| Aptech Limited |

Big Data Hadoop and Spark Developer

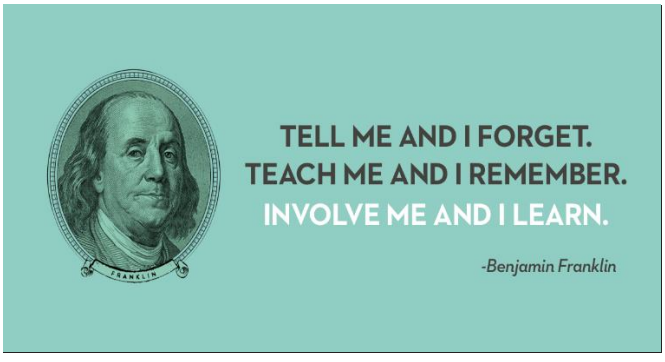
FACILITATOR GUIDE

Aptech®
Unleash your potential

About the Facilitator Guide

This guide is designed for the facilitator/faculty to enable the learning objectives. The main goal is to help the participants meet the learning goals through extensive guidance and excellent facilitation in a blended learning mode.

This guide explains in detail the process of conducting the training, mode of delivery at each content point, additional support required, and activities to be carried out to ensure effective and smooth learning.



Goal

This guide is designed extensively to help the facilitator engage students in learning about Big Data Hadoop and Spark. It aims at providing information about the subject among students. This program intends to teach the concepts of Hadoop with Big Data and Spark, and their applications in organizations and enable students to become a Big Data Hadoop and Spark Developer.

Context

Hadoop framework can store data and run applications on clusters of inexpensive computer hardware. It includes the capacity to store huge volumes of data. It includes great processing power and can handle unlimited concurrent jobs. As per Forbes, the Hadoop market will reach almost $99 billion by 2022.

Big data processing lifecycle uses various components of the Hadoop ecosystem. Users can execute real-life, industry-based projects with an in-depth knowledge of the Big Data framework using Hadoop and Spark.

Course Structure

Facilitator Guide	Lesson Number	Start Topic	End Topic	Class Duration	Video Content Duration (Approx.)
1	1 and 2	Introduction	HDFS Architecture and Components	2 Hours	00:35:02
2	2	Block Replication Architecture	Key Takeaway	2 Hours	00:31:59
3	3	Introduction	Data Types in Hadoop	2 Hours	00:32:54
4	3 and 4	Joins in MapReduce	Key Takeaway	2 Hours	00:43:06
5	5	Introduction	Key Takeaway	2 Hours	00:28:36
6	6 and 7	Introduction	Key Takeaway	2 Hours	00:31:35
7	8	Introduction	Key Takeaway	2 Hours	00:28:06
8	9	Introduction	Key Takeaway	2 Hours	00:18:08
9	10	Basics of Apache Spark	Key Takeaway	2 Hours	00:39:54
10	11 and 12	Introduction	Key Takeaway	2 Hours	00:30:04
11	13 and 14	Introduction	Key Takeaway	2 Hours	00:23:03
12	15	Introduction	Key Takeaway	2 Hours	00:25:54
13	16	Introduction	Key Takeaway	2 Hours	00:18:49

Course Objectives

After completion of this course, student will be able to:

- Master the concepts of Hadoop framework and its deployment in a cluster environment
- Understand how the components of Hadoop ecosystem such as Hadoop 2.7, Yarn, MapReduce, HDFS, Pig, Hive, Impala, and so on fits in with data processing lifecycle
- Describe how to ingest data using Sqoop and Flume
- Explain the process of distributed data using Spark
- Learn about Spark SQL, Graphx, and Mlib
- List the best practices for data storage
- Explain how to model structured data as tables with Impala and Hive

Audience

This course is beneficial for:

- Software Developers and Architects
- Analytics Professionals
- Senior IT professionals
- Testing and Mainframe Professionals
- Data Management Professionals
- Business Intelligence Professionals
- Project Managers
- Aspiring Data Scientists
- Graduates looking to build a career in Big Data Analytics

Pre-requisite for Students

Basic understanding of Core Java and SQL is essential for this course.

Session Preparation Guidelines

1. Using Video Content

Each session of this course is delivered in a blended learning mode. The delivery will be structured around the video content. The faculty/facilitator will ensure effective learning, through various value additions that will be provided in the classroom along with the video content. For each session, detailed delivery guidelines are provided in this guide.

2. Session Material

Following are the basic materials required for a classroom session:

- Whiteboard and Markers
- Be ready with Flip charts or any other material that is specified in each session
- Printed copies of **Participant Worksheets** (if any)
- Projector and pen drive with the video file to be played
- **Check List** (Annexure 1): This checklist is to be used and signed before every session to ensure that everything is in place
- **Self-Assessment Sheet** (Annexure 2): Use this sheet at the end of every session to self-assess the quality of facilitation/teaching

To be printed for each Session:

Annexure 1

Check List

Week Before the Session	Day of the Session
✓ Gone through the session details in this guide. ✓ Gone through the links (if any) mentioned in session details. ✓ Collected supplementary material required as mentioned in session details. ✓ Taken prints of the worksheets (if any) for the students. ✓ Ensured all material required for hands on is in place. ✓ Checked and played the video twice and rehearsed timings.	✓ Projector is working. ✓ Markers are working. ✓ All material is in place (Flip charts, Notes, and so on). ✓ Attendance sheet is printed. ✓ Self-Assessment sheet is printed. ✓ Video is playing with proper audio.

Session Name:
Signature:
Date:

Facilitator Notes:

Self-Assessment Sheet

Session Name:
Duration:
Date:

Assessment Pointers	Assess Yourself
- Was the Quality of Video Played (Audio and visual) OK?	
- Was the timing of the Session appropriate to complete the learning process?	
- Were students actively participating in discussions?	
- Were students able to complete the tasks assigned?	
- Were links added for value addition components of the content appropriate for students?	
- Were the additional explanations adequate for the students?	
- Were the questions asked by students demonstrating their understanding of topics?	
- Rate the quality of teaching and delivery.	

Session 1

Session Objectives:

Understand the concept of Big Data and its challenges
Explain Hadoop and how it addresses Big Data challenges
Describe the Hadoop ecosystem
Explain how Hadoop Distributed File System or HDFS stores data across a cluster
Explain how to use HDFS with the Hadoop User Experience or Hue File Browser and the HDFS commands
Illustrate the YARN architecture
List the different components of YARN
Explain how to use Hue, YARN Web UI, and the YARN command to monitor a cluster

Video Content: **Lesson 1 and 2**

INTRODUCTION TO BIG DATA AND HADOOP ECOSYSTEM

CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 1.1

FACILITATION GUIDELINES

Greeting: Welcome students to the course. Introduce yourself and seek introduction of students. During introduction, ask students what they expect from this course.
After the introduction, play the course video.

Play Video Content: Lesson 1.1

Content: After playing the video, initiate a discussion on value of this course to professionals. Explain that this course is beneficial for all IT professionals, be it analyst, developer, or data architect. Reinforce that this course is especially relevant for professionals who manage big data. Reiterate each objective listed on the screen (00:35).

Ask students if they can define big data.

Say: Let us now see what big data and Hadoop are.

OVERVIEW TO BIG DATA AND HADOOP

CLASS DURATION: 22 MINUTES
VIDEO DURATION: LESSON 1.2
FACILITATION GUIDELINES

Play Video Content: Lesson 1.2

Content: Big data refers to huge volume of data. This data cannot be stored and processed by traditional data-processing application software within a given period. It is a misconception when people say big data is referred as data that is in Tera Bytes (TBs) or Peta Bytes (PBs), or greater than these.

The term big data is nowadays widely used. It refers to data that is high in size and cannot be captured, managed, or processed by RDBMS effectively. Big data may be high in volume, velocity, or variety. Major sources of big data are social media platforms, mobile apps, Artificial Intelligence (AI) applications, and so on. All these sources generate highly complex and varied data with high volumes. Analysis of big data is important as it allows a business to make better and effective decisions. To process and analyze big data sets, one needs to look beyond traditional mechanisms.

Reference: Use this link to explain an analogy for big data:
http://www.tomorrowtodayglobal.com/2013/08/27/an-analogy-to-explain-big-data-turning-hay-into-needles/
Now, let us focus on social networking sites such as YouTube, Facebook, and Twitter. Ask students if they are aware how much of data is processed each day. Tell them that Facebook receives approximately

100 TB of data each day. Twitter processes 400 million tweets each day. Similarly, other popular networking sites receive tons of TBs of data each day. Imagine the amount of data that is stored and processed on these sites. Note that number of users using these sites is also growing. Hence, storing and processing this data is challenging. This data stores valuable information that needs to be processed in a short time.

Ask: Ask students if such data can be processed within a given period? Are computing resources of a traditional system sufficient? Allow students to respond.

Say: It is now clear that traditional systems cannot or process handle big data. What has come to the rescue is a distributed computing and processing system. What does this mean? It is a type of computing and processing system or framework that uses multiple machines to do a task. Such a system includes multiple computers, but runs as a single system. Computers share different resources and capabilities and provide high performance by completing portions of overall tasks. If one of the computers is down, work will not stop as there are other computers to take care of the task. In addition, the system is horizontally scalable. This means the system can be expanded by adding more machines.

However, a distributed system has certain challenges. Security is a challenge as computers are connected through local network or Wide Area Network. It must be ensured that secure networks are used and various resources of the system must be protected from unauthorized access. Unlike a centralized system, it is difficult to troubleshoot a distributed system due to distribution across multiple servers. Bandwidth is also an issue when it comes to a distributed system as resources are shared. Processing huge data can be a challenging task. Another disadvantage is high programming complexity. It has to be ensured that programs written by different developers follow common standards; otherwise, these programs will be unable to communicate with one another.

Hadoop overcomes all these challenges posed by distributed systems. It is an open source distributed computing and processing framework developed by Apache Software Foundation. It was, primarily designed to make large scale computing more affordable. Hadoop can handle different types of data and huge volumes of data. It can handle various forms of structured and unstructured data. This gives users more flexibility to collect, process, and analyze data.

Hadoop works with clusters of commodity computers that are standard PCs. Computers in each cluster are connected to work as a single system. Each cluster includes a server. These are low-cost and low-performing servers.

However, when they run parallel across a shared cluster, they provide powerful computing capabilities. Hadoop includes following features:

- ***Flexibility***: Most organizations generate 70% of unstructured data. This data is not well-handled due to lack of technologies. Hadoop offers flexibility here as it can deal structured, semi-structured, and unstructured data.
- ***Fast Data Processing***: Unlike traditional systems, Hadoop processes large amount of data at a very high speed.
- ***Robust***: Hadoop has a very robust ecosystem. It includes many component and services to ingest, store, analyze, and maintain data ensuring high-availability of data.
- ***Cost Effective***: Hadoop does not depend on any proprietary technology. It uses commodity hardware as a node in a Hadoop cluster. This makes Hadoop a low-cost big data processing framework.
- ***Scalable***: Hadoop is highly scalable. New nodes can be easily added in the system if data processing demands without altering anything in the existing systems or programs.

  Use this analogy to explain scalability and scaling:

  Consider a real-world analogy to understand this well. Let us assume you are going on a holiday trip with your 20 of friends. The travel company sent only one bus with a capacity of 10 people. What will you do? You need two buses. You can ask for either two buses or one double-decker bus

that can carry all 20 friends in one bus. That means you are scaling the resources, which is the bus in this case.

In this example, if the travel agent sent double-decker bus, then it is vertical scaling as you increased the capacity of bus not the number of buses. However, if the travel agent sent two buses, then it is like horizontal scaling as you increased the number of resources and not the capacity of single bus.

- ***Fault Tolerant***: Hadoop uses replication to deal with fault tolerance. Data automatically is replicated at two other locations. Hence, even if one or two of the systems collapse, the file is still available on the third system.

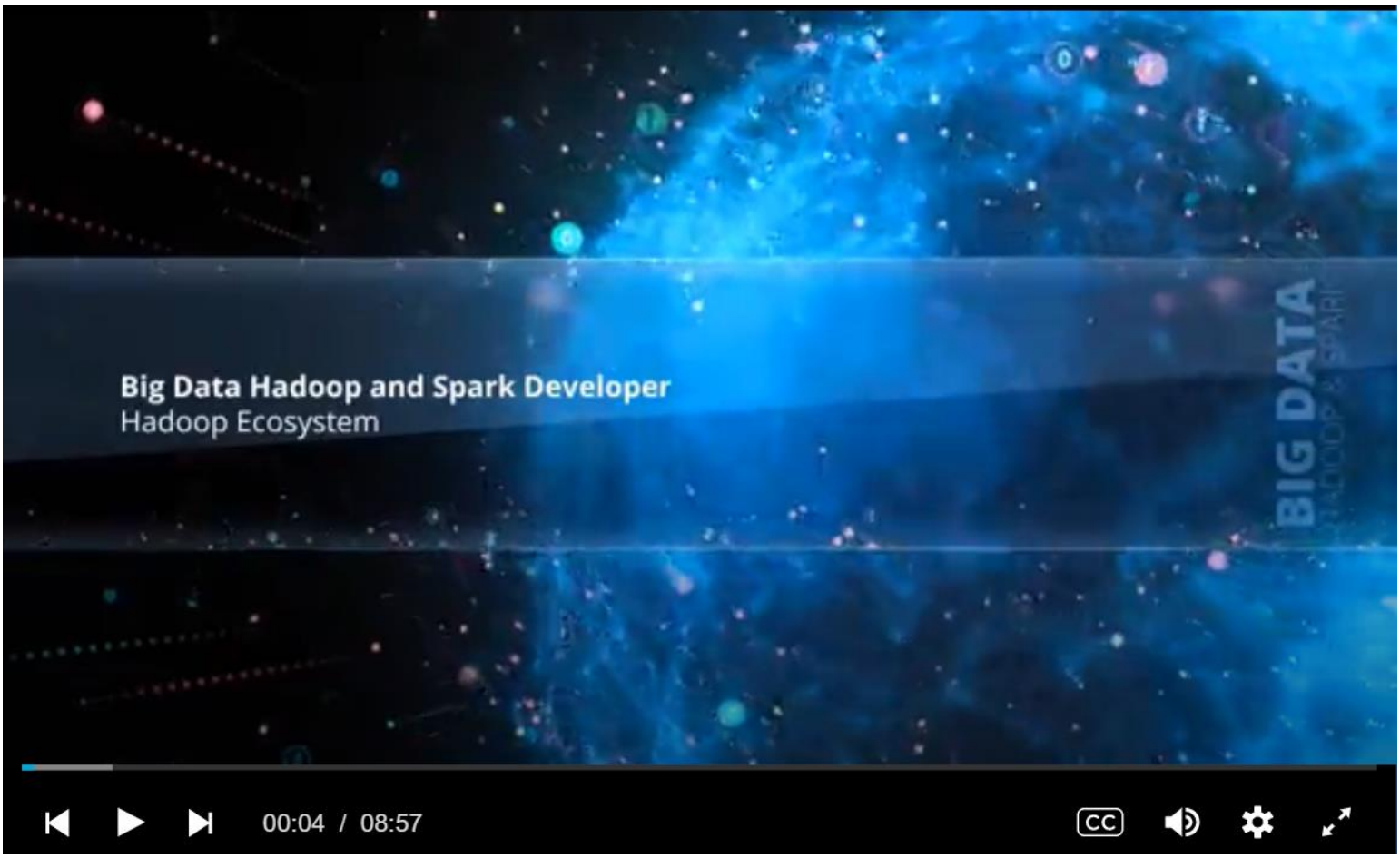
Resource: Ask students to view this image on their computers and zoom in to see an enlarged view:

http://mattturck.com/wp-content/uploads/2017/05/Matt-Turck-FirstMark-2017-Big-Data-Landscape.png

ACTIVITY

Ask students to perform some research and determine a brief history of Big Data technologies. They should present their results in brief on a sheet of paper.

HADOOP ECOSYSTEM



CLASS DURATION: 22 MINUTES
VIDEO DURATION: LESSON 1.3
FACILITATION GUIDELINES
Play Video Content: Lesson 1.3

Say: By now, you all know that Apache Hadoop is an open source framework. Let us now see the components of a Hadoop ecosystem.

Content:
Explain using the following explanation for each component:

Hadoop Distributed File System (HDFS):

It is the most important component of the Hadoop Ecosystem that runs on commodity hardware. HDFS is the primary storage system of Hadoop and is a Java-based file system. It is this component that provides scalability, fault tolerance, and reliability. HDFS has the capability to handle different types of data sets, such as structured and unstructured data. Since HDFS is a distributed file system, it helps store data across various nodes and maintains the log file about the stored data, which is the metadata.

HDFS includes two core components, NameNode and DataNode. **NameNode** is the master node. It stores the metadata, such as which DataNode has what data about a particular file. It consists of files and directories.

All data is stored on **DataNodes**. A DataNode is a slave node. Therefore, they require more storage resources. DataNodes are commodity hardware, such as your laptops.

The fault tolerant feature of HDFS provides multiple copies of a DataNode. Consider there are five DataNodes and a file is split in these nodes. Then, there will be three different copies across these nodes. By default, the replication factor is three. The files are read and written in parallel in this model for faster data processing.

HBase

HBase also follows master slave architecture. Unlike HDFS, there can be multiple masters in HBase. This reduces single point failure. However, only one master will be active at a time HBase stores the structured data in a tabular format of columns and rows. It gives access to the real-time data to read or write on HDFS.

There are two components in HBase: HBase Master and RegionServer. **HBase Master** performs load balancing across all RegionServers. It is an interface for creating, updating, and deleting tables. It also handles Data Definition Language (DDL) operations, such as creating and modifying the structure of database objects in database. **RegionServer** is similar to DataNode of the HDFS architecture. It handles read, write, update, and delete requests from clients. Note that the RegionServer process runs on every node in Hadoop cluster.

Sqoop

Sqoop imports and exports data. It imports data from external sources, such as Oracle and MySQL, into Hadoop ecosystem components, such as HDFS and HBase. It exports data from Hadoop to other external sources.

Flume Data ingestion is the process of importing, transferring, or loading data into a persistent storage layer (HDFS). This enables data storage and facilitates usage of data either immediately or later. Flume is a service that helps in ingesting unstructured and semi-structured data into HDFS.

Flume collects log data from many sources. It then aggregates and writes it to HDFS. Flume is generally used for data sources such as application logs, sensor/machine data, social media feeds such as Facebook and Twitter; e-commerce sites such as Amazon and so on; geo-location data where large volume of data is generated in real time.

Using Flume, you can get data from multiple servers immediately into Hadoop.

Spark
Spark is an open source cluster-computing framework. It does real-time data processing and batch processing with huge data. It is 10x faster than Hadoop.

Hadoop MapReduce
It is the core component of processing in a Hadoop Ecosystem. It provides the logic of processing. It performs large-scale data analysis using multiple machines in the cluster. It takes a job from the user, divides it into smaller tasks. Next, it assigns them to the worker nodes.

In other words, it processes large data sets using distributed and parallel algorithms in a Hadoop environment.

A MapReduce program includes two functions: Map and Reduce. The Map function converts data into keys and values called tuples. Based on this key-value pair, the data is grouped, sorted, and processed. Based on this key-value pair, data is assigned to the nodes for processing.

The Reduce function takes data from the Map function as an input. It further breaks the data into smaller sized datasets. It formats datasets based on key and value. Key is the record identifier and value is what the key identifies. The final output is written into HDFS.

Pig

Now programming Map and Reduce applications is very time consuming. Apache Pig changes this. While programming Map and Reduce applications takes 100 lines of code statements, Pig programming takes only 10 lines of code statements.

Pig provides a SQL-like interface for Hadoop applications. It is used to analyze and query large data sets that are stored in HDFS. Pig is made up of language, Pig Latin, and runtime environment. Pig Latin programs are executed in this environment. The programs result as MapReduce jobs that run on a cluster.

Pig has the capability to execute multiple queries at the same time.

Impala

It is a database management system for data stored in Hadoop cluster. It is a query engine that runs on Apache Hadoop. It helps users to issue low-latency SQL queries without requiring data movement or transformation. Impala provides the fastest way to access data that is stored in HDFS.
Hive: It was difficult for organizations with traditional database warehouses to interact with Hadoop as developers used SQL queries to extract data. Apache Hive was developed to resolve this issue. Hive provides SQL intellect so that developers can write SQL-like queries like HQL or Hive query language to extract data from Hadoop. Hive converts SQL queries to MapReduce queries. This is how it talks to HDFS file system or Hadoop ecosystem.

Cloudera Search

It provides advanced search capability for data in Hadoop cluster.

Oozie

At times, several jobs need to be performed to execute a task. Oozie, an open source Java Web application, takes care of such tasks. It ensures that the jobs are run in sequential order to achieve the desired output. It acts as a job coordinator and simplifies workflow. Oozie is a tool that schedules Apache Hadoop jobs.

There are two types of Oozie jobs: Oozie workflow and Oozie coordinator. Oozie workflow stores and runs workflows composed of Hadoop jobs, such as MapReduce and Hive. Oozie coordinator runs workflow jobs based on predefined schedules and data availability.

