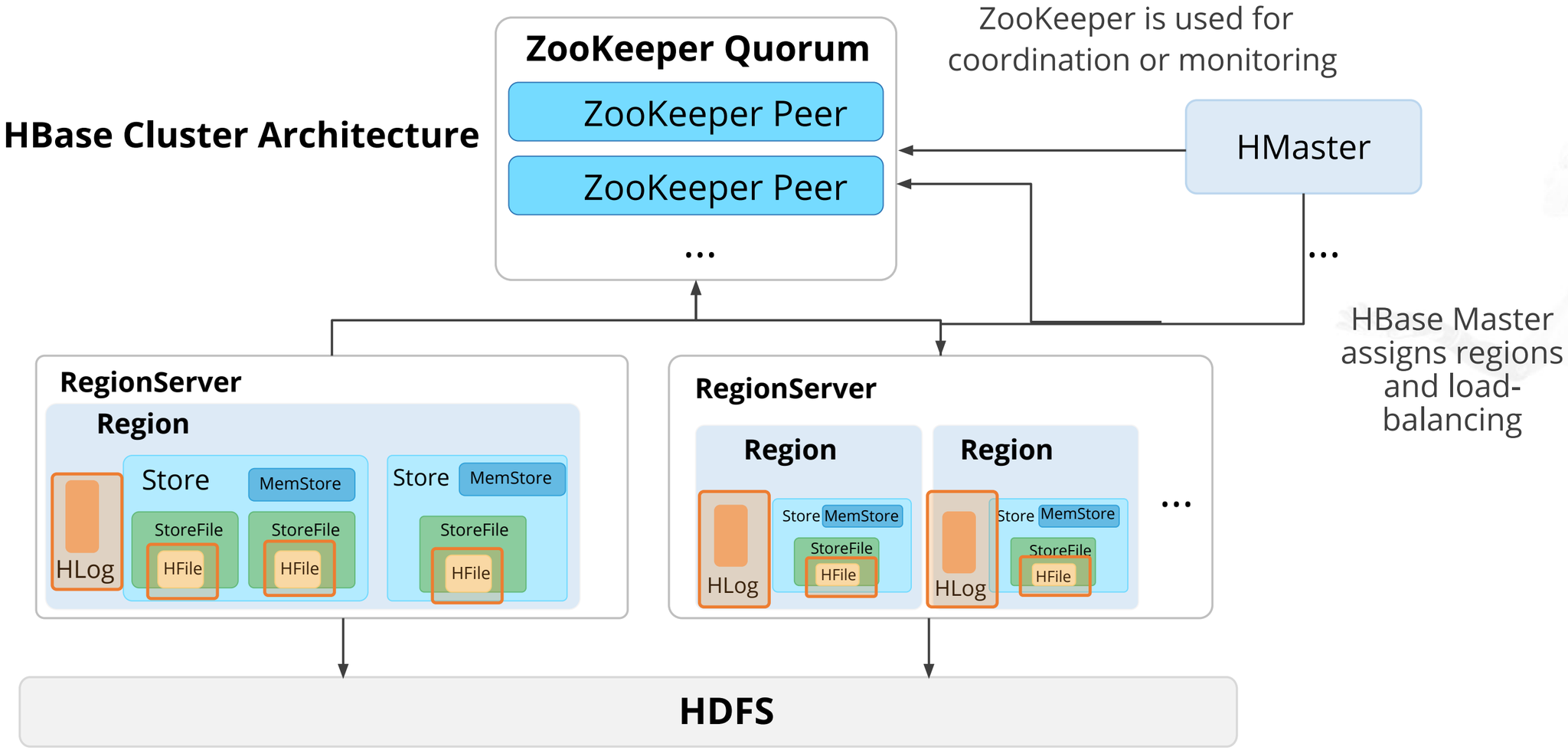
RegionServer

- One or more RegionServers running at a time
- Hosts tables and performs reads and buffer writes
- RegionServer is communicated in order to read and write

A region in HBase is the subset of a table's rows. The Master node detects the status of RegionServers and assigns regions to it.

HBase Components

The HBase components include HBase Master and multiple RegionServers.