Hue

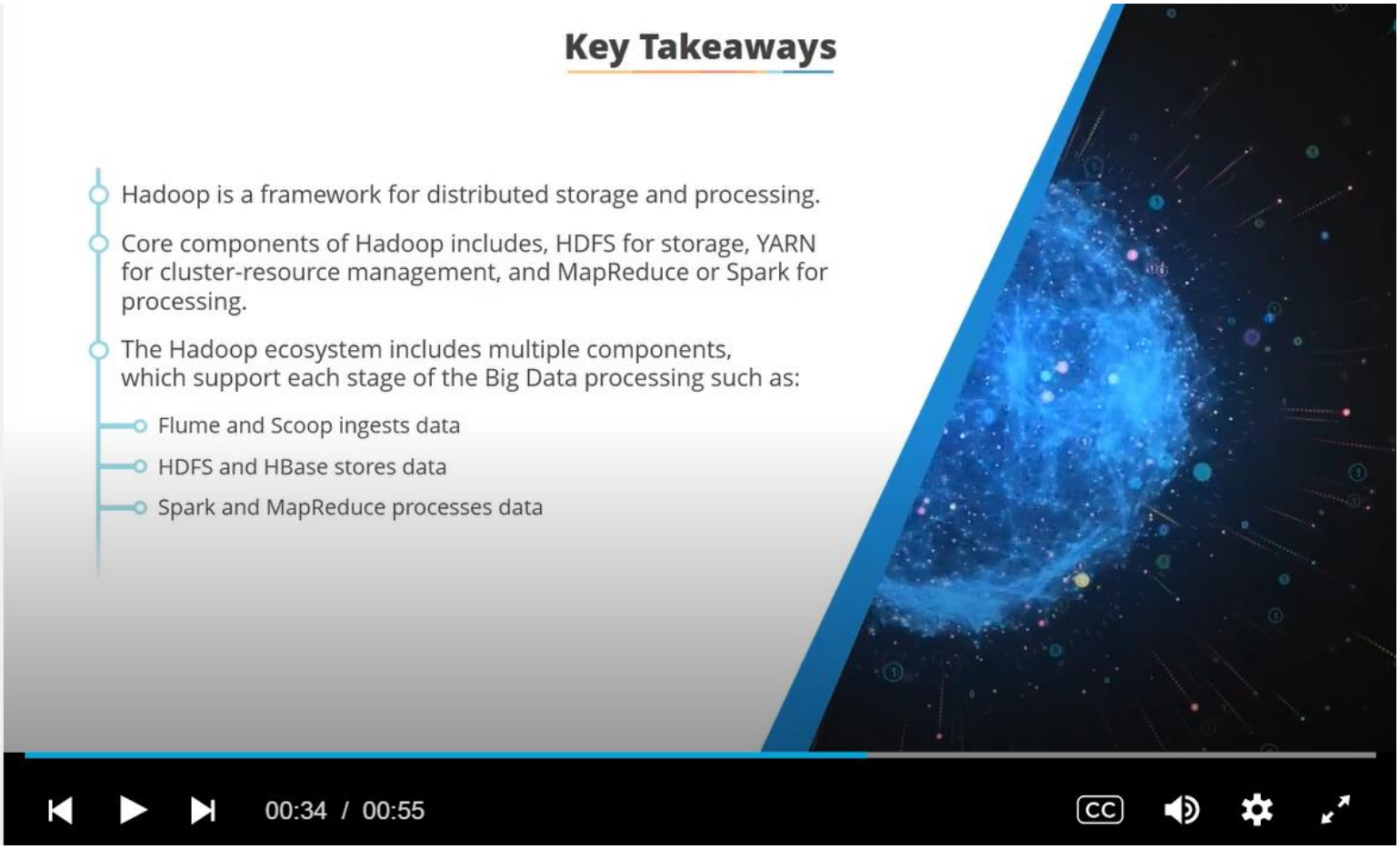
Hue is an open source project. It provides a Web user interface to interact with HDFS and MapReduce applications. There is no need to use command line interface to access the Hadoop ecosystem.

ACTIVITY

CLASS DURATION: 12 MINUTES

Identify and discuss how Facebook uses big data.

KET TAKEAWAY



CLASS DURATION: 5 MINUTES
VIDEO DURATION: LESSON 1.4
FACILITATION GUIDELINES
Play Video Content: Lesson 1.4

Summarize topics covered in the lesson as listed on screen. Rewrite them on the whiteboard. Tell students that in the session, they have learnt about big data and Hadoop Ecosystem in brief.
Say: Let us have a small quiz to check your understanding.

QUIZ

CLASS DURATION: 10 MINUTES
VIDEO DURATION: LESSON 1.5
FACILITATION GUIDELINES
Play Video Content: Lesson 1.5 (Quiz).

Content: Show each question and give students some time to recall the concepts and attempt the question. Once they finish answering the question, continue the video, and show the correct answer and explanation. Repeat this process for all questions in the Quiz. If students have queries regarding the questions, discuss, and resolve the queries.

You may choose to ask few more questions as given here:

Questions:
1. What is a distributed system?
Answer: In a distributed system, multiple computers run as a single system. Computers share different resources and capabilities.

2. What is Hadoop?
Answer: It is an open source distributed processing framework. Hadoop can handle various forms of structured and unstructured data. This gives users more flexibility to collect, process, and analyze data.

3. What is HDFS?
Answer: Hadoop Distributed File System (HDFS) is the most important component of the Hadoop Ecosystem. It is the primary storage system of Hadoop.

4. What stores structured data in a tabular format of columns and rows?
Answer: HBase

5. What is the component that imports and exports data?
Answer: Sqoop

6. What is data ingestion?
Answer: Data ingestion is the process of importing, transferring, or loading data into a persistent storage layer (HDFS).

Say: Let us now move on to Lesson 2.

INTRODUCTION

CLASS DURATION: 12 MINUTES
VIDEO DURATION: LESSON 2.1
FACILITATION GUIDELINES
Play Video Content: Lesson 2.1 till 00.58

Content: Reiterate each objective listed on the screen.

Play Video Content: Continue Lesson 2.1

Say: Let us take an analogy to understand HDFS. Consider a library. Here, a librarian is analogous to a master in HDFS. Shelves store books. These shelves act as data nodes. Consider books as actual data. The person who helps the librarian is analogous to secondary name node. The librarian (Master) has all the information of books, that is, which book is in which shelf and in which compartment; a book is to be placed and so on. If a customer comes to the library, the librarian accepts request of customer. As output, the librarian tells exactly where that book is placed (that is location of the book). This is exactly what the master does in HDFS architecture. It stores metadata of the complete system.

HDFS is highly scalable as we can add/remove devices. Hadoop with its distributed processing and distributed storage architecture processes huge amounts of data with high speed. It divides input data file into a number of blocks and stores data in these blocks over several nodes.

Many big companies, such as Royal Bank of Scotland, British Airways, Vodafone, EBay, and so on have successfully implemented Hadoop in the production environment.

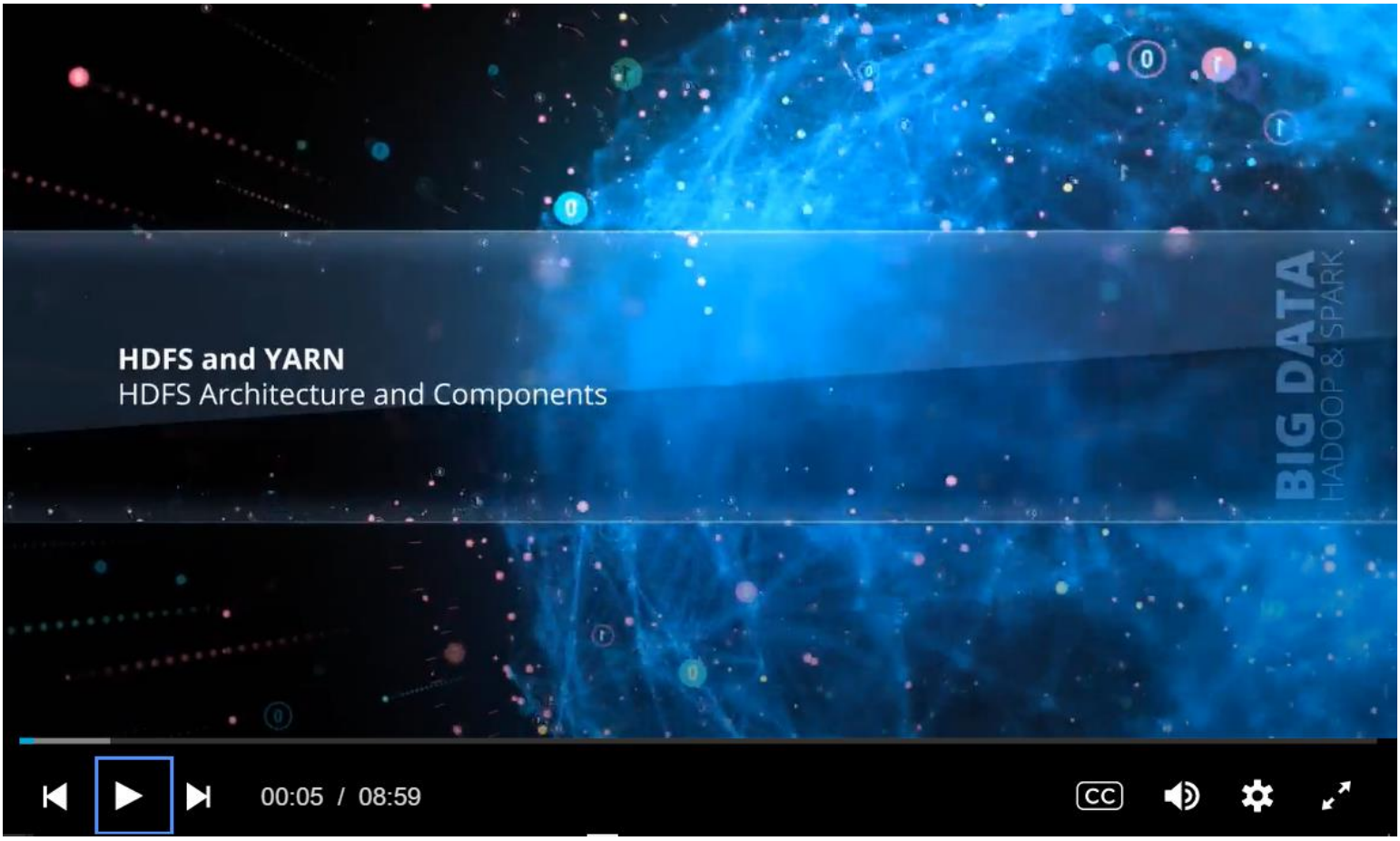
When a file is to be written in HDFS, it is broken into small pieces of data known as blocks. HDFS has a default block size of 128 MB. This size can be increased as per requirements. These blocks are stored in

the cluster in distributed manner on different nodes. This provides a mechanism to process data in parallel to the cluster.

Multiple copies of each block are stored across the cluster on different nodes. This is replication of data. By default, HDFS replication factor is three.

Say: Let us move to our next topic.

HDFS ARCHITECTURE AND COMPONENTS



CLASS DURATION: 12 MINUTES

VIDEO DURATION: LESSON 2.2

FACILITATION GUIDELINES

Play Video Content: Lesson 2.2

Content: Hadoop has two logical layers, storage layer and processing layer. Storage layer includes name node, secondary name node, and data node. Processing Layer includes node manager and resource manager. Name node is the centrepiece of HDFS. It is called the master node in Hadoop. Name node only stores the metadata of HDFS. Name node is a single point failure in HDFS. If name node fails, entire HDFS file system is lost. Therefore, in order to overcome this, secondary name node is included in Hadoop. Its main function is to store a copy of metadata that is FSImage and edit logs. Data node is responsible for storing actual data in HDFS. It is also known as slave node.

Name node and data node are in constant communication. Node manager is responsible for launching and managing containers in a node. Resource manager is a dedicated scheduler that assigns resources to requesting applications.

Say: With this, we end our session.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

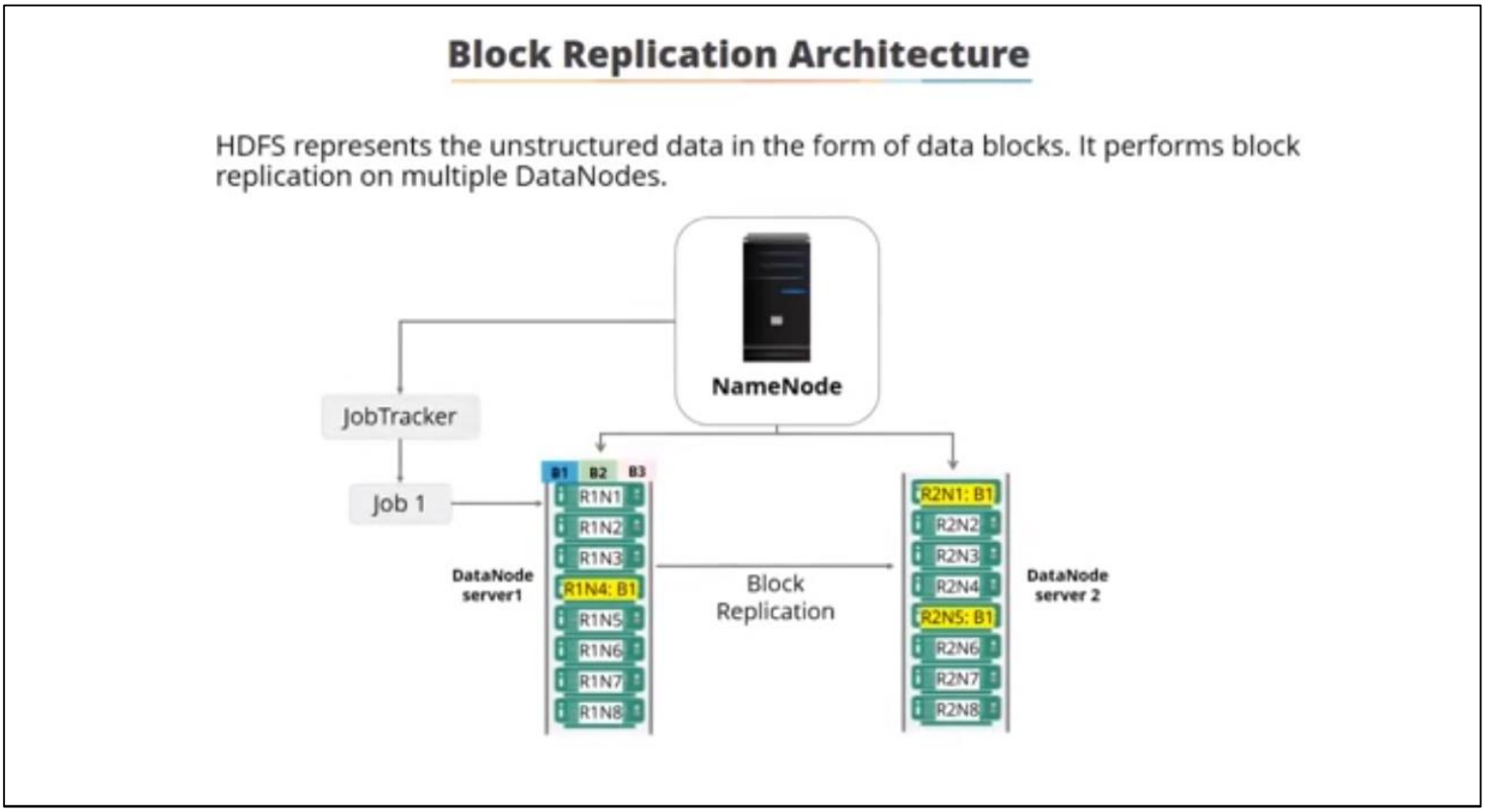
FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1: Prepare a document on HDFS architecture. Illustrate with graphics.

Session 2



Session Objectives:

- Discuss Hadoop Distributed File System (HDFS)
- Explain HDFS architecture and components
- Describe YARN and its features
- Explain YARN architecture

Video Content: Lesson 2

QUICK RECAP

CLASS DURATION: 5 MINUTES

VIDEO DURATION: NONE

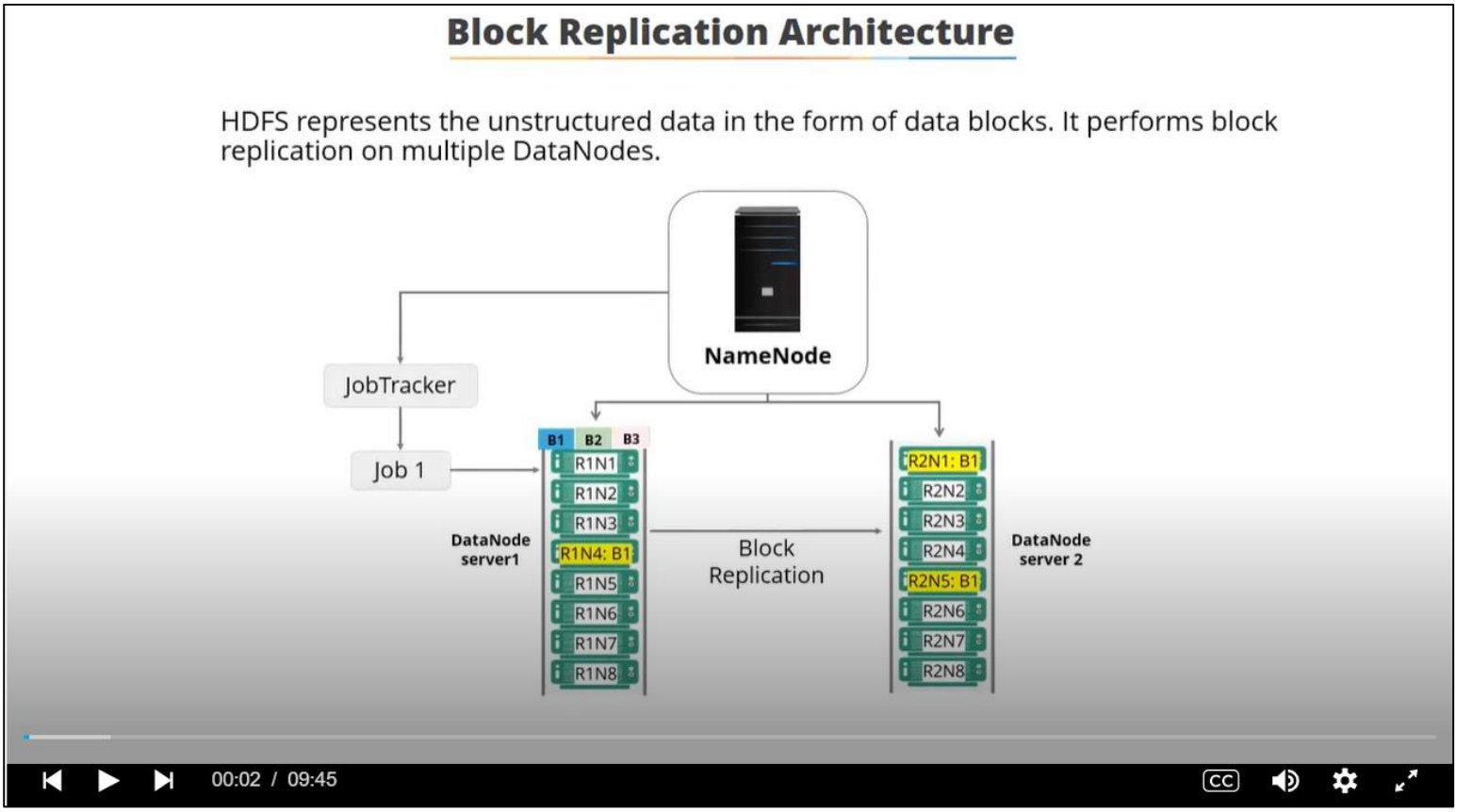
FACILITATION GUIDELINES

Greeting: Welcome the students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

Ask students about the assignment which was given to them in the last session.

Make random students stand and ask them one-two liners questions about previous session. Then start with the new session.

BLOCK REPLICATION ARCHITECTURE



CLASS DURATION: 15 MINUTES

VIDEO DURATION: LESSON 2.3

FACILITATION GUIDELINES

Play Video Content: Lesson 2.3

Content: By now, you know that HDFS is designed to handle big data. HDFS breaks a large size file into various blocks depending on the size of the file. HDFS writes the blocks of the file on multiple nodes.

Ask: Can anyone tell the advantages in this type of operation? (Allow students to respond.)

Say: This has the advantage of the distributed memory and processor load.

When a file is written, it is saved to HDFS into one or more blocks. Typically, these blocks are of 128 MB in size. HDFS copies the blocks to various other nodes in the cluster. Every block in a cluster will have three replicated copies. This provides data resilience and faster processing. Note that where blocks are placed depends on applications and various other factors in a cluster setting and block replication mechanism.