Storage Model of HBase

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

Partitioning:

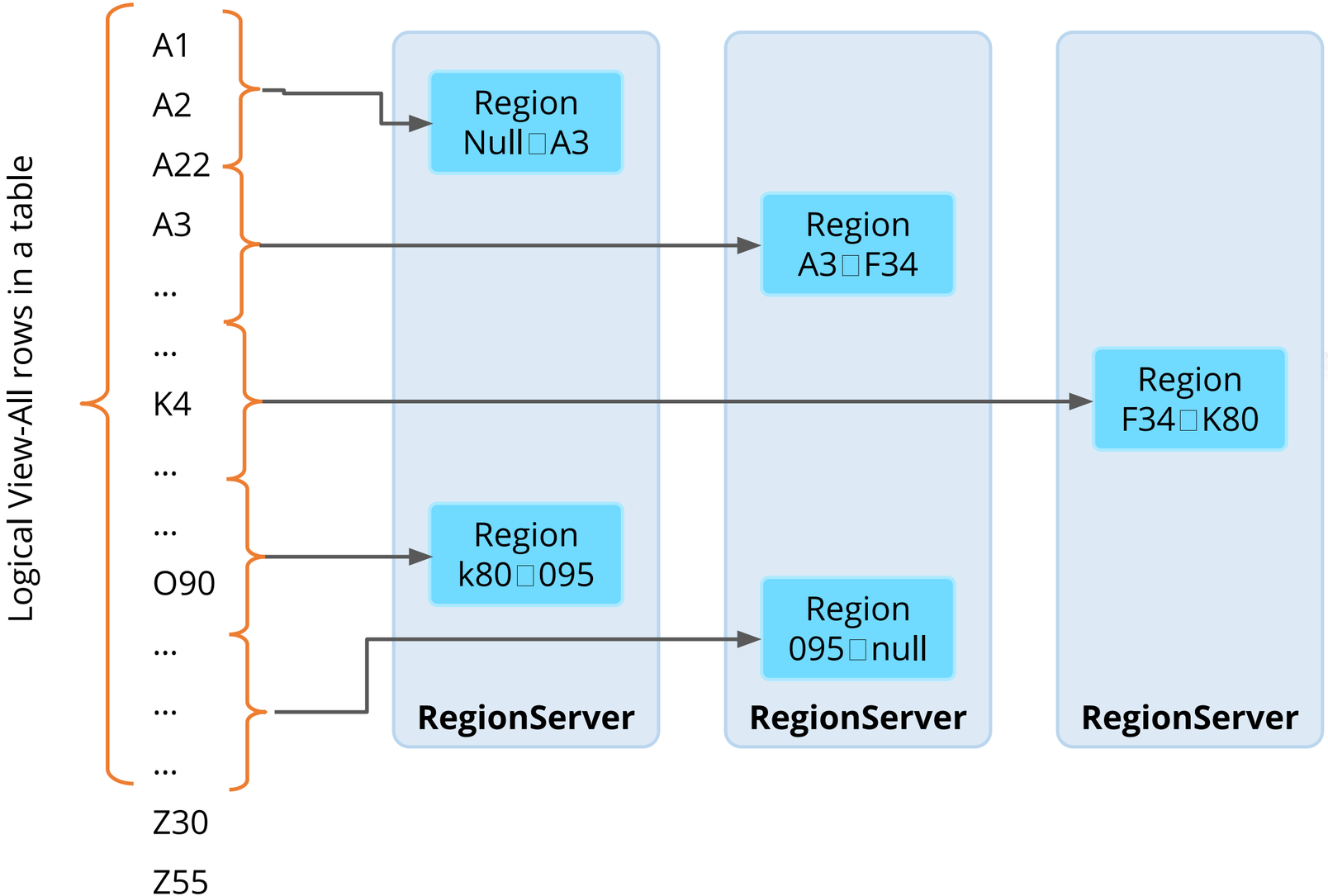
- A table is horizontally partitioned into regions.
- Each region is managed by a RegionServer.
- A RegionServer may hold multiple regions.

Persistence and data availability:

- HBase stores its data in HDFS, does not replicate RegionServers, and relies on HDFS replication for data availability.
- Updates and reads are served from the in-memory cache called MemStore.