The default replication count is three. It can be extended beyond three. As the blocks are copied in their entirety, the data is ready for use in processes. Consider if a process is running and a node fails,

the process can be restarted on another node. The best part is that this resilient architecture is built using commodity computers.

Reference: Refer to following links to learn more about HDFS:
https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
https://www.coursera.org/lecture/hadoop/read-write-processes-in-hdfs-A2xLb

Resources: Ask students to refer to the following videos:

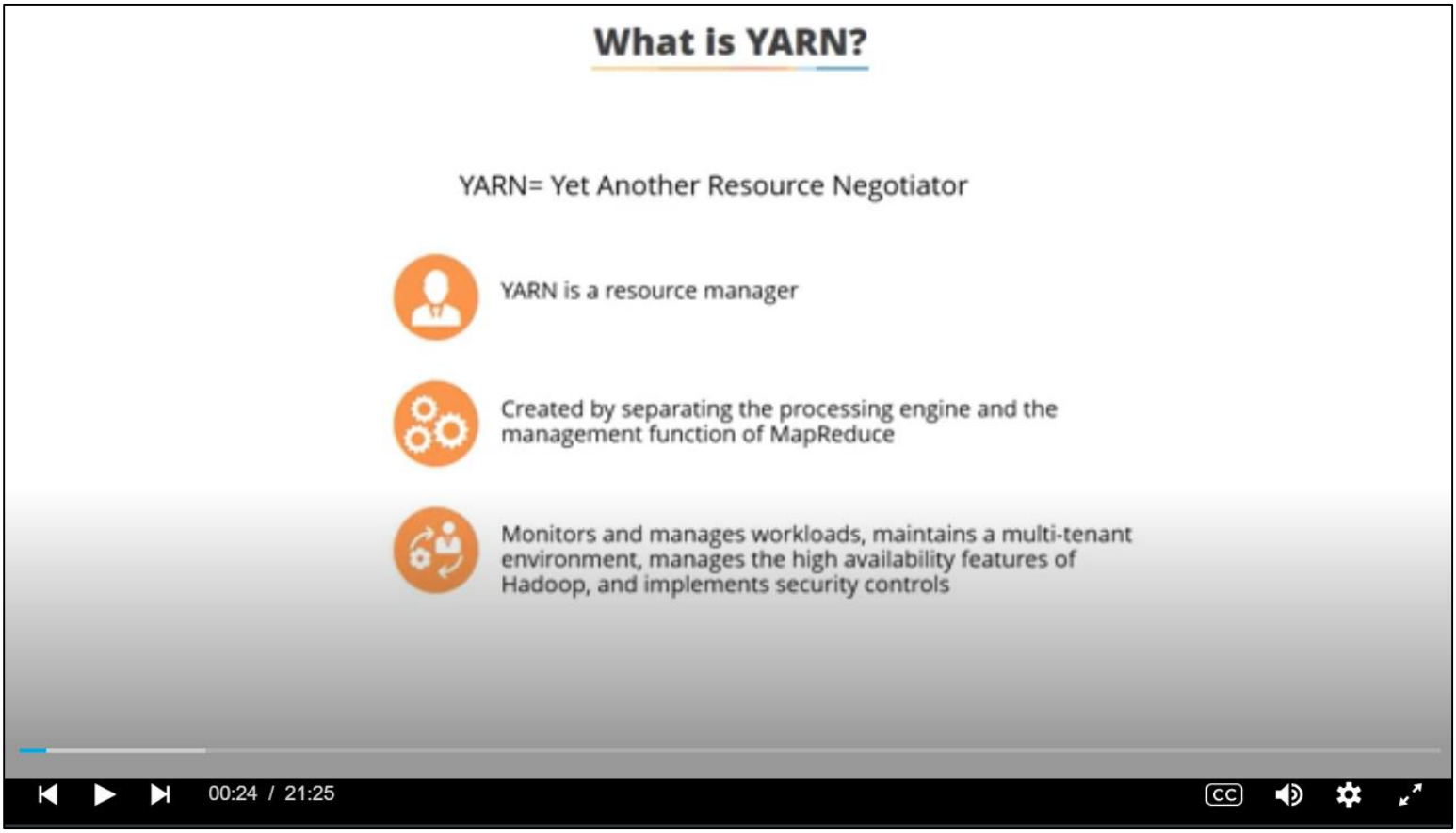
https://www.youtube.com/watch?v=CwoYV9EdCi0
https://www.youtube.com/watch?v=bIFDsfR3k4M
https://www.aosabook.org/en/hdfs.html

ACTIVITY 1

CLASS DURATION: 20 MINUTES

Illustrate the block replication architecture using a diagram. Name all the components of the architecture.

YARN INTRODUCTION



CLASS DURATION: 30 MINUTES

VIDEO DURATION: LESSION 2.4

FACILITATION GUIDELINES

Play Video Content: Lesson 2.4

Content: Consider Yarn as the operating system of Hadoop. It controls resources.

In earlier versions, Hadoop passed data to MapReduce engine. It then executed corresponding MapReduce programs. However, it was then found that MapReduce was not enough. There were several limitations. One such limitation is that MapReduce could not perform real-time analysis as MapReduce processes data in batches. Another limitation was increased latency. That is, MapReduce used to take a lot of time to process huge data because the processes were broken into two phases, namely, Map and Reduce.

You can consider Yarn as a middle layer between HDFS and execution engines. It manages cluster resources, such as memory and network bandwidth. Execution engines executes their programs, such as MapReduce engine executes map reduce programs.

Yarn provides APIs to request and work with Hadoop cluster resources. The goal of Yarn is to bring all distributed computational capabilities into a cluster.

Yarn includes two daemons (Daemon is a program that runs continuously at the background in the cluster, resource manager, and node manager). The resource manager is a master service. There is one active resource manager and one standby resource manager. If active resource manager goes down, the standby becomes active resource manager. There is one node manager for each node in the cluster. The node manager is a slave service. It launches and monitors containers. What is a container? It is an allocated resource such as memory on a single node to execute a user process.

Let us now see what happens when an application is submitted to Yarn. When an application is submitted to Yarn, the request goes to resource manager that locates a node manager. It then asks the node manager to launch a container. This is the first container of an application, which is called the application master. It is responsible for executing and monitoring the job. The application master's functionality depends on the application framework.

Resource: Ask students to refer to the following links for more information:

http://geekdirt.com/blog/introduction-and-working-of-yarn/
https://www.youtube.com/watch?v=nmaA5_d4E8c

Resources: Ask students to refer to the following videos:

https://www.youtube.com/watch?v=GCFCohlmWgs
https://www.youtube.com/watch?v=RncoVN0l6dc
https://www.youtube.com/watch?v=HHv2pkIJjR0

ACTIVITY 2

CLASS DURATION: 15 MINUTES

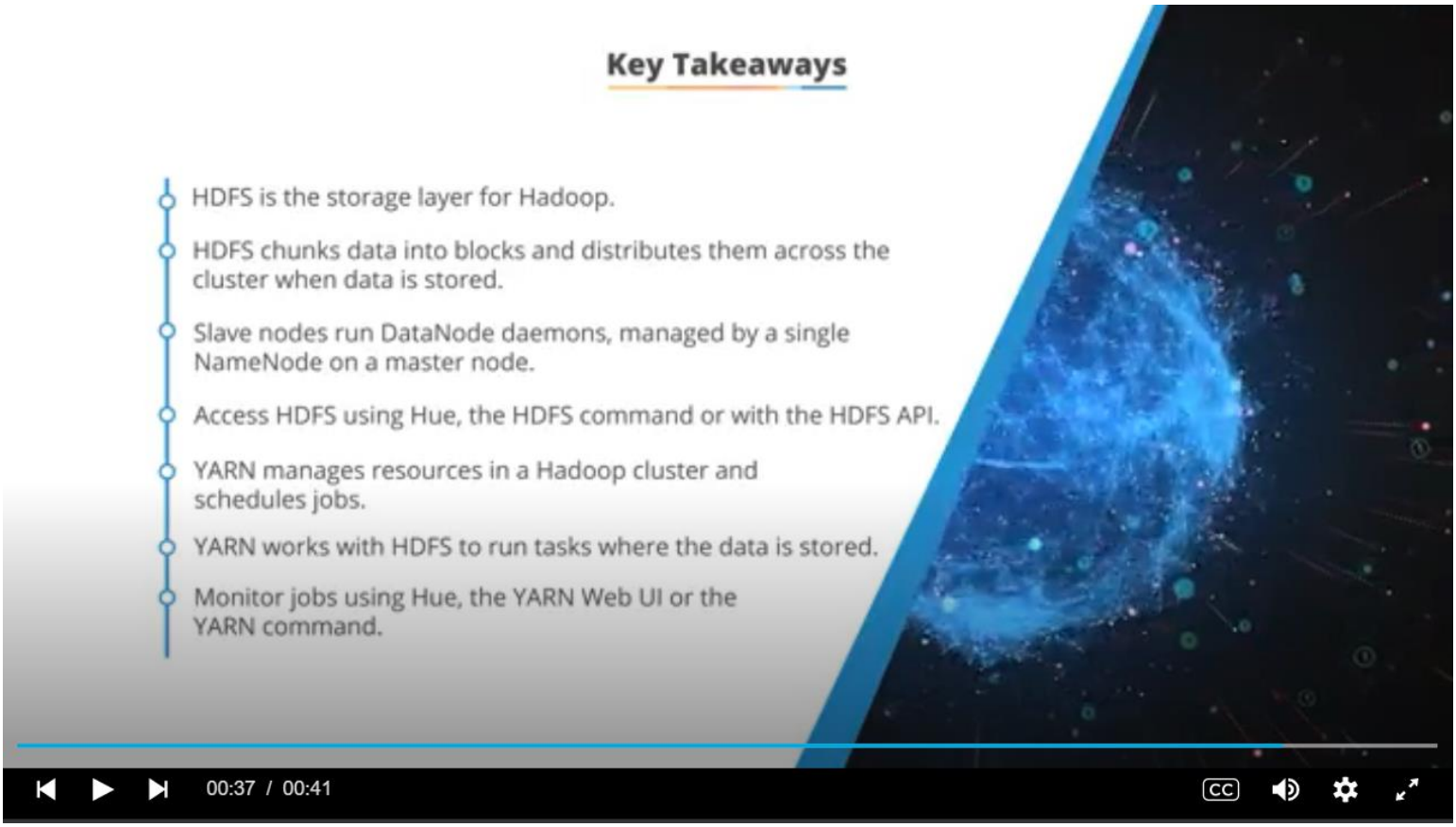
Research on limitations of MapReduce. Prepare a concise document that describes these limitations. Describe Yarn in your own words.

ACTIVITY 3

CLASS DURATION: 15 MINUTES

Divide the class into groups. Ask each group to identify and discuss how the application masters of the different execution engines work.

KEY TAKEAWAY



CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSION 2.5

FACILITATION GUIDELINES

Play Video Content: Lesson 2.5

Summarize topics covered in the lesson as listed on the screen. Rewrite them on the whiteboard. Tell the students in the session that they have learnt about block replication architecture and Yarn.

Say: Let us have a small quiz to check your understanding.

QUIZ

CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 2.6

FACILITATION GUIDELINES

Content: Lesson 2.6 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions.

Questions:

1. How does HDFS break large files?

Answer: HDFS breaks large files into various blocks.

2. What is the default count of replication in HDFS?

Answer: Three

3. How can you execute commands on HDFS?

Answer: FS shell

4. What allows multiple processing frameworks to be implemented in HDFS?

Answer: Yarn

5. What happens when an application is submitted to Yarn?

Answer: The request goes to resource manager that locates a node manager. It then asks the node manager to launch a container.

6. What are the two daemon services of Yarn?

Answer: Resource manager and Node manager

QUERIES REGARDING SESSION

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1: Execute the commands in HDFS for the following:

- Create a directory
- View the details of the current directory
- View if a file exists on a particular HDFS location
- Delete a directory and its contents

Session 3

Session Objectives:

- Define MapReduce
- Explain characteristics of MapReduce
- Understand advanced MapReduce
- Define Sqoop and its uses
- Analyze the process of importing and exporting data from Hadoop using Sqoop
- Understand basics of Sqoop 2

Video Content: Lesson 3

QUICK RECAP

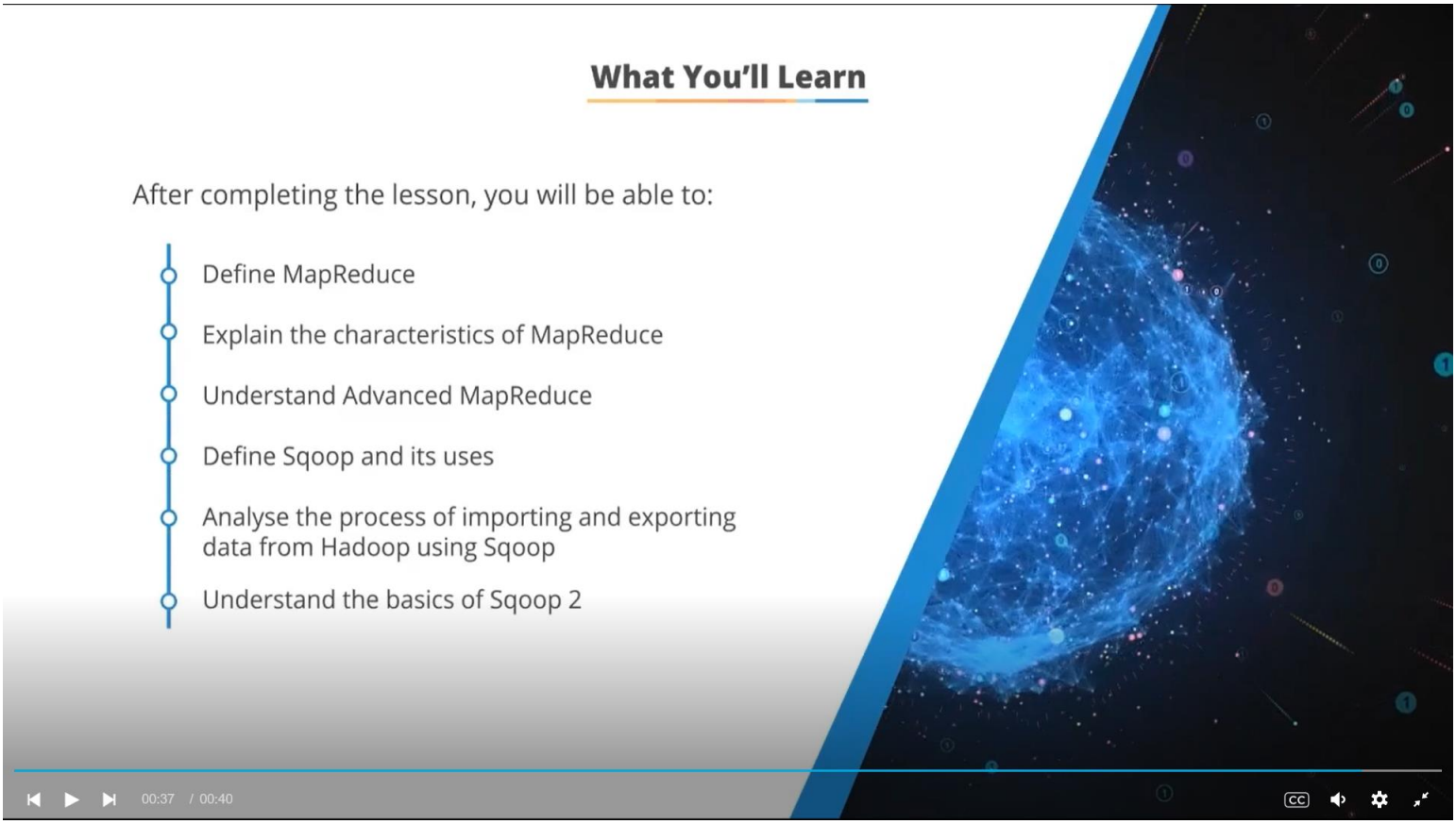
CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

INTRODUCTION



CLASS DURATION: 2 MINUTES

VIDEO DURATION: LESSON 3.1

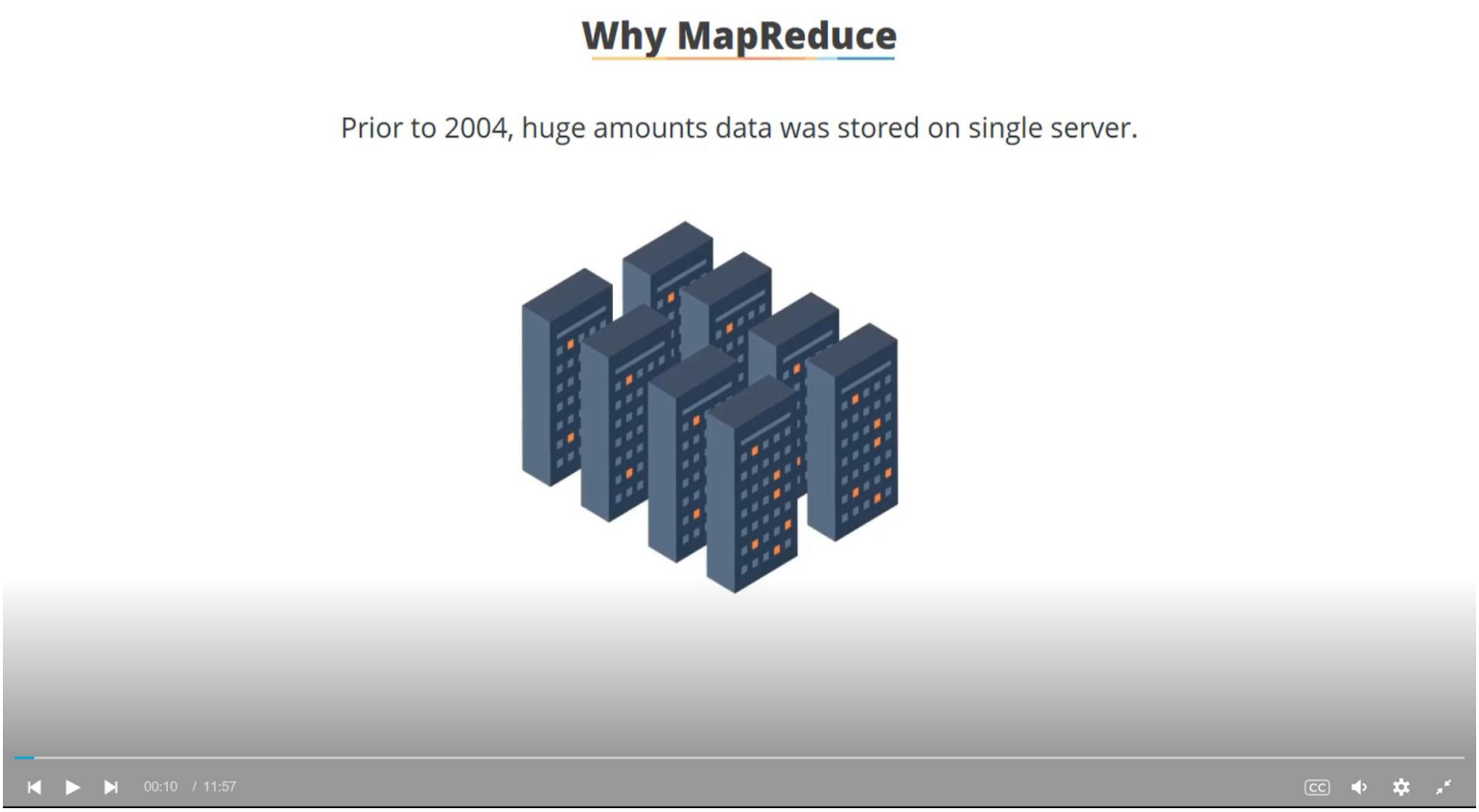
FACILITATION GUIDELINES

Play Video Content: Lesson 3.1

Content: After playing the video, repeat all the points and tell students that in this session, they will learn about MapReduce and Sqoop in the Hadoop Ecosystem.

Say: Let us begin!

WHY MAPREDUCE?



CLASS DURATION: 22 MINUTES

VIDEO DURATION: LESSON 3.2

FACILITATION GUIDELINES

Play Video Content: 3.2

Content: MapReduce is a programming framework for big data applications. Before MapReduce was introduced, it was a very tedious job to store huge amounts of data (for the most parts in terabytes). Also, processing the data was laborious. As a way to deal with this, some engineers in Google came up with a neat solution - MapReduce. In MapReduce, a large amount of data is divided into smaller chunks. These small chunks of data are processed upon as individual tasks. This is carried out using several inexpensive nodes simultaneously/parallelly, thereby saving the processing time. It then produces a listed output.

The MapReduce system is divided into two main phases: *Mapping* and *Reducing*. In the mapping part, the mapper takes in the input in the key-value format. The *key* references or points to a particular data, so it is a unique identifier whereas the *value* is the actual data. Every input will be processed in the mapping phase. In the second phase that is the reducer, the data is sorted as per the key. The data is then processed as per the requirement in the reducer and output is produced.

Apart from these two phases, shuffling/sorting is an intermediate stage. This stage takes in key-value pair as input, forms, and sorts the data using the key values.