Row Distribution of Data between RegionServers

The distribution of rows of structured data using HBase is illustrated here:



Data Storage in HBase

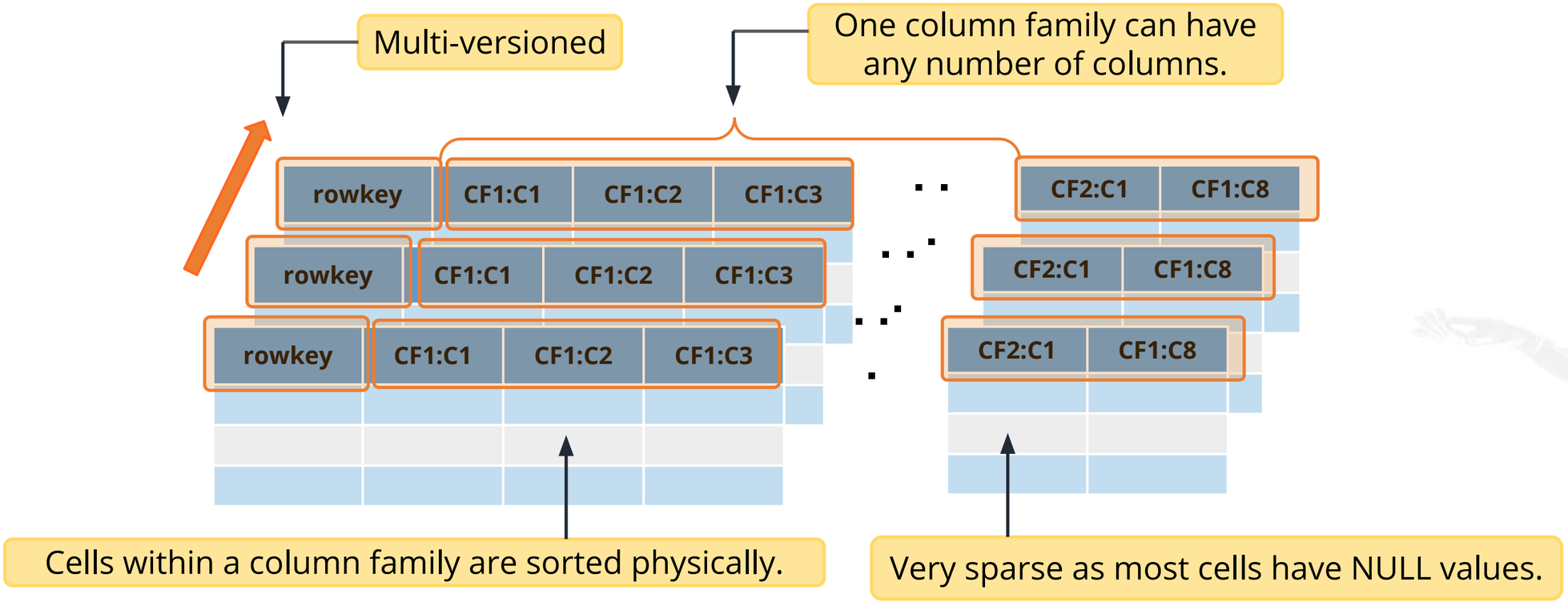
Data is stored in files called HFiles or StoreFiles that are usually saved in HDFS.

- 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 memory, MemStore.
- HFiles are immutable, since HDFS does not support updates to an existing file.
- HBase periodically performs data compactions to control the number of HFiles and to keep the cluster well-balanced.