A MapReduce job means a full MapReduce program. Here are certain important terms related to a MapReduce job.

- ***ApplicationMaster***: It is application specific. It is responsible for executing a single MapReduce job.
- ***ResourceManager***: Is the master. It is one per cluster. One of the tasks of the resource manager is to decide how to assign available resources.
- ***Cluster***: It is a set of host machines (nodes).
- ***NodeManager***: It is the slave. There are many of them per cluster. The node manager informs the status of resources available to the resource manager. This allows the resource manager to schedule the resources for a particular task.
- ***Containers***: Each container is a logical reservation of resources and allocates these resources to perform a specific task. The node managers are responsible for managing the containers and informing the resource managers the status of available resources.

First, a client requests for a particular task to be performed to the resource manager. The resource manager accepts the job and searches for a slave node that can be allocated to perform the task. The node manager allocates the available container that executes the application master. The application master requests for containers from resource scheduler. Once the required resources are obtained, the application master distributes the input to different node managers. In case of failure, it resubmits the task to alternate node managers.

Here are certain characteristic features of MapReduce. It handles large scale data. It allows data to be processed parallelly. It leverages commodity hardware and storage. The nodes being inexpensive, saves on costs too.

Ask the students if they can come up with any example where the MapReduce program can be used. (Allow the students to respond.)

Here are some of the examples you can give:

- Consider a railway booking application where a traveler needs to book tickets to go to a specific destination. The traveler checks for seat availability to travel on a specific date to a specific destination. Thus, the data needs to be sorted as per the destination, date, and seats available. As an output, the client/traveler would expect a result that matches all the needs.

  The application now needs to run through a large database. The MapReduce method can well fit this example. As per the request, a list of trains will be displayed that matches the client's search criteria.

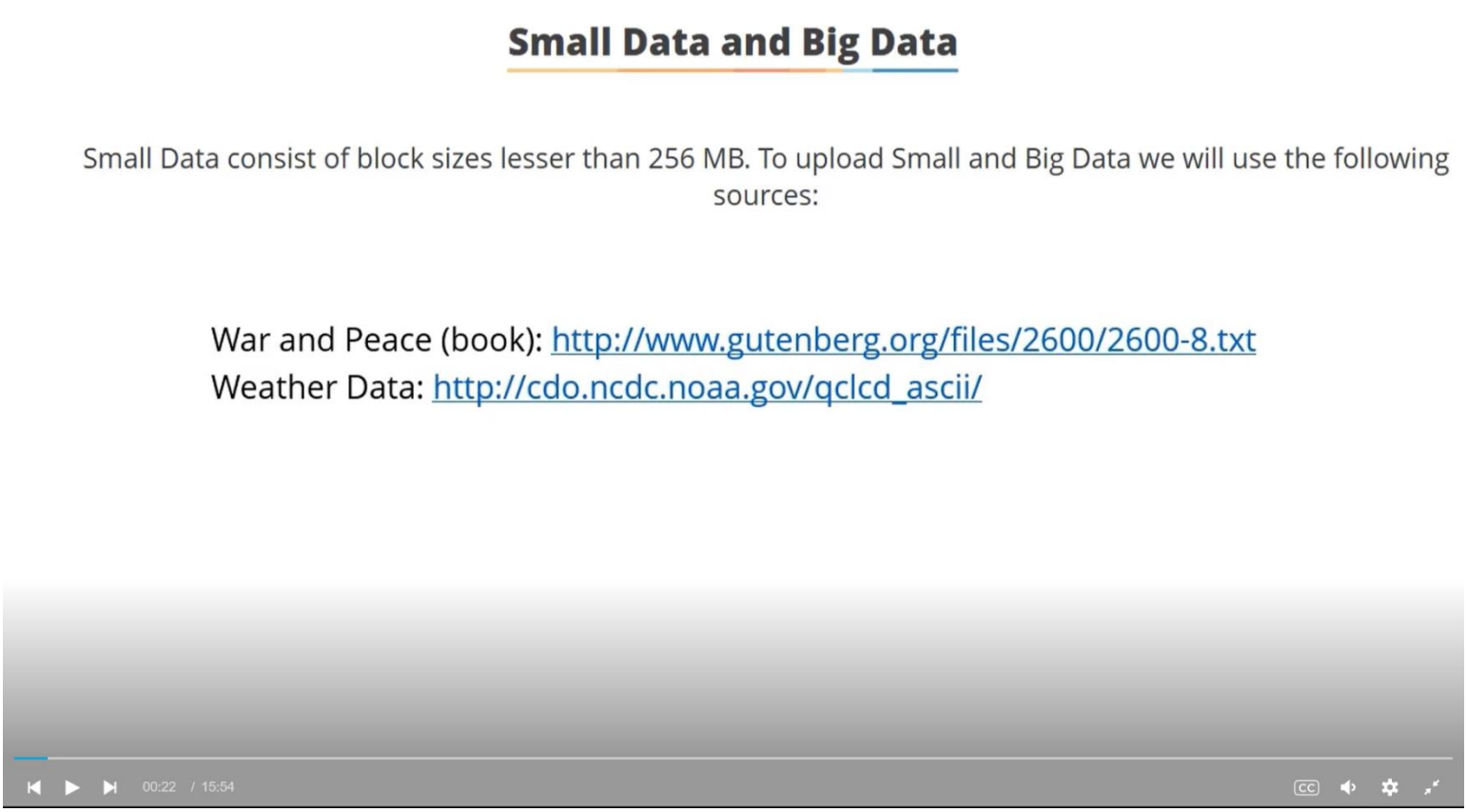
- Consider an online ecommerce retailer such as Amazon. Now say, it needs to calculate the rankings for bestselling gadgets (Example, mobile phones or a particular TV brand) of the previous year. This might take a lot of time if the whole of the data is processed in a serial fashion due to the huge amount of data that it has to consider. In this case, Amazon can adopt the MapReduce method.

Reference: Refer to the following links for more information:

http://hadoop.apache.org/docs/r2.9.1/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
https://www.ibm.com/analytics/hadoop/mapreduce
https://www.youtube.com/watch?v=PhdRyrmbRYQ
https://www.youtube.com/watch?v=QSzJSPtmsO8

Say: Let us now understand how to upload small and big data as well as the steps to build a MapReduce program.

SMALL AND BIG DATA



CLASS DURATION: 30 MINUTES

VIDEO DURATION: LESSON 3.3

FACILITATION GUIDELINES

Play Video Content: Lesson 3.3

Content: Both small and big data can be uploaded in Hadoop. When the data is uploaded in Hadoop, the MapReduce program performs data analysis by performing two main tasks, Map task, and Reduce task. However, in some cases, it is possible that only Map task is required and Reduce task is not required.

Here is what happens at the Map stage. The input data is processed and broken down into several chunks. Consider the input data (write on the whiteboard):

Tokyo New Jersey Sydney

Ohio Ohio Sydney

Tokyo Ohio New Jersey

This input data will be split across map nodes as follows: (Show on the whiteboard)

Tokyo New Jersey Sydney

Ohio Ohio Sydney

Tokyo Ohio New Jersey

Now, based on the logic that each data item will occur once, the mapper allocates the value of 1 to each data item. Consider this. (Show on the whiteboard)

Tokyo, 1
New Jersey, 1
Sydney, 1

Ohio, 1
Ohio, 1
Sydney, 1

Tokyo, 1
Ohio, 1
New Jersey, 1

See that the mapper generates a key-value pair. Next, the data is sorted and shuffled to include tuples with the same key. Consider this: (Show on the whiteboard)

Tokyo, (1, 1)

New Jersey, (1, 1)

Sydney, (1, 1)

Ohio, (1, 1, 1)

This data is then sent to the reducer that then counts the values. The data in the reducer will be this: (Show on the whiteboard)

Tokyo, 2

New Jersey, 2

Sydney, 2

Ohio, 3

The reducer then shows the final output. (Show on the whiteboard)

Tokyo, 2
New Jersey, 2
Sydney, 2
Ohio, 3

To set this framework, a developer needs to consider location of the job input, location of job output, input format, output format, class containing Map function, and class containing reduce function. The MapReduce responsibilities of a developer are setting up the job, specifying the location of the input, and ensuring that the input is in the correct format and location. Responsibilities of framework are distributing jobs on ApplicationMaster and NodeManager, running the Map, sorting/shuffling, reducing and placing output in output directory and finally informing completion of the job.

In order to perform Hadoop MapReduce, a set of Java classes and methods are supplied to the developer for developing Hadoop MapReduce operation. These Java classes and methods are called the *user supply*. The work flow of a MapReduce job is defined by the *framework supply*. If the input location and the input format is as per the requirement of the program, then the ResourceManager hands over the job to the ApplicationManager, which then divides the job into smaller tasks. These tasks are then split between various NodeManagers. The NodeManager performs the map task of shuffling, partitioning, and sorting for each map output. Once the sorting is complete, the reducer starts the merging process, which is called the reduce task. The final output is then obtained once all tasks are reduced.

Eclipse is an IDE used for Java programming. In order to write a MapReduce program, the eclipse needs to be configured.

In a Hadoop environment, the objects must obey Writable interface. A Writable is a wrapper class for most of the primitive data type of Java. IntWritable in Hadoop can be compared to Integer class of Java. Writable creates serialized data in Hadoop. Serialization is used to convert Java object’s values or states into bytes to send it over a network. Java serialization is too tedious for Hadoop; hence, the Writable interface is introduced in Hadoop. Serialization of data reduces data size and allows easy transfer of data within the networks.

Reference: Refer to the following links for more information:

https://factspan.com/hadoop_architecture/
https://dzone.com/articles/word-count-hello-word-program-in-mapreduce
https://www.analyticsindiamag.com/why-mapreduce-is-still-a-dominant-approach-for-large-scale-machine-learning/
https://www.cloudera.com/documentation/enterprise/5-3-x/topics/cdh_ig_mapreduce_to_yarn_migrate.html

Say: Let us move to our next topic.

ACTIVITY

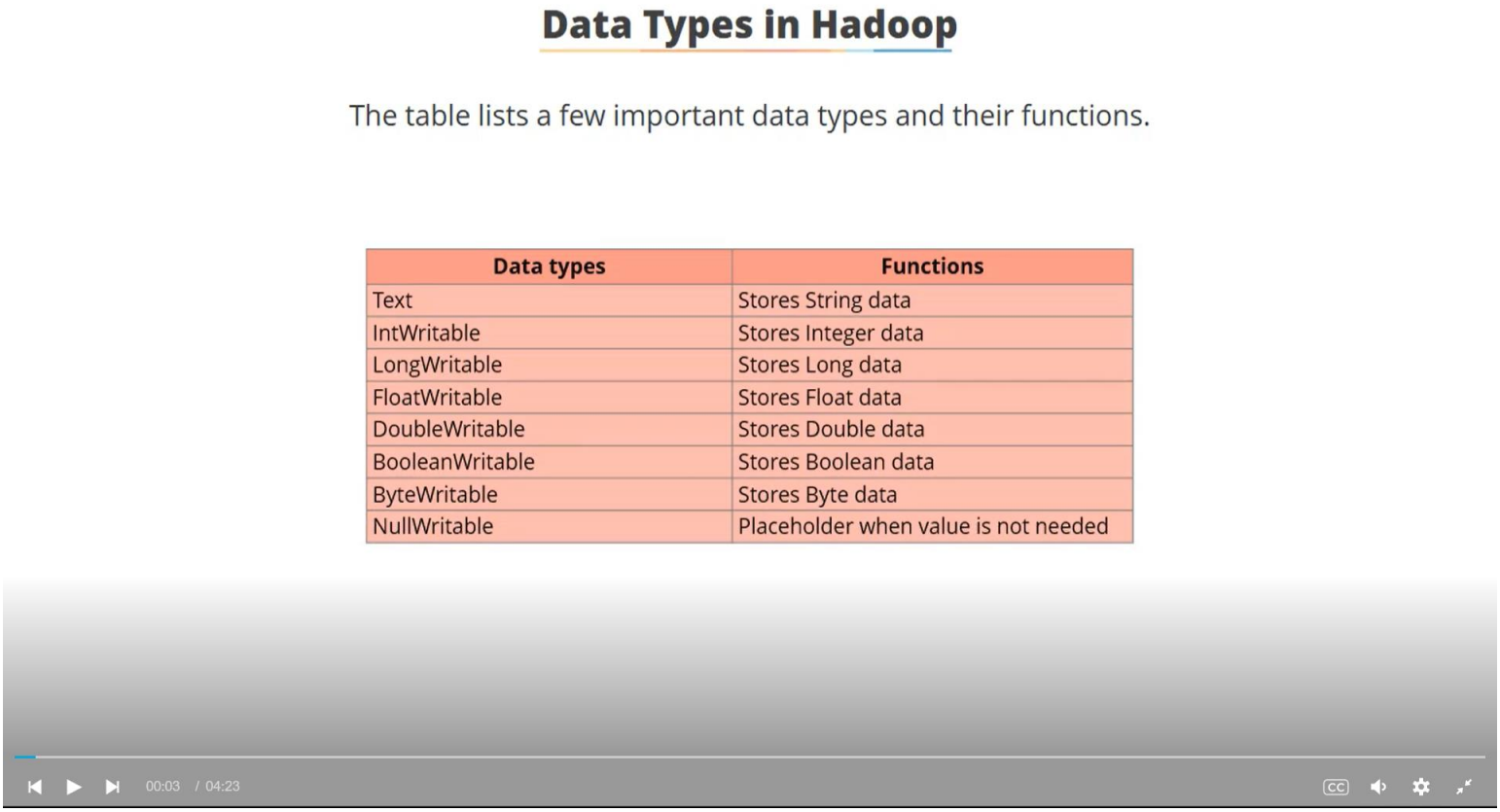
CLASS DURATION: 44 MINUTES

Consider that you have a simple ecommerce Website that sells T-shirts and shorts. You want to develop a MapReduce program that identifies the product sold out the maximum in a month.

Identify and describe every component that is required to develop this program including the programming elements such as the classes that the developers will use.

(Allow the students to refer online and discuss with peers for this exercise.)

DATA TYPES IN HADOOP



CLASS DURATION: 7 MINUTES

VIDEO DURATION: LESSON 3.4

FACILITATION GUIDELINES

Play Video Content: Lesson 3.4

Content: Similar to Java, there are various datatypes to store data such as integers, strings, and so on but as can be seen they are named a bit differently. This is because Hadoop uses the writable interface. Writable acts as a wrapper class to all primitive datatypes. Java already has wrapper classes such as integer, long, double, and so on but still there is a separate wrapper class for Hadoop. This is because in MapReduce, serialization needs to be considered. Thus, writables were accepted in Hadoop. Some of the datatypes in Hadoop are IntWritable to store integer data, LongWritable that stores long data and so on.

Reference: Refer to the following links for more information:

https://dzone.com/articles/hive-data-types
https://hadoop.apache.org/docs/r1.2.1/api/org/apache/hadoop/io/Writable.html
https://learnhadoopwithme.wordpress.com/tag/writablecomparable/
https://beyondcorner.com/learn-apache-hadoop/writable-writablecomparable-interfaces/

QUIZ

CLASS DURATION: 5 MINUTES

FACILITATION GUIDELINES

Content: Let us play a small quiz to check the understanding.

Questions:

1. What are the two phases in the MapReduce system?

Answer: Mapping and Reducing

2. What is the role of the application master?

Answer: It executes a single MapReduce job.

3. What is the role of the resource manager?

Answer: It decides how to assign available resources**.**

4. What is a cluster?

Answer: It is a set of host machines.

5. What is a container?

Answer: It is a logical reservation of resources.

QUERIES REGARDING SESSION

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1: Add the necessary jar files in the Hadoop environment to the MapReduce programs.

Session 4

Session Objectives:

- Understand what are joins
- Define and describe what is Sqoop
- Understand Basics of Hive and Impala

Video Content: Lesson 3 (3.5 onwards) and 4

QUICK RECAP

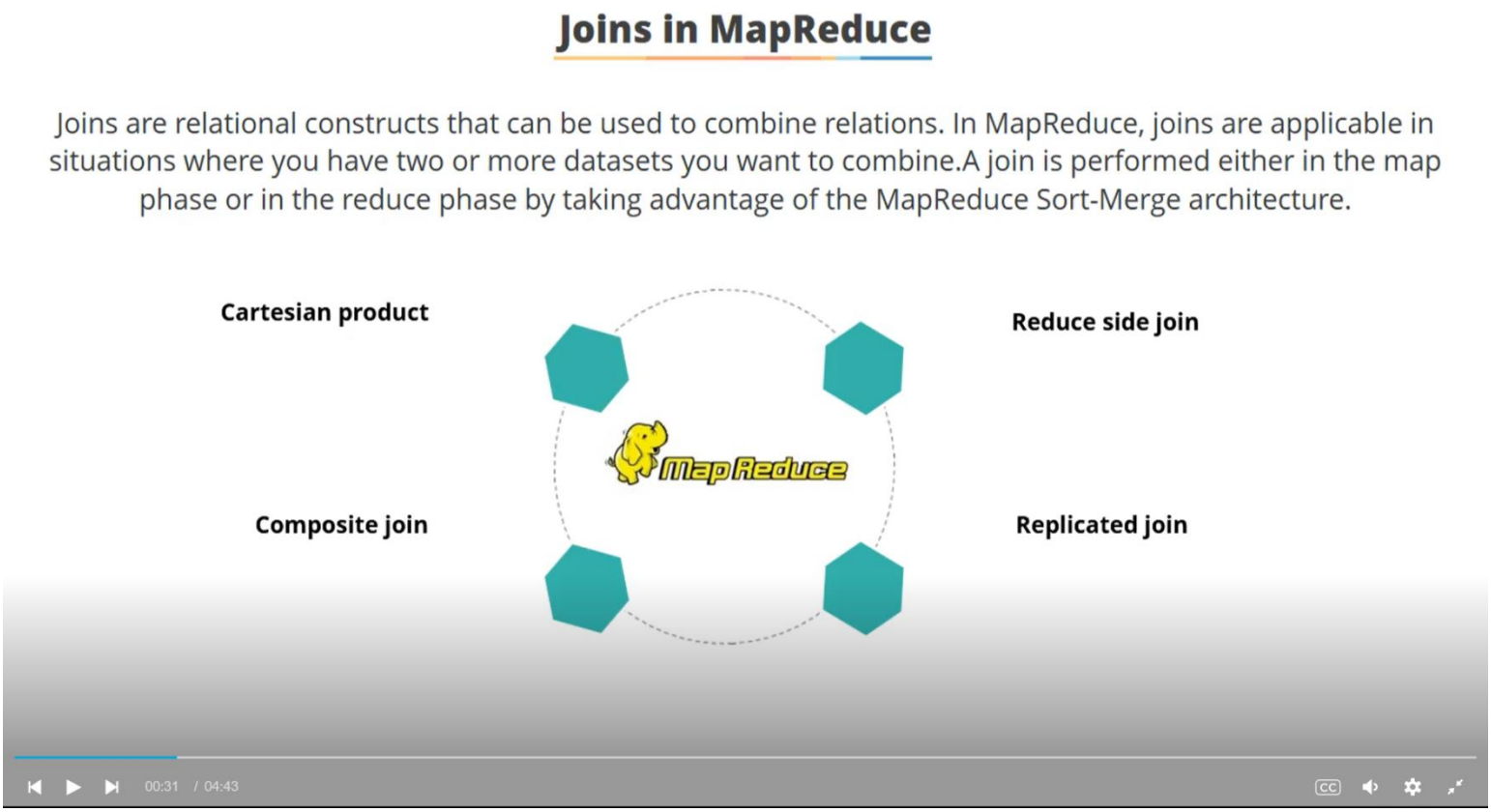
CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

JOINS IN MAPREDUCE



CLASS DURATION: 15 MINUTES

VIDEO DURATION: LESSON 3.5

FACILITATION GUIDELINES

Play Video Content: Lesson 3.5

Content: Joins in SQL are used to join two or more datasets or rows based on a foreign key. The final record obtained is usually a combination of two records. Joins in MapReduce are similar in purpose to those in SQL. Joins in MapReduce are of two types: Map side join and Reduce side join.
For a normal join operation, the job is assigned to the MapReduce task, which completes the task in two stages: Map Stage and Reduce stage. The mappers in the Map stage read the data from the two data tables. Then, they return the data with a key and value to an intermediate stage. In the shuffle stage, the intermediate data is sorted, merged, and fed to the reducer. The reducer performs the join task.
In the Map-only join, the task is performed at the Map side only and reduce stage is not needed. Here, one of the data tables needs to be smaller and the other one large. The smaller table should be able to fit into in-memory hash table. During a task, the data is moved from the in-memory hash table to the Hadoop distributed cache. These files are then populated to the mapper's local disk and the join function is carried out by the mapper. The Map side join reduces the time for a given task. It also reduces the cost as reduce side is no longer required. The disadvantage of the Map side join is that it works only when one set of data is small enough to fit into the memory. It will not work in cases where both the data tables are huge.

Let us now understand three different types of Map-side joins.