Data Model

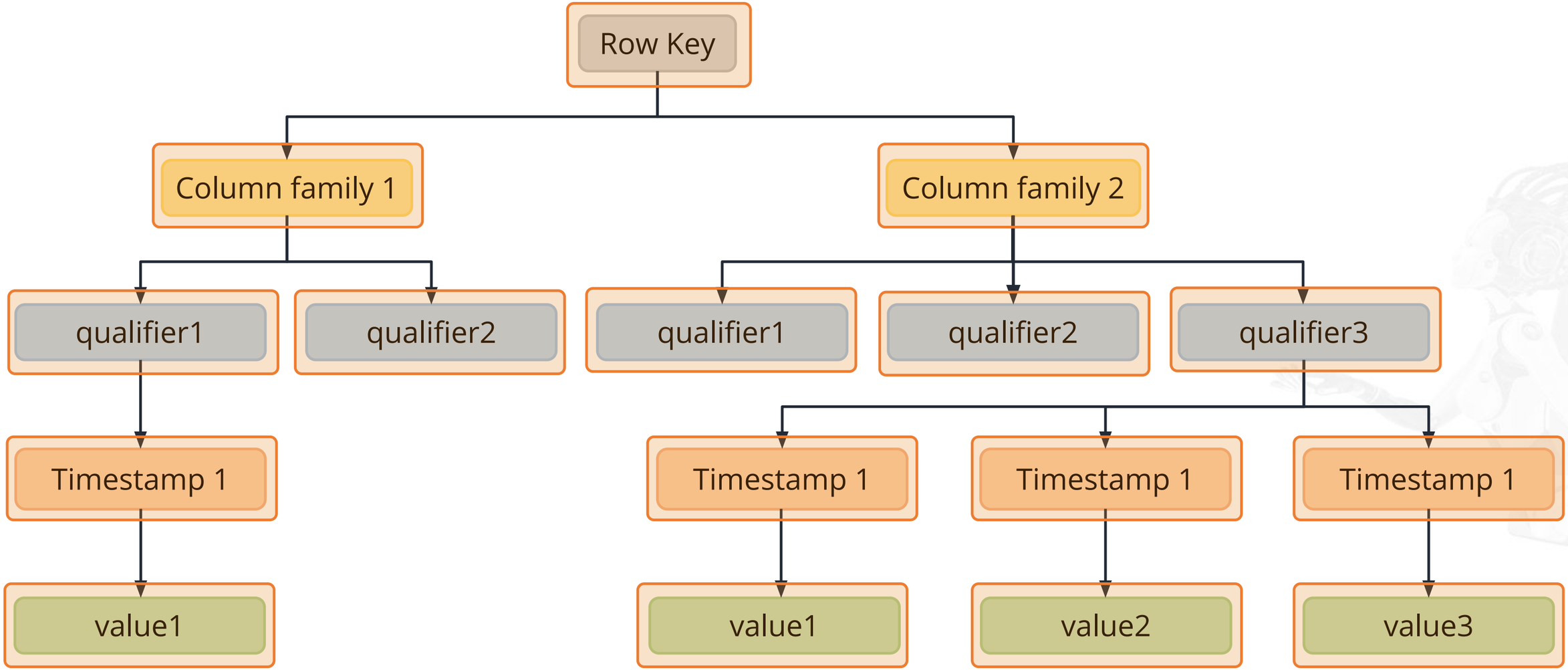
Data Model

Following are the features of the data model in HBase:

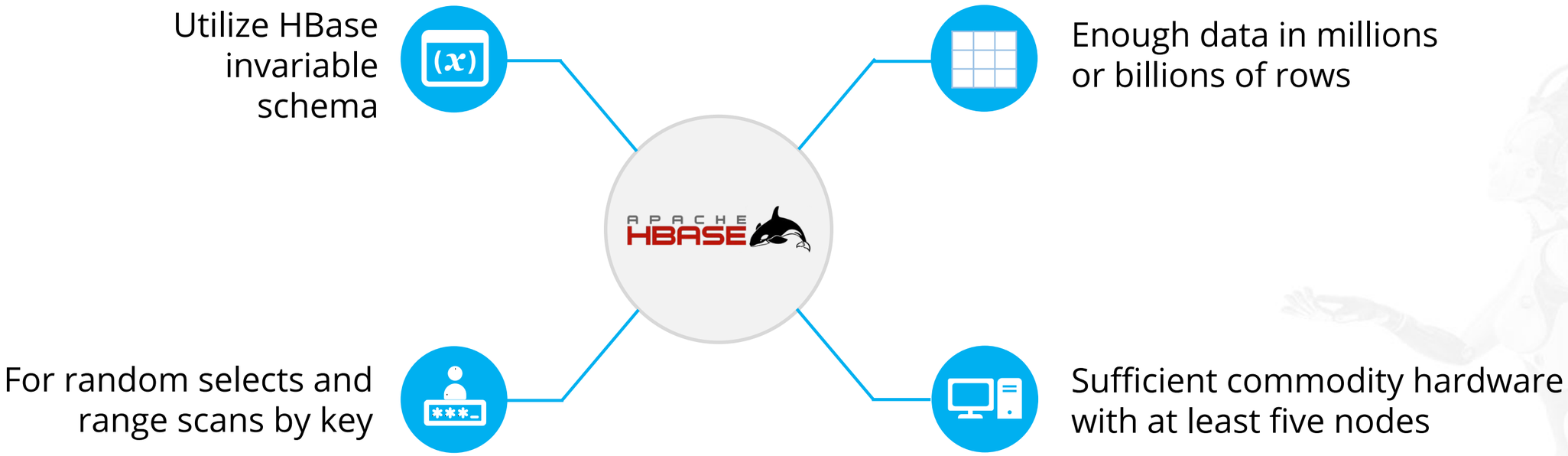


Everything except table names are stored as ByteArrays.

Data Mode: Features



When to Use HBase?



HBase vs. RDBMS

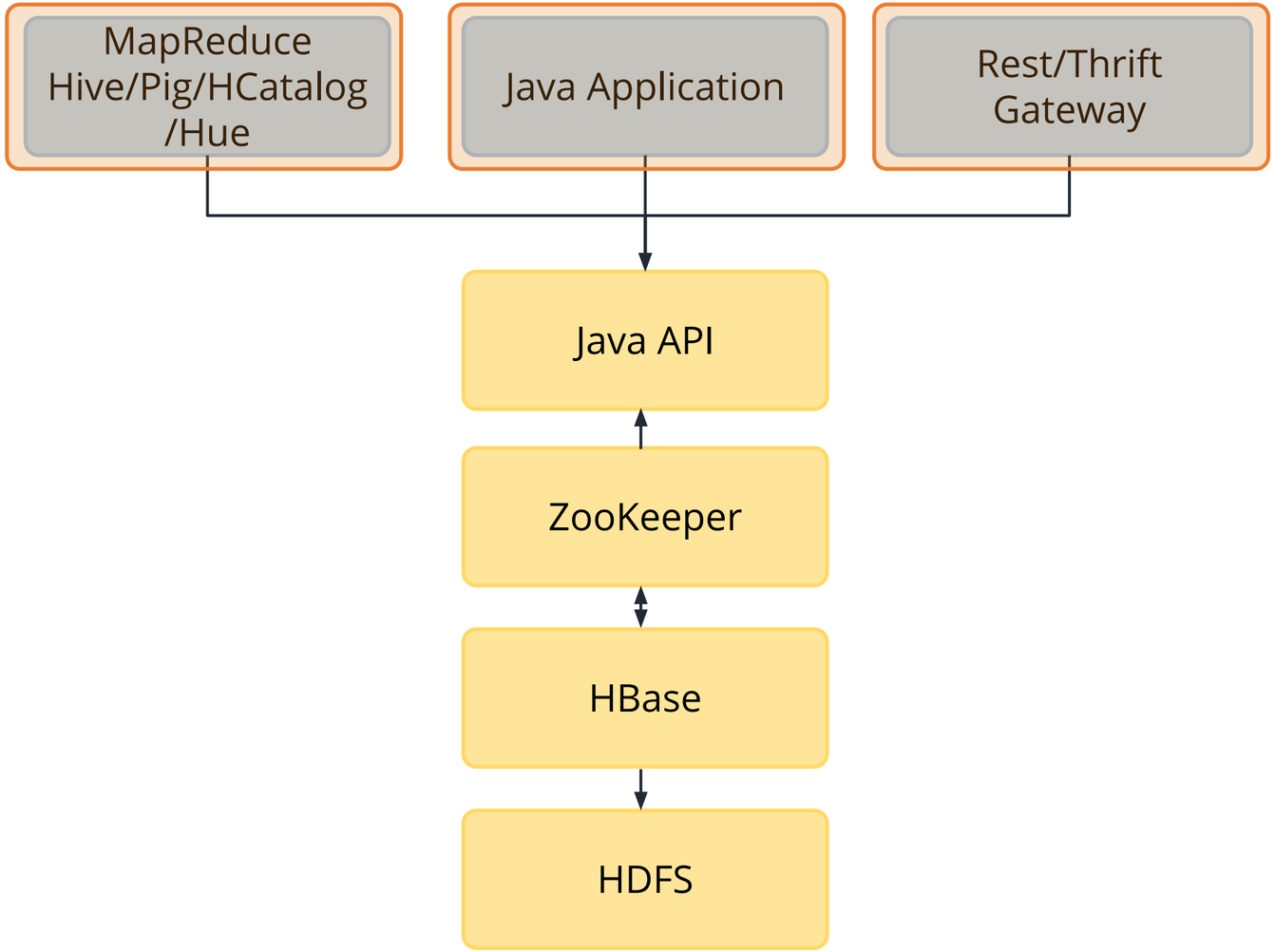
The table shows a comparison between HBase and a Relational Database Management System (RDBMS):