Replicated joins: They are very useful but can be carried out only with strict limitations. All of the datasets except the largest one should be read into memory of each map task that is limited by a Java Virtual Machine (JVM) heap. JVM heap is the amount of memory that can be allocated for a particular task. Also, replicated join is useful only for an inner or a left outer join where the large dataset is the left dataset. If these criteria can be fulfilled then replicated joins is a useful time saving and resource saving method.

Composite joins: This is used when large datasets are to be worked with. It is used when there is a need for inner join or full outer join. These datasets must be first sorted by a foreign key and partitioned by the foreign key.

Cartesian product: In a Cartesian product type of join, a mapper pairs every record of a dataset with every record of all other datasets. It can be used when there is no foreign key to perform the join operation. This requires a lot of computation time and resources. Thus, it is used in rare case scenarios.

Example: This example shows how joins can be used in organizations. Consider a multinational company where there are thousands of employees. Most of the companies maintain a separate table for the personal details of the employees such as the Employee name, age, address, and maintain a separate one for their professional details such as the department, qualification, and so on. Now, both the tables should contain a common field for linking. For that, the company provides an employee ID and this ID can be taken as the key that relates both these tables. Using this key, joins can be performed. Now, whenever full details of the any employee are required the joins can be used.

Reference: Refer to the following links for more information:

http://sungsoo.github.io/2014/09/18/what-is-map-side-join-and-reduce-side-join.html
http://codingjunkie.net/mapreduce-reduce-joins/

Resource: Ask students to go through these videos in their spare time:

https://www.youtube.com/watch?v=yU1M_ku-Q0c
https://www.youtube.com/watch?v=2CJyNP1BYc4

Say: Let us now move to our next topic.

WHAT IS SQOOP?

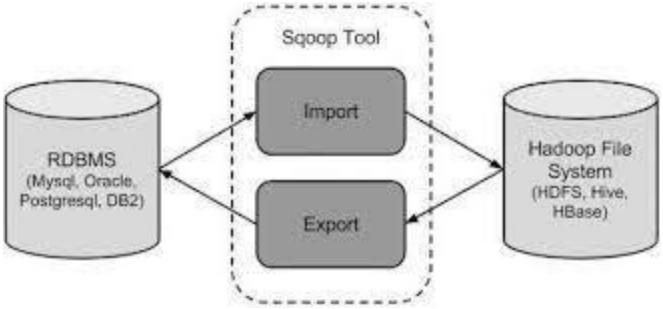
CLASS DURATION: 30 MINUTES

VIDEO DURATION: LESSON 3.6

FACILITATION GUIDELINES

Play Video Content: Lesson 3.6

Content: Sqoop is an Apache Hadoop Ecosystem project designed for supporting data transfer from SQL to Hadoop and Hadoop to SQL. It is a command line tool that transfers data between Hadoop and relational database servers and vice versa. This means that it imports data to Hadoop HDFS from relational databases such as MySQL, Oracle, and Teradata (loads of other RDBMS supported) and also exports data from HDFS to the relational databases.



(Image Courtesy: https://commons.wikimedia.org/wiki/File:Sqoop.jpg)

Traditionally, the Big Data generated by RDBMS was stored in relational database servers. To process data in Hadoop, first the data had to be transferred to the local file system and then transferred to Hadoop. This was a tedious procedure as well as time consuming. This is when Sqoop came into picture. Using Sqoop, data can be directly transferred to the HDFS or into Hive or HBase. Sqoop can

accommodate a variety of RDBMS and is also flexible in terms of data storage format. Sqoop ensures smooth transition of data from RDBMS to Hadoop.

Working of Sqoop import is as follows:

When a Sqoop command is submitted through a command line interface, Sqoop gathers the metadata (meaning information about the data such as data type, columns, and so on). As a second step, Sqoop assigns a Hadoop map-only job. There is no reduce phase as aggregation of data does not happen here. There is only fetching of data. By default, Sqoop assigns four mappers and splits the data amongst the mappers equally. Now, the mapper fetches the data assigned to itself by Sqoop and populates the HDFS output file.

While exporting data from HDFS back to RDBMS, the Sqoop export command is used. The database should contain the target table. For exporting new data into the target database, the Sqoop export command prepares INSERT statements. Similarly, for updating an existing row, Sqoop uses the UPDATE statement.

Similar to transferring data from RDBMS to HDFS, the data can be imported to HBase as well but some points need to be taken care of. Sqoop will not allow direct import of relational database to HBase as the structure does not map exactly to the relational database structure. First an HBase table needs to be created.

Sqoop connectors provide the necessary connectivity when connecting with various databases. The generic JDBC connector is the most basic connector in Sqoop. It uses the JDBC interface for accessing metadata and data transfer. It serves for most of the databases. Other than that, there are default Sqoop connectors, which is designed for MySQL, PostgreSQL, Oracle, Microsoft SQL Server, DB2, and Netezza. The third type is the Fast-path connector, which uses database-specific tools that ensures better data transfer performance.

Reference: Refer to the following links for more information:
https://sqoop.apache.org/docs/1.4.2
https://blogs.perficient.com/2016/08/11/apache-sqoop-a-means-to-work-with- traditional-database/

Resource: Ask students to go through this video in their spare time:

https://www.youtube.com/watch?v=9S4KifXQgmk

Say: With this, we end our current topic.

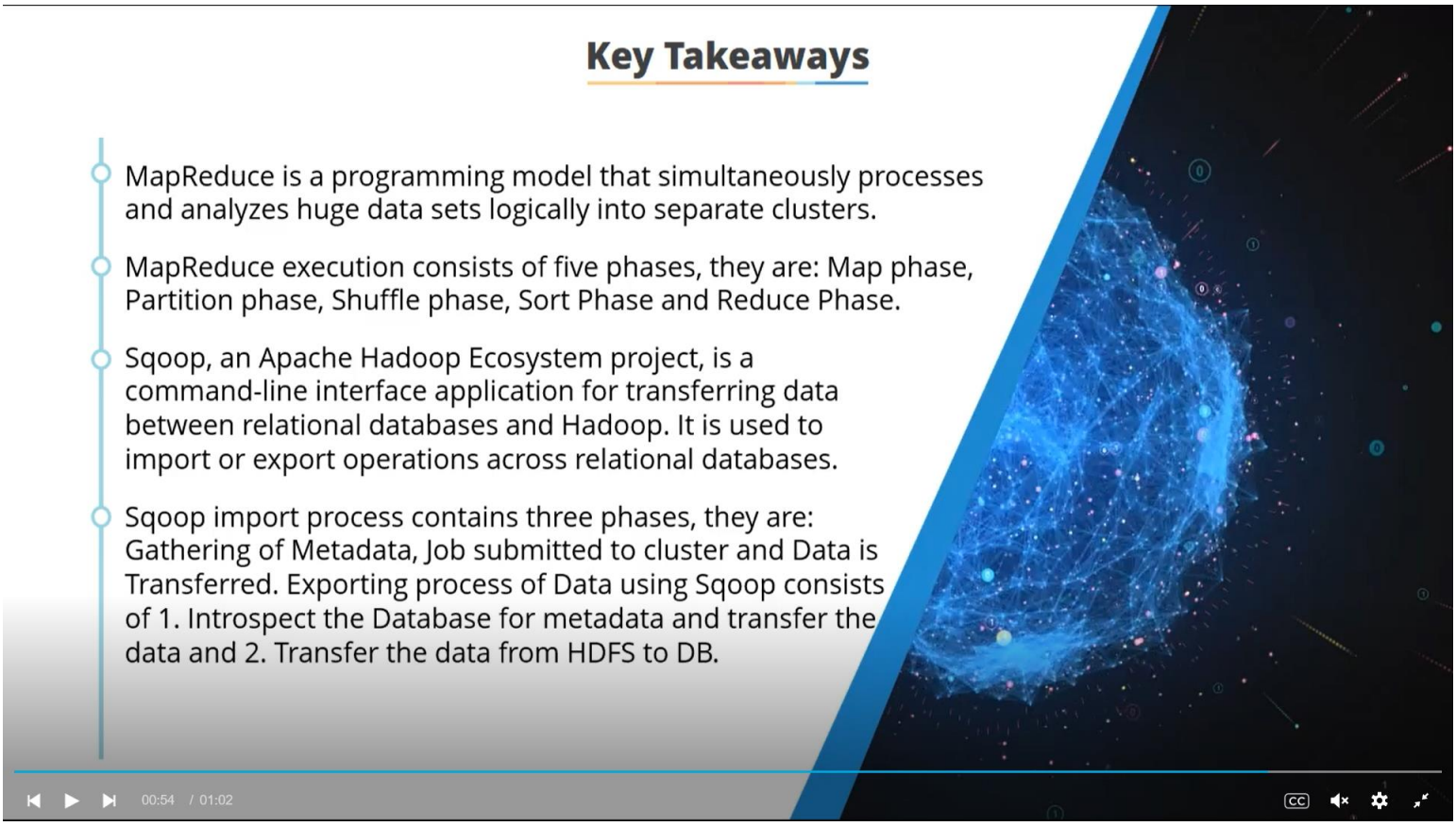
ACTIVITY 1

CLASS DURATION: 10 MINUTES

Make two groups of students. Ask one group to list out features of Sqoop and ask the second group to list out the limitations of Sqoop. Allow them to research on the Internet for this purpose.

Randomly, pick two students, one from each group, to present their findings.

KEY TAKEAWAYS



CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 3.7

FACILITATION GUIDELINES

Play Video Content: Lesson 3.7

Content: Rewrite the key takeaways of the session on the whiteboard. Tell the students that they have learnt about MapReduce and Sqoop.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Content: Lesson 3.8 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions from the following set:

Questions:

1. What is the function of the Shuffle phase?

Answer: In the shuffle phase, the input data is fetched from all map tasks for the portion corresponding to the reduce task's bucket.

2. Which are the main configuration parameters that should be specified in MapReduce?

Answer:
- The input location of the job.
- The output location of the job.
- The input's and output's format.
- The classes containing map and reduce functions.

3. What are the three InputFormat classes in MapReduce?

Answer:
- KeyValueTextInputFormat
- MultiFileInputFormat
- NLineInputFormat

4. What is the function of SequenceFileOutputFormat?

Answer: It writes sequence files to save the output. It is compact and compressed.

Say: Let's now move on to Lesson 4.

INTRODUCTION

CLASS DURATION: 15 MINUTES

VIDEO DURATION: LESSON 4.1

FACILITATION GUIDELINES

Play Video Content: Lesson 4.1 till 00:40 seconds.

Content: After playing the video, repeat all the points and tell the students that in this session, they will learn about basics of Hive and Impala.

Play Video Content: Resume Lesson 4.1

Content: Hive is a data warehousing system built on top of Hadoop. It is used for data analysis. Hive uses HiveQL to write queries. Impala is an open source SQL engine for Hadoop. It is a Massive Parallel Processing (MPP) SQL query engine that processes voluminous data stored in the Hadoop cluster.

Hive and Impala, both provide SQL-like interface to access data from Hadoop.

Similarities of Impala and Hives are that both can be operated upon without having any knowledge of Java. The data can be processed with enough knowledge of traditional SQL. By default, Hadoop uses MapReduce programming to process data in the Hadoop cluster. However, it is time consuming as it requires massive amounts of code to be written. HiveQL and Impala SQL require a few lines of code as compared to a MapReduce program. Using Hive and Impala, the data is made accessible to a broader audience. Both Hive and Impala offers interoperability with other systems.

Now that we have discussed similarities between Hive and Impala, let us discuss differences between the two.

Hive is a product of Facebook whereas Impala is developed by Cloudera. Hive is a data warehousing package built on top of Hadoop whereas Impala is a hardcore SQL query engine. Hive is slower as

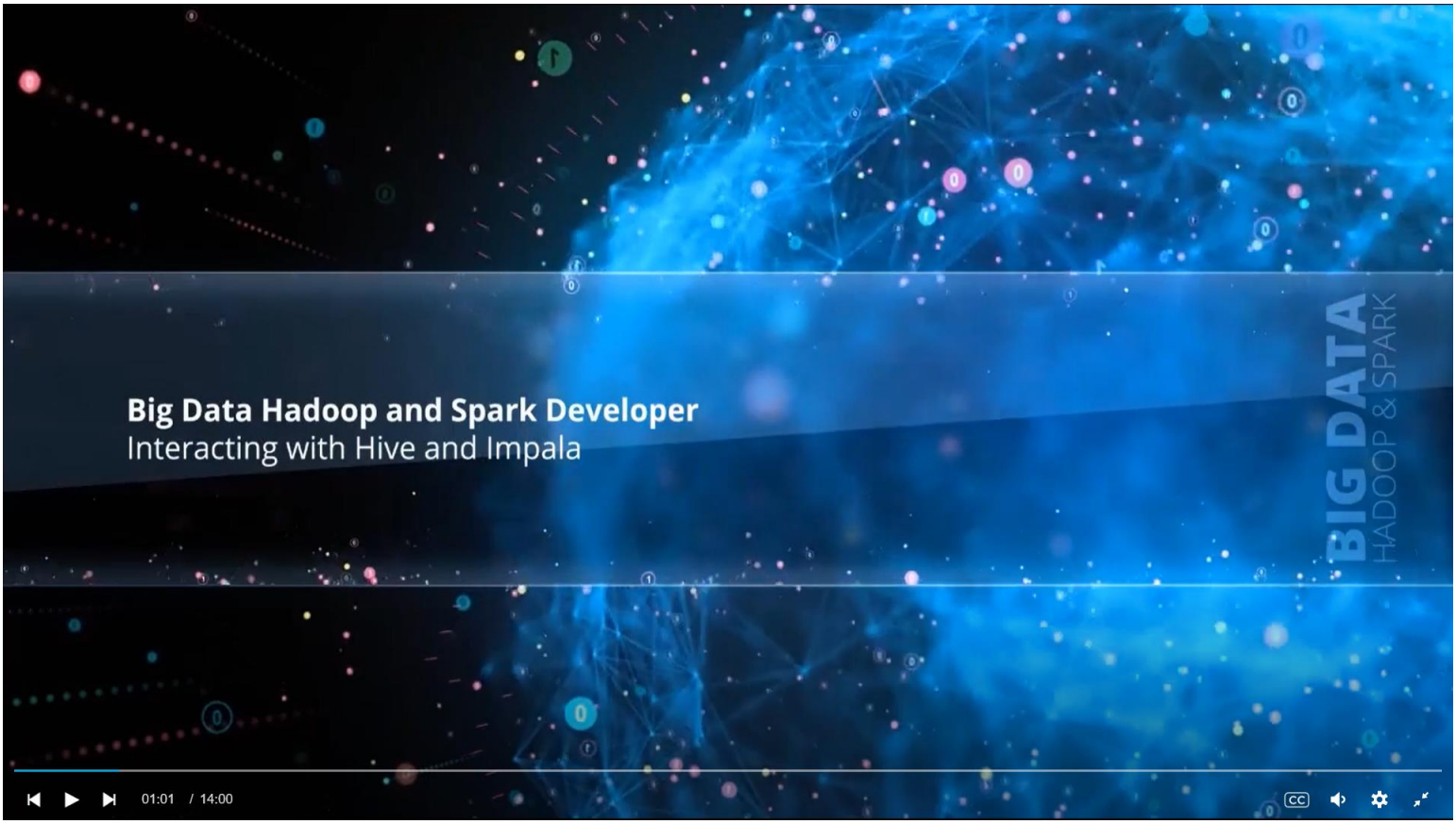
compared to Impala. Considering the types of file format, Hive is more general. Impala, on the contrary, is much faster as it specializes in certain specific file formats. Impala has low query latency. Hive is mostly suitable for querying structured data whereas Impala is used for high-concurrency and ad hoc queries.

Cloudera Impala and Apache Hive are considered as fierce competitors. Impala can perform interactive computing whereas Hive cannot. Hive is known for its cold start, that is, it takes time in starting a query. This happens with the systems that use MapReduce program. Impala provides a quick startup as it avoids any startup overheads. The daemon processes are started at boot time itself making it query ready. Also, Hive follows batch processing whereas, Impala can accommodate concurrent users. Hive is fault tolerant and processes queries even if one node gets faulty during operation. In such a situation, Impala will start all over again as it is not fault tolerant. Basically, Impala is used for starting new projects and Hives can be used for projects that require upgrades.

Reference: Refer to the following links for more information:
https://mapr.com/products/apache-hive/
https://impala.apache.org/overview.html
https://www.youtube.com/watch?v=vmiWOlcnFW8

Say: Now, let us move to our next topic.

INTERACTING WITH HIVE AND IMPALA



CLASS DURATION: 16 MINUTES

VIDEO DURATION: LESSON 4.2

FACILITATION GUIDELINES

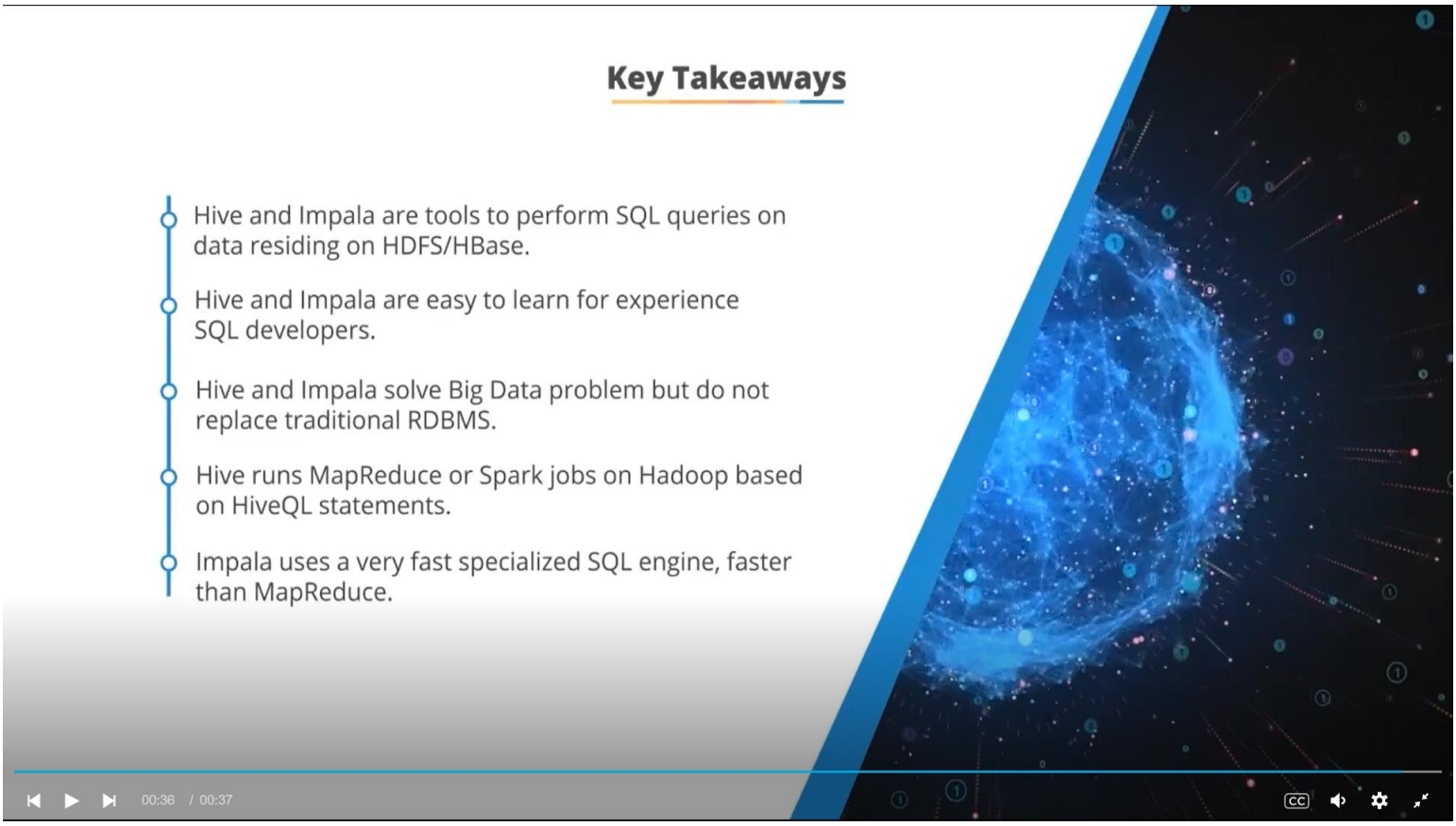
Play Video Content: Lesson 4.2

Content: Reiterate that Impala is faster than hive. This is because while executing a query Impala does not process the data using MapReduce, but executes the query on the cluster. The command line shell for Impala is Impala shell whereas, for Hive, it is Beeline. Hue is an open source Web interface for Apache Hadoop. It can be used to write Hive and Impala query from a user interface. The user need not use a command line interface to use Hadoop if Hue is used. In Hue Web UI, Hive can be accessed using Hive Query Editor and Impala through Impala Query Editor.

Reference: Refer to the following links for more information:
https://www.cloudera.com/documentation/enterprise/5-9-x/topics/impala_impala_shell.html
https://www.educba.com/hive-vs-impala/

Say: With this, we conclude our session.

KEY TAKEAWAYS



CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 4.3

FACILITATION GUIDELINES

Play Video Content: Lesson 4.3

Content: Rewrite the key takeaways of the session on the whiteboard. Tell the students that they have learnt about Hive and Impala along with their similarities and differences.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Content: Lesson 4.4 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.
You may choose to ask few more questions.

Questions:

1. What are the two similarities between Hive and Impala?

Answer:
- Both bring large-scale data analysis to a broader audience.
- Both are more productive than writing MapReduce code.

2. Which are the two differences between Hive and Impala?

Answer:
- Hive is fault tolerant and the query will be executed even in case of a node failure whereas Impala is not fault tolerant.
- Hive is a high level abstraction layer whereas Impala is an SQL engine.

3. What are the command line shells for Impala and Hive called?

Answer: The command line shell for Impala is Impala shell and that for Hive is Beeline.

4. Which is the command to show added details of queries in Beehive?

Answer: !verbose

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1:

a. Assuming you have a training data set and you use MySQL JDBC connector, write the Sqoop command to replace null with /n.

b. Use Sqoop to import only accounts where the person lives in California.

Session 5

Session Objectives:

- Explain Hive and Impala Metastore
- Use Impala SQL and HiveQL DDL to create tables
- Create and manage Hive tables using Hue or HCatalog
- Load data into Hive and Impala tables using HDFS and Sqoop

Video Content: Lesson 5

QUICK RECAP

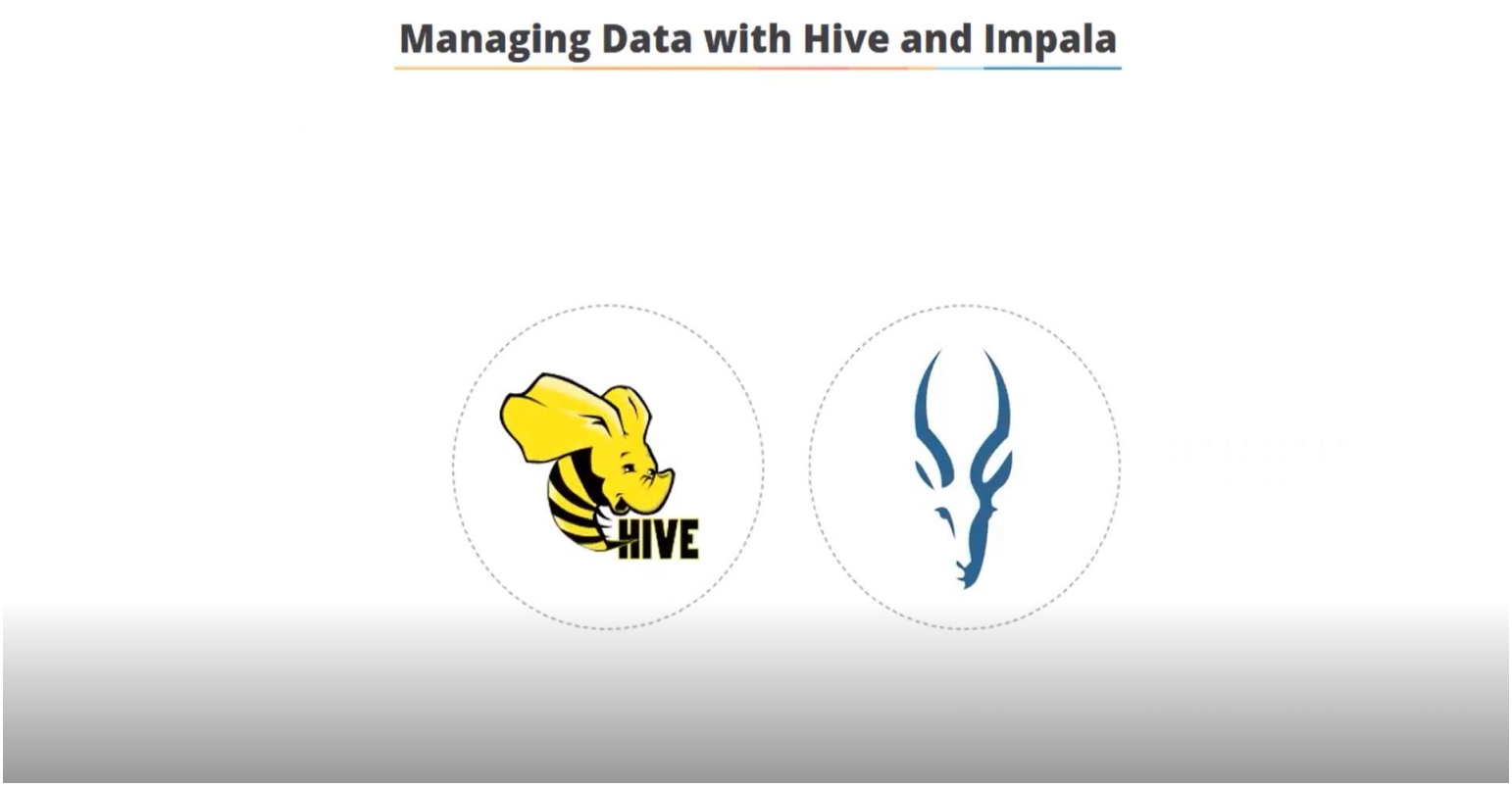
CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome the students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

WORKING WITH HIVE AND IMPALA



CLASS DURATION: 20 MINUTES

VIDEO DURATION: LESSON 5.1

FACILITATION GUIDELINES

Play Video Content: Lesson 5.1

Content: Unlike traditional RDBMS, Hive and Impala do not store data. The data is actually stored in a separate system such as HDFS. In case, data is stored in some other supported file system, it can be queried from there. No data needs to be moved. When a table is created, it is stored at the default location: `/usr/hive/warehouse` directory of HDFS.

Example: If a Hive table by the name *employee_details* is created, then it will be stored in a default path */usr/hive/warehouse/employee_details*. Hive data can be split into more than one file in the HDFS.

Metadata for Hive is stored in Metastore. The Metastore is the central repository in Apache Hive. The Metastore Hive is RDBMS. This is because with any change in table structure, the Metastore needs to be updated. In HDFS, update is not feasible. Thus, Hive stores metadata in Metastore, which is then stored in RDBMS. Hive and Impala work with the same data.

Metastore can be thought of as a data dictionary, which contains metadata. A Metastore stores information about tables such as table structure, table columns, data type, and so on. The metadata is stored in the traditional RDBMS format but Hive comes with Derby Database to store the metadata. Derby is okay to use for testing for developer purposes, however, for running multiple instances in real time, JDBC compliant databases such as MySql need to be used. Here, MySql stores only the metadata and not actual data.

Hive and Impala, both use Metastore to interpret the table structure and find the location of the data.

The query is first sent to Hive or Impala, which is then forwarded to the Metastore. The Metastore then provides details of the structure and location of the table in the HDFS. The server then queries the actual data present in the HDFS.

Example of Metastore: Consider an online grocery store like the Walmart Grocery. It needs to maintain a record of groceries ordered every day. To maintain the data about the orders, typically a table is maintained in HDFS. The metadata to this table, however, can be maintained in Hive in a table by the name *order_table*. It will contain columns such as *order_id, order_item, order_addr*. Now, this information is the metadata stored in the Metastore. It is not the actual data. The actual data is stored in files in HDFS. Thus, Hive references to the HDFS files. The Metastore contains the schema of the table and the location of the file in HDFS. Whenever a query is raised for *order_table*, the Metastore will point to the location of the file in the HDFS directory. The data can then be fetched from that particular location in HDFS.

When creating a database and tables, Hive and Impala use Data Definition Language (DDL) such as HiveQL and Impala SQL respectively. There are some minor differences between HiveQL and Impala SQL. Some of them are: Impala does not support *DESCRIBE DATABASE, EXPORT TABLE, IMPORT TABLE* and so on. The *serialization.null.format* table property holds true only for TEXT tables and this property is ignored for Parquet and other formats in Impala. What is *serialization.null.format*? It is the property to treat an empty field in the data files as NULL when querying a table. NULL values are then written to the data files as empty fields when rows are inserted to the table. Parquet is an open source file format for Hadoop. Hive respects the *serialization.null.format* property for Parquet and other formats and converts matching values to NULL during the scan.

Syntax to create a table in Hive is as follows: (Write this on the whiteboard.)

```
CREATE TABLE <tablename> (colnames DATA TYPES)
```

This is the first line of code syntax.

```
ROW FORMAT DELIMITED
```

This is the second line of the code syntax. This line tells Hive that fields in each file are delimited by a character, which is `Control-a` by default. However, it can be developer specific also.

```
FIELDS TERMINATED BY char
```

This is the third line of the code, which specifies that each field of data is terminated. This indicates the delimiter that separates columns.

```
STOREAS {TEXT FILE OR SEQUENCEFILE}
```

This is the last line of the code. The file format to be stored is declared here.

Example: So, the table *emp_detail* can be created in the following manner:

```
CREATE TABLE emp_detail(emp_id int, emp_name string, emp_age int);
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

Reference: Refer to following links for more information:

www.cloudera.com/documentation/enterprise/5-8-x/topics/cdh_ig_hive_metastore_configure.html

https://jaceklaskowski.gitbooks.io/mastering-spark-sql/spark-sql-hive-metastore.html

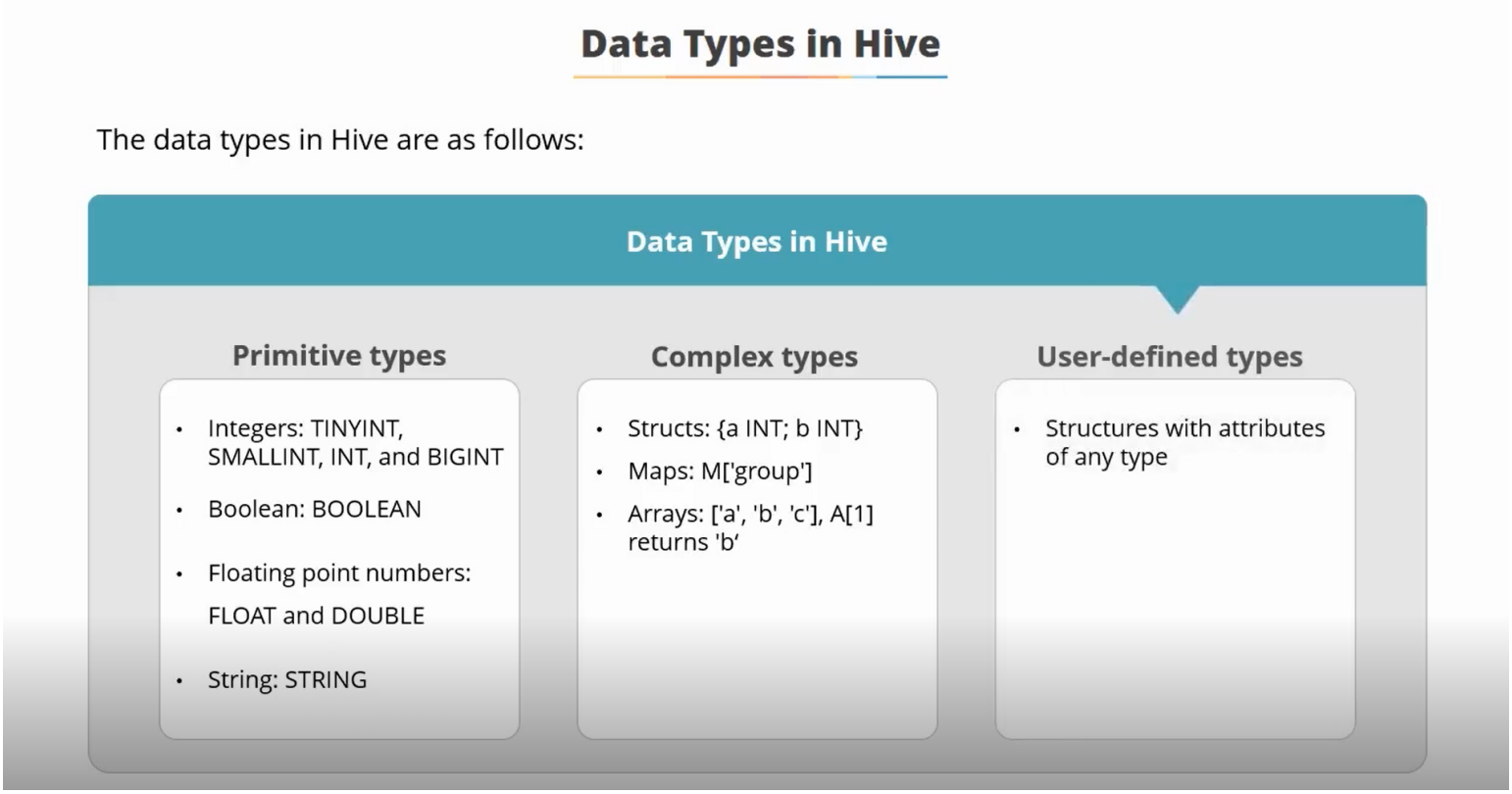
https://impala.apache.org/docs/build/html/topics/impala_ddl.html

https://www.youtube.com/watch?v=OHS_ZOaDeFk

Ask students to write down statements to create their own table in Hive. Give them 5 minutes to complete the task.

Say: Let us move to the next topic.

DATA TYPES IN HIVE



CLASS DURATION: 18 MINUTES

VIDEO DURATION: LESSON 5.2

FACILITATION GUIDELINES

Play Video Content: Lesson 5.2

Content: The data types in Hive can be classified into three categories: primitive, complex, and user-defined. (Write the data types on the whiteboard.)

Primitive data types include:

- Numeric – TINYINT, SMALLINT, FLOAT, DOUBLE, DECIMAL
- Date/time – TIMESTAMP, DATE
- String – STRING, VARCHAR
- BOOLEAN, BINARY

Complex data types include Structs, Maps, and Arrays.

User defined data types include structures with attributes of any type.

Reference: Refer to following links for more information:

https://cwiki.apache.org/confluence/display/Hive/LanguageManual+Types

http://support.sas.com/documentation/cdl/en/fedsqlref/67364/HTML/default/viewer.htm#p1k4nui8i2daihn1n5ll3nzsx0ww.htm

Now, let us understand how to change the data location in Hive.

Default location of table created is `/user/hive/warehouse`. If this location needs to be changed, then, use LOCATION to pick a location in HDFS.

Syntax for the table *emp_detail* will be as follows: (Write on the whiteboard.)

```
CREATE TABLE emp_detail(emp_id int, emp_name string, emp_age int);

ROW FORMAT DELIMITED

FIELDS TERMINATED BY '\t'