HBase	RDBMS
Automatic partitioning	Usually manual and admin-driven partitioning
Scales linearly and automatically with new nodes	Usually scales vertically by adding more hardware resources
Uses commodity hardware	Relies on expensive servers
Has fault tolerance	Fault tolerance may or may not be present
Leverages batch processing with MapReduce distributed processing	Relies on multiple threads or processes rather than MapReduce distributed processing

Connecting to HBase

Connecting to HBase

HBase can be connected through the following media:



HBase Shell Commands

Common commands include, but are not limited to, the following:

Create table. Pass table name from a dictionary of specifications per column family, and a dictionary of table configuration which is optional	HBase> create ‘t1', {NAME => ‘f1'}, {NAME => ‘f2'}, {NAME => ‘f3'} HBase> # The above in shorthand would be the following: HBase> create ‘t1', ‘f1', ‘f2', ‘f3'
Describe the table named	HBase> describe ‘t1'
Start the disabling of the table named	HBase> disable ‘t1'
Drop the table named. Table must first be disabled	HBase> drop ‘t1'
List all tables in HBase. Optional regular expression parameter can be used to filter the output.	HBase> list

HBase Shell Commands

Count
Counting the number of rows in a table

Delete
Deleting a cell value

Get
Getting the contents of a row or a cell

Put
Putting a cell value

Scan
Scanning a table's value

HBase Shell

Duration: 15 mins

Problem Statement: Create a sample HBase table on the cluster, enter some data, query the table, then clean up the data and exit.

Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

Steps to Perform

- **HBase Shell**

// Start the HBase shell
hbase shell

// Create a table called simplilearn with one column family named stats:
create 'simplilearn', 'stats'

// Verify the table creation by listing everything
list

// Add a test value to the daily column in the stats column family for row 1:
put 'simplilearn', 'row1', 'stats:daily', 'test-daily-value'

Steps to Perform

- **HBase Shell**

// Add a test value to the weekly column in the stats column family for row 1:
put 'simplilearn', 'row1', 'stats:weekly', 'test-weekly-value'

// Add a test value to the weekly column in the stats column family for row 2:
put 'simplilearn', 'row2', 'stats:weekly', 'test-weekly-value'

// Type **scan 'simplilearn'** to display the contents of the table.

// Type **get 'simplilearn', 'row1'** to display the contents of row 1.

Type **disable 'simplilearn'** to disable the table.
Type **drop 'simplilearn'** to drop the table and delete all data.
Type **exit** to exit the HBase shell.

NoSQL Graph Database

NoSQL Graph Database

A database designed to treat the relationships between data as equally important to the data itself.

It is intended to hold data without constricting it to a predefined model.

It focuses on the relationships between entities and is able to infer new knowledge out of existing information.

Why Graph Databases?

Accessing nodes and relationships in a native graph database is an efficient, constant-time operation and allows you to quickly traverse millions of connections per second per core.

Independent of the total size of your dataset, graph databases excel at managing highly connected data and complex queries.

Property Graph Model

Nodes	Relationships
Nodes are the entities in the graph. Nodes can be tagged with *labels*, representing their different roles in your domain.	*Relationships* provide directed, named, semantically relevant connections between two node entities. It always has a direction, a type, a start node, and an end node.