LOCATION 'emplocation'
```

`emplocation` is the location where the table needs to be stored.

Tables in Hive are classified as: *internal tables* and *external tables*. Internal table is also called the managed table. When a table is created in Hive, it moves the data into its warehouse directory. External table on the other hand refers to data that is present outside its warehouse directory, in some external location.

Some of the differences between internal and external tables are: Hive is solely responsible for security of the data in a managed (internal) table whereas external table can be accessed by anyone who has access to the HDFS directory. Internal data is deleted and data is lost forever when deleting the managed tables. However, in case of dropping external tables, only the metadata is lost and not the actual data.

Reference: Refer to following links for more information:

https://acadgild.com/blog/managed-and-external-tables-in-hive

https://www.youtube.com/watch?v=QrkUDq7jXQc

https://www.youtube.com/watch?v=QrkUDq7jXQc&t=113s

Discussion: Give reason why on dropping an external table in Hive, only metadata is lost and one can still have access to the actual data.

Reason: External table is just a metadata, which does not contain the actual data. The actual data resides in a location in the HDFS directory. This is why whenever an external table in Hive is dropped only the actual data remains safe.

Say: Let us do an activity.

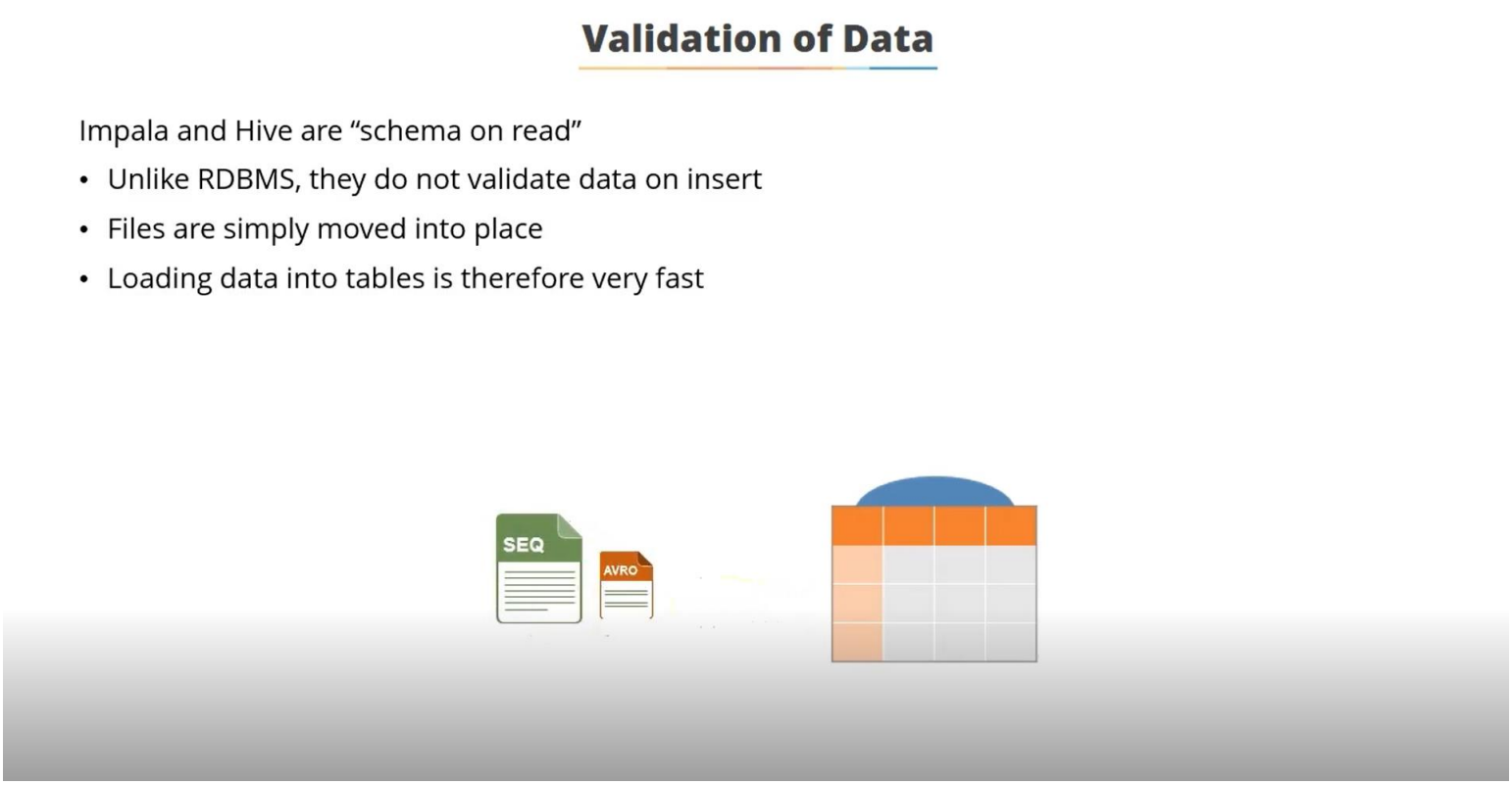
ACTIVITY 1

CLASS DURATION: 10 MINUTES

Write the difference between internal and external tables of Hive.

Internal Table	External Table

VALIDATION OF DATA



CLASS DURATION: 18 MINUTES

VIDEO DURATION: LESSON 5.3

FACILITATION GUIDELINES

Play Video Content: Lesson 5.3

Content: Traditional RDBMS follows 'schema on write' strategy. Here, a schema is created before any data is loaded into the table. For example, to create the **emp_detail** table, the structure of the table is defined beforehand, that is, it will contain emp_id, emp_name, emp_age, and emp_department. Next, the data is filled into its predefined position using the Insert statement.

However, unlike the traditional RDBMS, Impala and Hive follow 'Schema on read' data analysis strategy. In this method of data analysis, unstructured data can be stored in the database. Thus, a schema need not be defined before loading the data into the database. This schema is used in HDFS where data of any type, shape, size can be loaded. Metadata for the actual data is created and stored to know about what data is stored and in what location in the HDFS directory. This makes 'schema on read' strategy much faster to load than the traditional method of 'schema on write'. However, when a particular data is queried from a particular location, then the schema needs to be defined. Thus, errors while using schema on read is discovered only when queries are performed.

Here are the advantages of 'schema on read':

- Faster to load the data.
- No data is discarded or altered, hence flexible querying capabilities.
- Data from different data sources are stored at the same place. Thus, data can be queried from any data source at a given time as per the requirement.

Here are the disadvantages of 'schema on read':

- Slow query speed.
- Since the data is not validated before being loaded, there might be possibilities of inaccurate or duplicate data that might lead to inappropriate query products.
- Data has to be structured when querying, which is quite a complex and time-consuming task.

Let us understand the concept of 'schema on write' and 'schema on read' with a small example.

Example: Consider that we need to create a table for order details for a particular online store. Let the table name be **order_details**. Let us create this table with only two values, say order_item, order_qty in a traditional RDBMS as well as Hive.

Consider following table: (Write this on the whiteboard.)

Order_item	Order_qty
Shirt	2
Skirt	5

Now, consider this table: (Write this on the whiteboard.)

Order_item	Order_qty
Shirt	2
5	Skirt

As shown in the table, if data similar to the second row is inserted, then RDBMS will not accept it due to data type mismatch. This is because the traditional RDBMS follows the 'schema on write'. Hence, the schema is predefined and any change from the predefined schema is outright rejected.
In the case of Hive, such type of data insertion is acceptable but only when the data is being loaded. However, when such data is queried, Null will be inserted in place wherever there is datatype mismatch. Apache Sqoop is used for this purpose of transferring bulk data to Hive and Impala from an RDBMS. Sqoop has a built-in option to load data to Hive and Impala. This can be done using the hive-import option. The hive-import creates a table accessible in both Hive and Impala.

Reference: Refer to following links for more information:

https://www.techopedia.com/definition/30153/schema-on-read

https://www.i2tutorials.com/hive-tutorial/hive-schema-read-vs-schema-write/

https://www.youtube.com/watch?v=cfEYEah1XMg

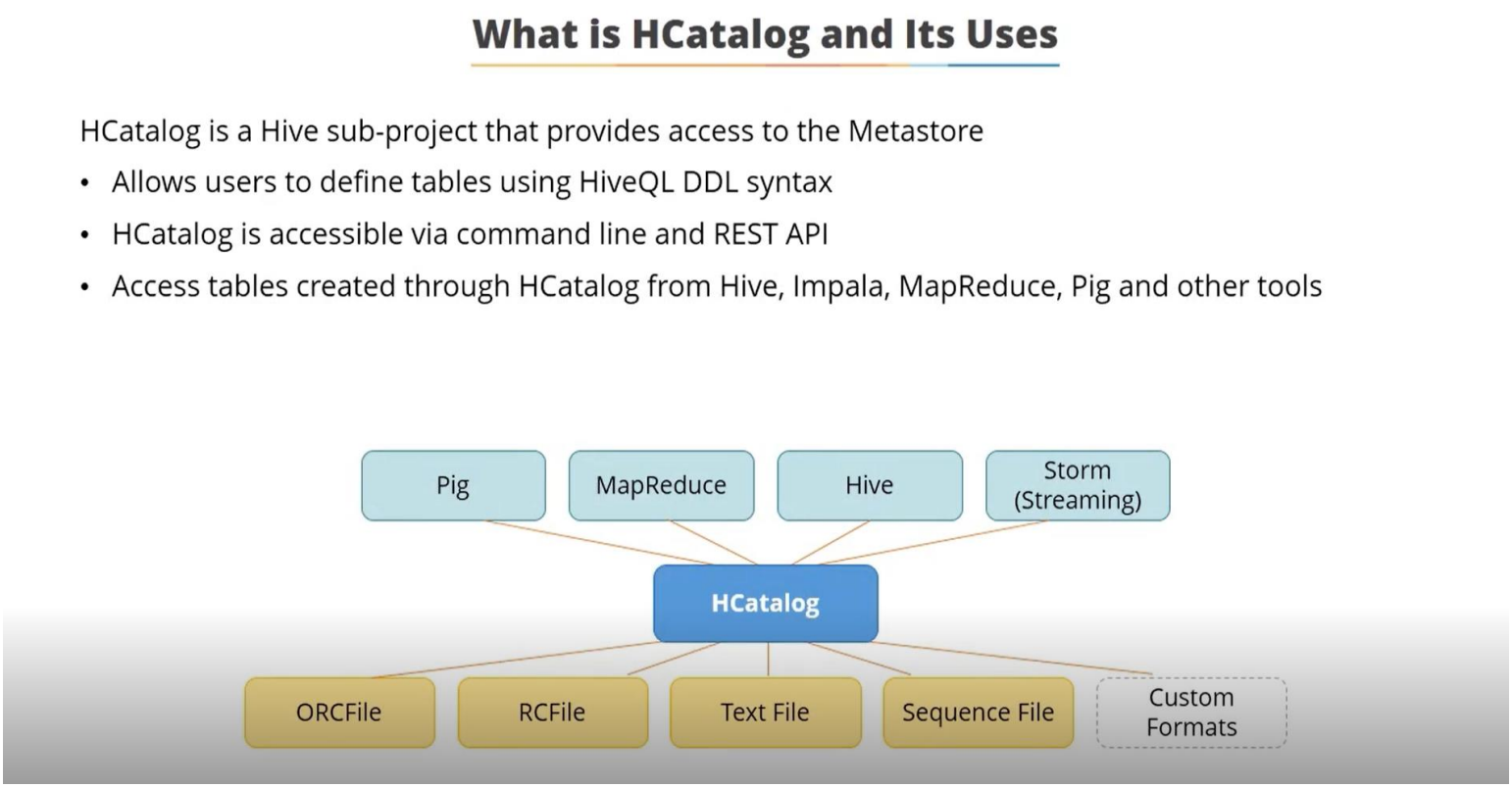
Say: Let us proceed with the next topic.

ACTIVITY 2

CLASS DURATION: 13 MINUTES

Research and present the Hcatalog architecture in the class.

WHAT IS HCATALOG AND ITS USES?



CLASS DURATION: 12 MINUTES

VIDEO DURATION: LESSON 5.4

FACILITATION GUIDELINES

Play Video Content: Lesson 5.4

Content: HCatalog is a table and metadata management tool for Hadoop. It allows users to share metadata amongst variety of systems such as Hive, Pig, Mapreduce, and so on. In any Hadoop environment, there are varieties of tools that are being run. These tools do not agree on many issues such as schema, data types, and what and where the data is stored. HCatalog tends to solve all these problems by providing one data model for all these tools, allows a shared schema, and also allows these tools to see when each other's data is available. It also provides a table abstraction by which the user's need not worry about the how and where the data is stored.

Catalog and Statestore are the components of Impala Master Node. Daemons are the processes that run in the background. The Statestore provides information as to which daemons in the cluster are healthy. It periodically checks their status to see which of them can accept new work. In case of an Impala daemon failure due to network error, hardware or any such issues, the Statestore informs all the other Impala daemons. This allows any future queries to be avoided to that particular Impala daemon.

The Impala component, Catalog passes on information about the metadata changes from Impala SQL statements to all the Impala daemons in a cluster.

The metadata cache contains all the file system metadata; this metadata includes all directory contents, and file status information.

Reference: Refer to following links for more information:

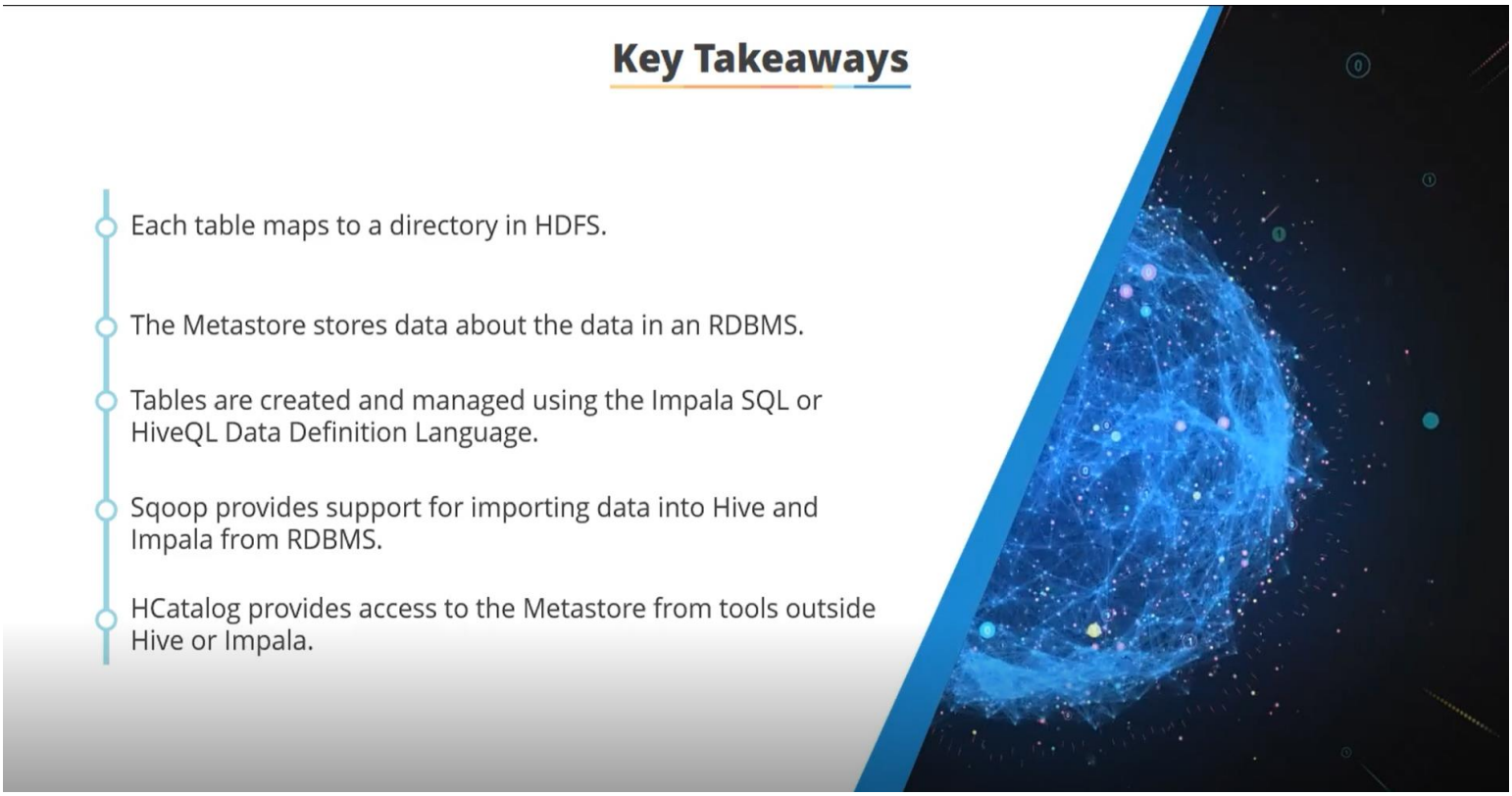
https://www.youtube.com/watch?v=gTwhSAEEe1I

https://www.bmc.com/blogs/what-is-apache-hcatalog-hcatalog-explained/

https://www.cloudera.com/documentation/enterprise/5-9-x/topics/impala_components.html

https://impala.apache.org/docs/build/html/topics/impala_components.html

KEY TAKEAWAYS



CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 5.5

FACILITATION GUIDELINES

Play Video Content: Lesson 5.5

Content: Rewrite the key takeaways of the session on the whiteboard. Tell the students that they have learnt about types of tables in Hive, Metastore concepts, and HCatalog.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Content: Lesson 5.6 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions.

Questions:

1. What is the default location of tables in Hive? Can we change the location of managed tables?

Answer: The default location where tables are stored in Hive is */usr/hive/warehouse*. Yes, it is possible to change the default location of a managed table using – LOCATION '<hdfs_path>'.

2. What is the use of Metastore in Hive and Impala?

Answer: The Metastore is used to get table structure and location of data.

3. What are the differences between external and internal tables in Hive?

Answer:

- Hive is solely responsible for security of the data in a managed (internal) table whereas external table can be accessed by anyone who has access to HDFS directory.
- Internal data is deleted and data is lost forever when dropping the managed tables. However, in case of dropping external tables, only the metadata is lost and not the actual data.

4. What is HCatalog?

Answer: HCatalog is a table and metadata management tool for Hadoop. It allows users to share metadata amongst variety of systems such as Hive, Pig, MapReduce, and so on.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1: Download a sample dataset from this link: https://openflights.org/data.html. Upload the dataset in Hive.