NoSQL Graph Database

Duration: 15 mins

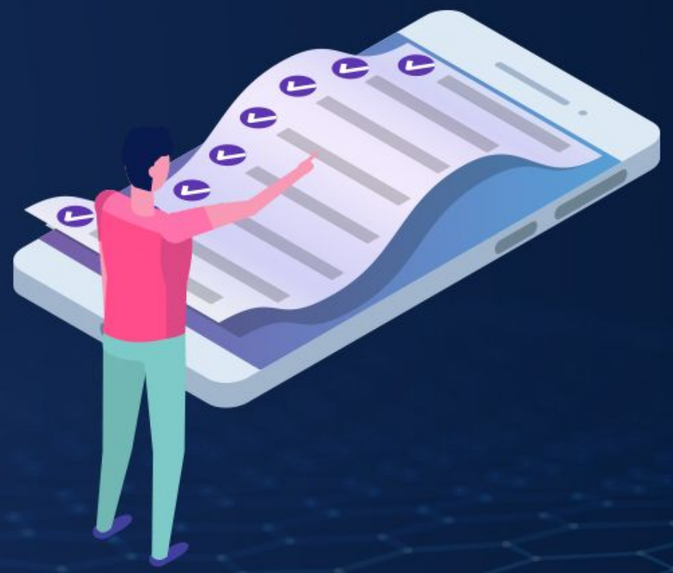
Problem Statement: In this demonstration, you will learn, how to create a NoSQL graph database.

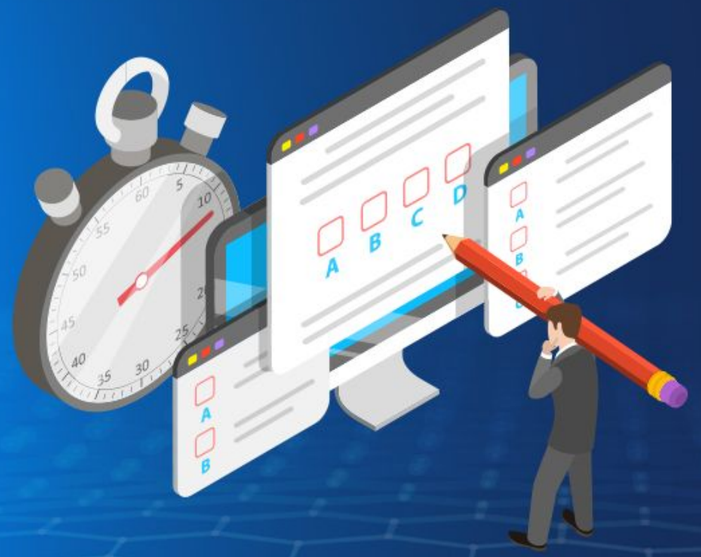
Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

Key Takeaways

You are now able to:

- Understand the need for NoSQL databases
- Analyze the HBase architecture and components
- Differentiate HBase from RDBMS





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 Regional Server

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 Regional Server

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

In which of the following scenarios can we use HBase?

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 of the above

Knowledge Check 2

In which of the following scenarios can we use HBase?

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 of 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 variable schema.

Lesson-End-Project

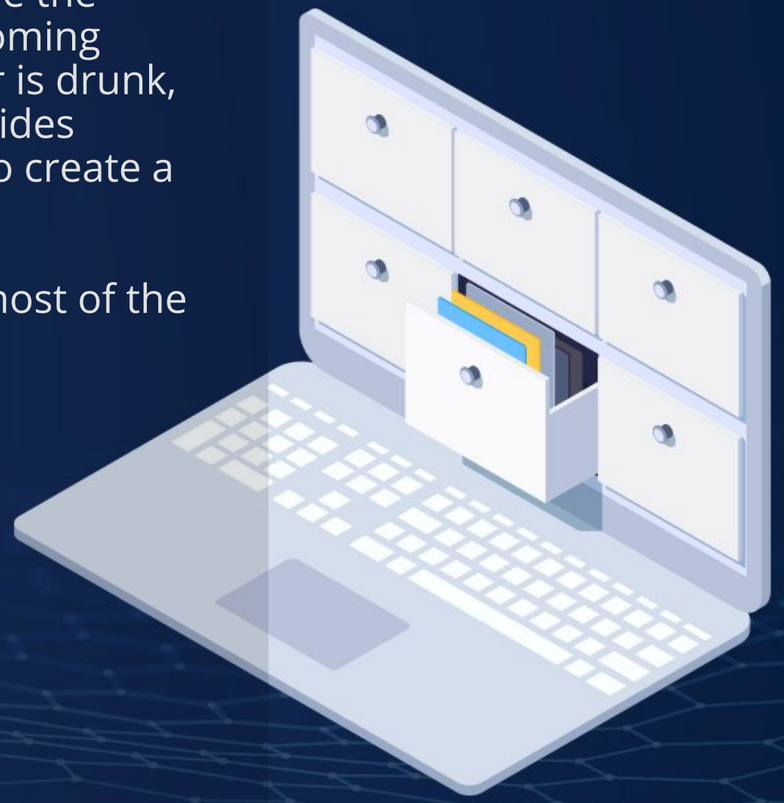
Problem Statement:

Global transport private limited is in transport analytics and they are keen to ensure the safety of people. Nowadays, as the population is increasing accidents are also becoming more and more frequent. Accidents occur mostly when the route is long, the driver is drunk, or the roads are damaged. The company collects data of all the accidents and provides important insights that can reduce the number of accidents. The company wants to create a public portal where anyone can see the accident's aggregated data.

Your task is to suggest a suitable database and design a schema which can cover most of the use cases.

You are given a file that contains details about the various parameter of accidents.
The column details are as follows:

1. Year
2. TYPE
3. 0-3 hrs. (Night)
4. 3-6 hrs. (Night)
5. 6-9 hrs (Day)
6. 9-12 hrs (Day)
7. 12-15 hrs (Day)
8. 15-18 hrs (Day)
9. 18-21 hrs (Night)
10. 21-24 hrs (Night)
11. Total



Lesson-End-Project

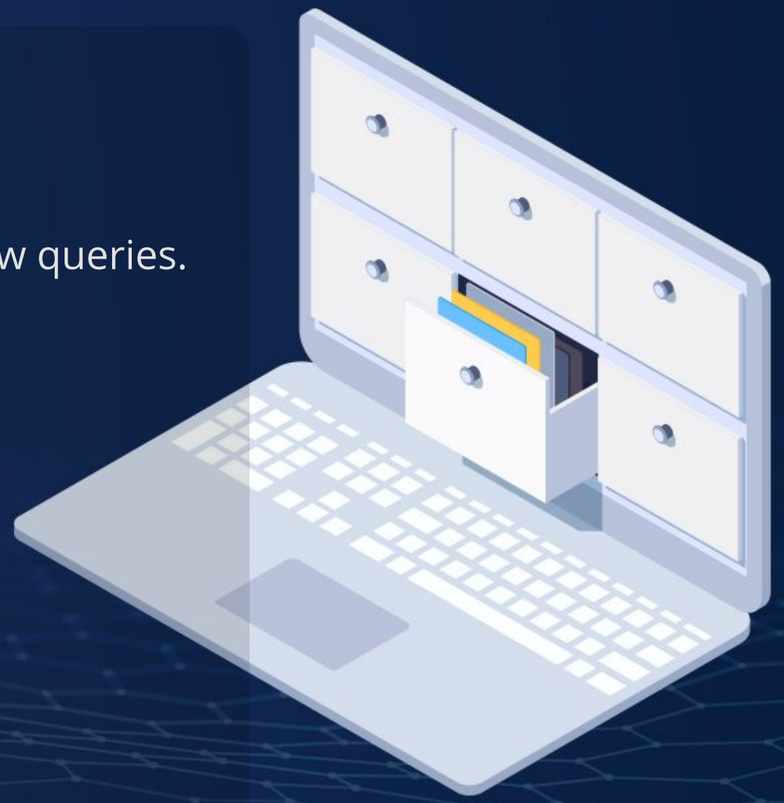
Problem Statement:

You have to save the given data in HBase in such a way that you can solve the below queries. Please mention what you are selecting as a row key and why.

1. Get the total number of accidents when you are given
   a. Year
   b. Type of Accident
   c. Time Duration

2. Get the total number of accidents when you are given
   a. Year
   b. Type of Accident

3. Get the total number of accidents in a given year



Thank You