Session 6

Session Objectives:

- Discuss the different types of file formats
- Explain data serialization
- Understand how to improve Query Performance with concepts of data file partitioning
- Define Hive Query language (HiveQL)
- Define ways in which HiveQL can be extended

Video Content: Lesson 6 and 7

QUICK RECAP

CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

INTRODUCTION

CLASS DURATION: 2 MINUTES

VIDEO DURATION: LESSON 6.1

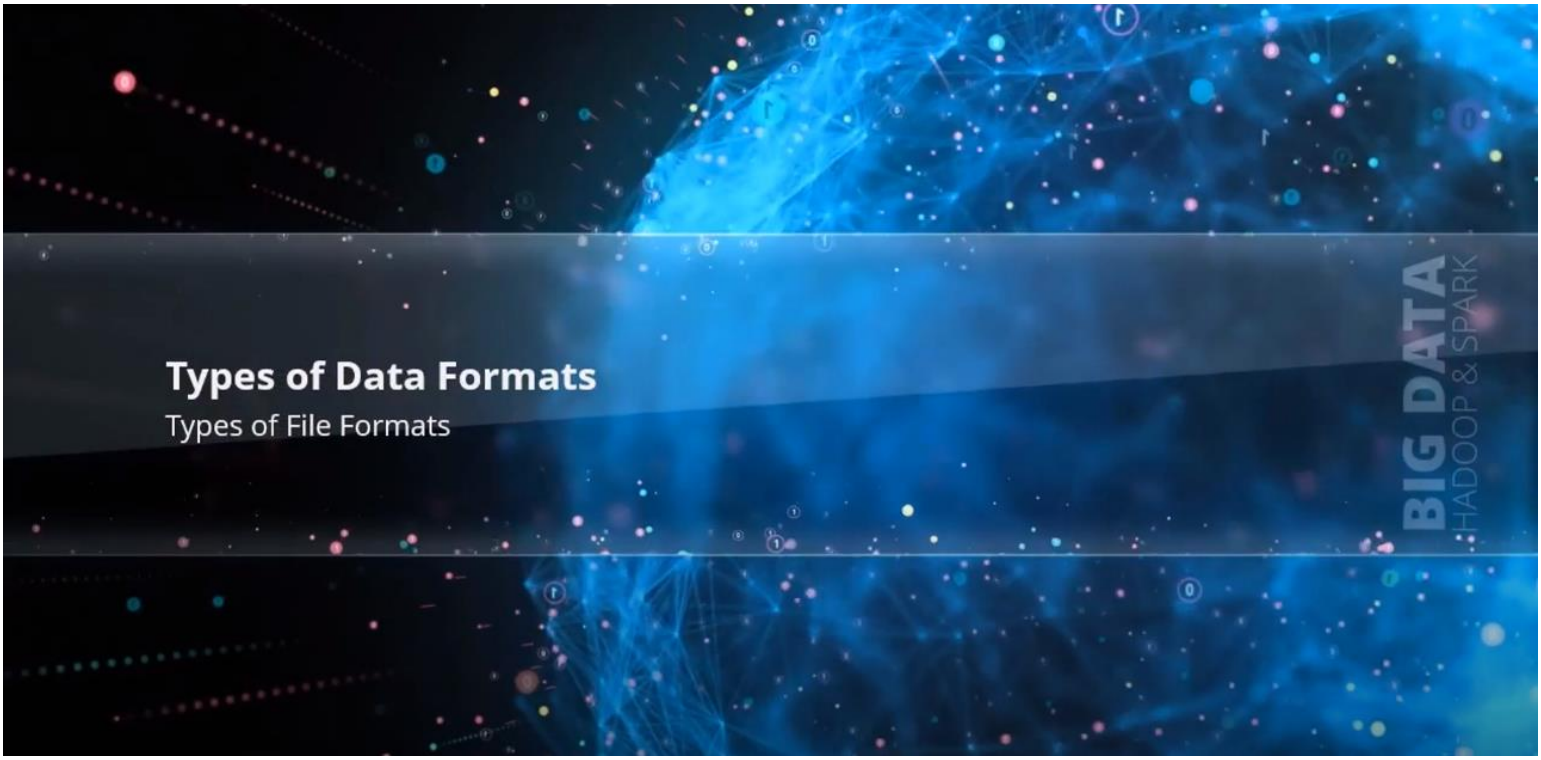
FACILITATION GUIDELINES

Play Video Content: Lesson 6.1

Content: After playing the video, repeat all the points and tell the students that in this session they will learn about different types of file formats and how data is represented in the storage memory as a series of bytes.

Say: Let us begin.

TYPES OF FILE FORMAT



CLASS DURATION: 12 MINUTES

VIDEO DURATION: LESSON 6.2

FACILITATION GUIDELINES

Play Video Content: Lesson 6.2

Content: Hive and Impala tables can be created in HDFS using following file formats: text files, sequence files, Avro data files, and Parquet files. Text files are the most commonly used file format because they can be easily read and written in any programming language. The only flaw is that it can only contain textual data. Sequence files are binary files. They store data in key-value pairs. A sequence file is primarily used to process several small files as a single file for MapReduce programs. Avro data format is ideal for long term storage of data as it can read and write in many languages. It stores schema and data is stored in binary format. It is a preferred file format to serialize data. In addition, it supports multiple data formats. This makes the data to be processed by different programming languages.

The Parquet file format refers to columnar storage format where values of each column are stored together. Consider the table: (Write this on the whiteboard.)

ID	Name	Department
Emp001	John Pat	Marketing
Emp002	Nick Steven	HR
Emp003	Joe Jonathan	Marketing

In the columnar storage format, this data will be stored as: (Write this on the whiteboard.)

Emp001	Emp002	Emp003	John Pat	Nick Steven	Joe Jonathan	Marketing	HR	Marketing

What are the benefits of a columnar storage format? Less data storage space is required as data of the same type are stored together and compression works best. Next fetching data is easy as values are stored together. This means less I/O and less bandwidth will be required. Also, when big data is loaded in Hadoop, columns are more, a parquet file format becomes a great choice when storing data.

Reference: Refer to the following links for more information:

https://nxtgen.com/hadoop-file-formats-when-and-what-to-use
https://www.slideshare.net/StampedeCon/choosing-an-hdfs-data-storage-format-avro-vs-parquet-and-more-stampedecon-2015
https://blog.matthewrathbone.com/2016/09/01/a-beginners-guide-to-hadoop-storage-formats.html
https://blog.cloudera.com/benchmarking-apache-parquet-the-allstate-experience/

Say: Let us do an activity.

ACTIVITY 1

CLASS DURATION: 30 MINUTES

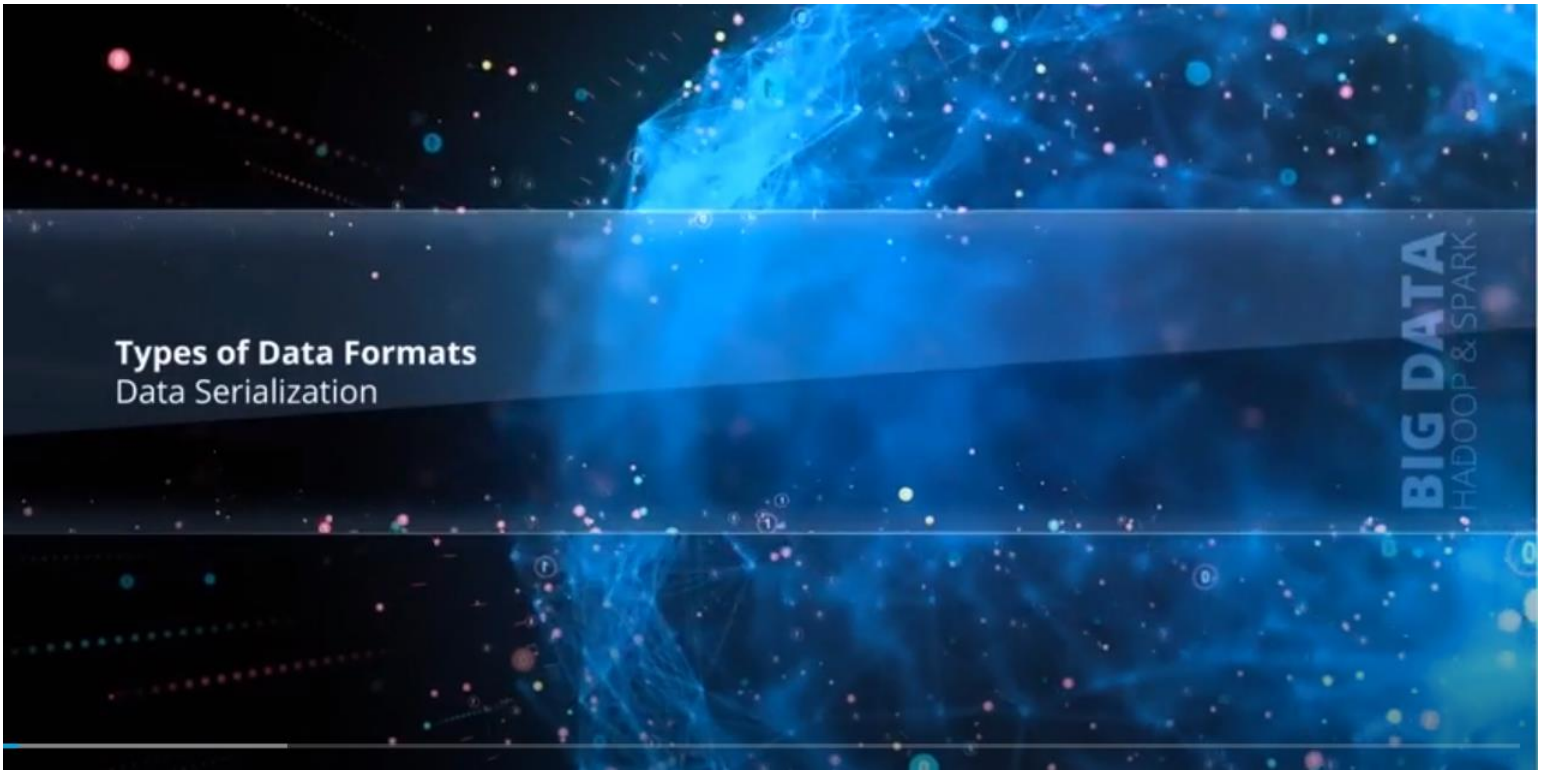
Hand out sheets of blank paper to students, draw this table structure on the whiteboard. Ask students to write (on the sheet given) two possible scenarios where each Hadoop file format can be used.

Text files	Scenario 1
	Scenario 2
Sequence files	Scenario 1
	Scenario 2
Avro data files	Scenario 1

	Scenario 2
Parquet files	Scenario 1
	Scenario 2

Collect the sheets back and examine their responses.

DATA SERIALIZATION



CLASS DURATION: 8 MINUTES

VIDEO DURATION: LESSON 6.3

FACILITATION GUIDELINES

Play Video Content: Lesson 6.3

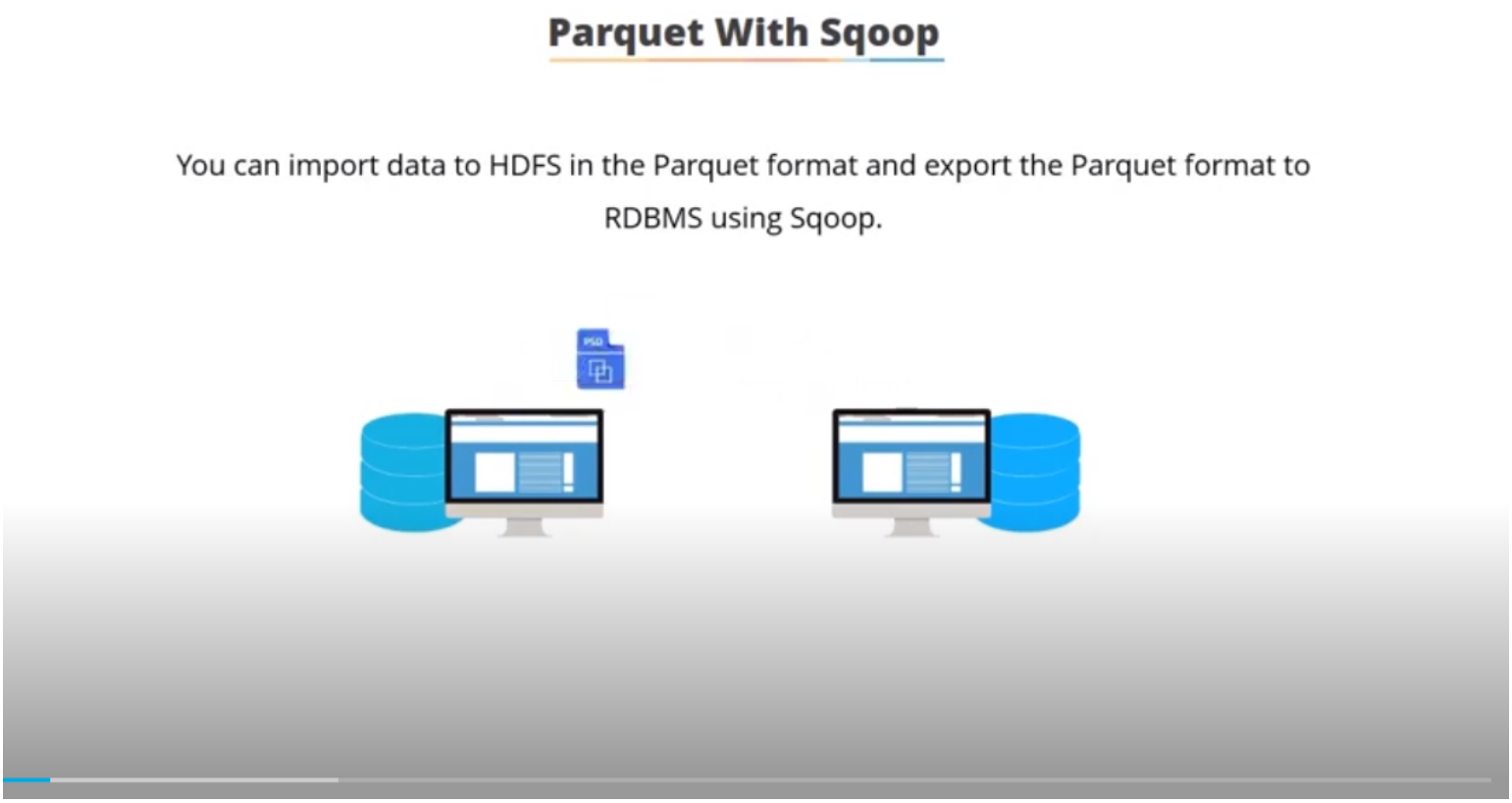
Content: Data serialization is the process of storing structured data in a series of bytes. The data can then be transmitted over a network or written to persistent storage. The reverse form of this process is called data deserialization where the series of bytes is converted back to structured data. Data serialization is important in Hadoop because there are a number of processes communicating with each other in the Hadoop environment. It then becomes important that a common format should be followed so that these processes talk to each other irrespective of the programming language used.

Besides interoperability, data serialization includes the following benefits. There is efficient use of storage space. There is minimal overhead in writing and reading terabytes of data. Data can be read and written in an older format.

Hadoop has its own serialization format called Writables. Most Hadoop MapReduce programs use it. However, it has its limitations as Writables use only Java APIs. Avro data files were designed to overcome this limitation. It is neutral to any language. Programs need not know about schema to process data as schema is already stored in the file.

Reference: Refer to the following links for more information:
https://www.waitingforcode.com/apache-avro/introduction-to-serialization-in-big-data/read
https://databaseline.bitbucket.io/an-overview-of-file-and-serialization-formats-in-hadoop/
Say: Let us move to the next topic.

PARQUET WITH SQOOP



CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 6.5

FACILITATION GUIDELINES

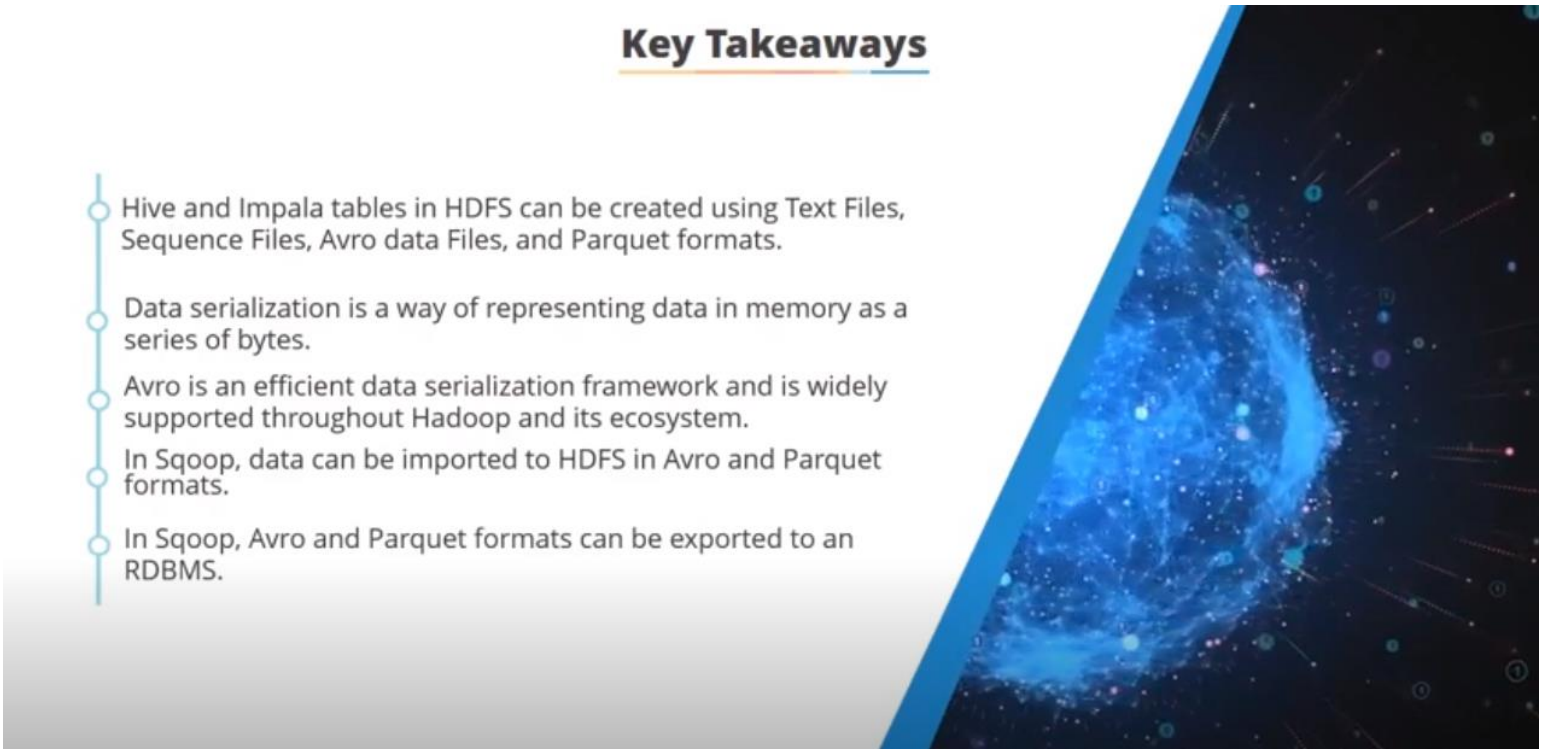
Play Video Content: Lesson 6.5

Say: We learned about Sqoop earlier. When Sqoop extracts data from a relational database, it extracts the raw information. The data is then stored in the appropriate file format, such as Avro or parquet.

Reference: Refer to the following links for more information:
http://davidiscoding.com/inport-and-export-data-with-sqoop-in-hdfs
https://data-flair.training/blogs/sqoop-import/
http://www.riveriq.com/blogs/2019/01/sqoop-import-to-text-avro-parquet-sequence

Say: Let us see the key takeaways from the lesson.

KEY TAKEAWAYS



CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 6.6

FACILITATION GUIDELINES

Play Video Content: Lesson 6.6

Content: Write the key takeaways of the session on the whiteboard. Tell the students that they have learned about different file formats and data serialization.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Play Video Content: Lesson 6.7. Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, continue the video and show the correct answer and explanation. Repeat this process for all the questions. If students have queries regarding the questions, discuss and resolve the queries.

Say: Let me test your understanding now by asking few more questions.

Questions:

1. Which file format stores data in key-value pairs?

Answer: Sequence files

2. What file format is schema dependent?

Answer: Avro

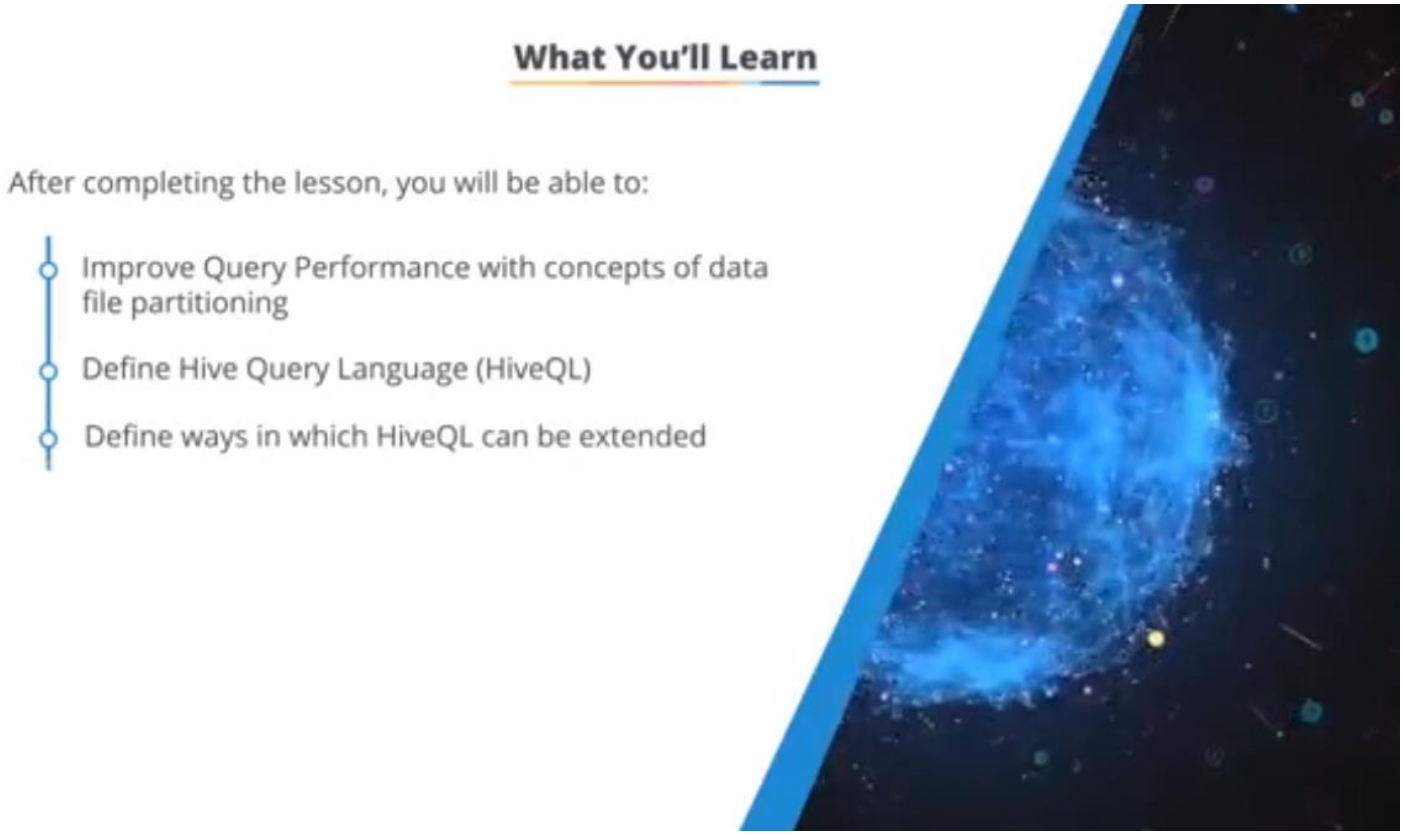
3. What file format is preferable when columns are more?

Answer: Parquet file format

4. Which file format is easily read and written in any programming language?

Answer: Text files

INTRODUCTION



CLASS DURATION: 11 MINUTES

VIDEO DURATION: LESSON 7.1

FACILITATION GUIDELINES

Play Video Content: Lesson 7.1 till 00:47 seconds

Content: After playing the video, repeat all the points and tell the students that in this session they will learn about data file partitioning and how HiveQL can be extended.

Say: Let us begin.

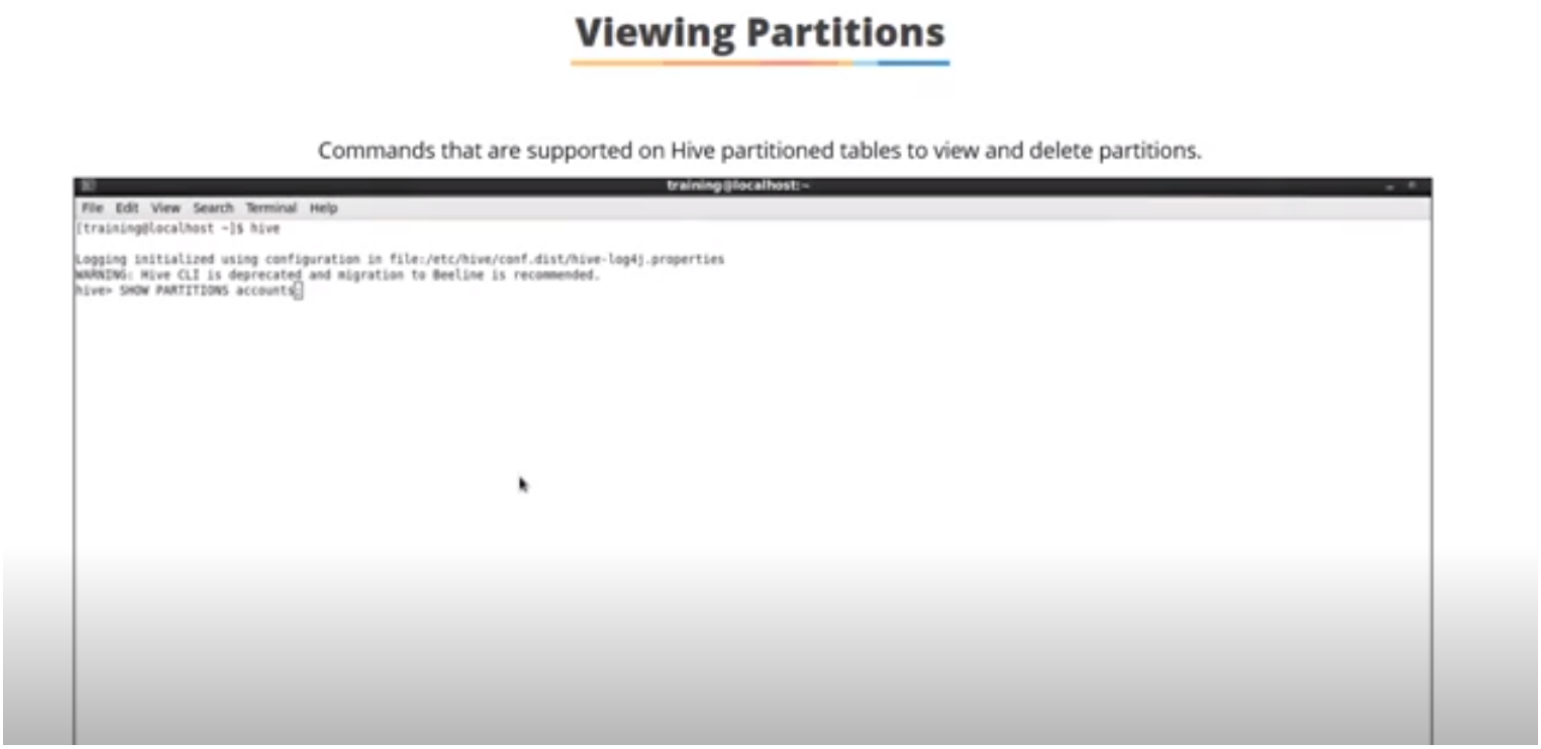
Play Video Content: Resume Lesson 7.1

Say: You all know that Hive is a tool to process structured data in Hadoop. Hive provides an interface to organize and query large amounts of data. It also helps write SQL-like queries. Hive can organize data into tables, partitions, and buckets. Hive can have two types of tables, managed table and external table. A managed table can be created in a specific location in HDFS. If this table is deleted, both the data and metadata will be deleted from HDFS. In an external table, data resides outside Hive. When the table is deleted, only the schema gets deleted and data remains untouched. In a Hive partition, table is divided into small slices. This means the table is partitioned based on the column of a table. This makes the query response time faster to process as the query will look into that particular portion of data only. The table is divided based on partition keys. A table can have more than one partition keys. A partition key determines how the data will be stored in a table. Still if the data remains huge to be queried, these partitioned tables can be further divided into more manageable parts called buckets. Bucketing uses the hash function. Bucketed tables are best used for testing and debugging.

Reference: Refer to the following links for more information:
https://resources.zaloni.com/blog/partitioning-in-hive
https://acadgild.com/blog/bucketing-in-hive
https://www.youtube.com/watch?v=MmJu0GzyNE4

Say: Let us move to the next topic.

OVERVIEW OF THE HIVE QUERY LANGUAGE



CLASS DURATION: 11 MINUTES

VIDEO DURATION: LESSON 7.2

FACILITATION GUIDELINES

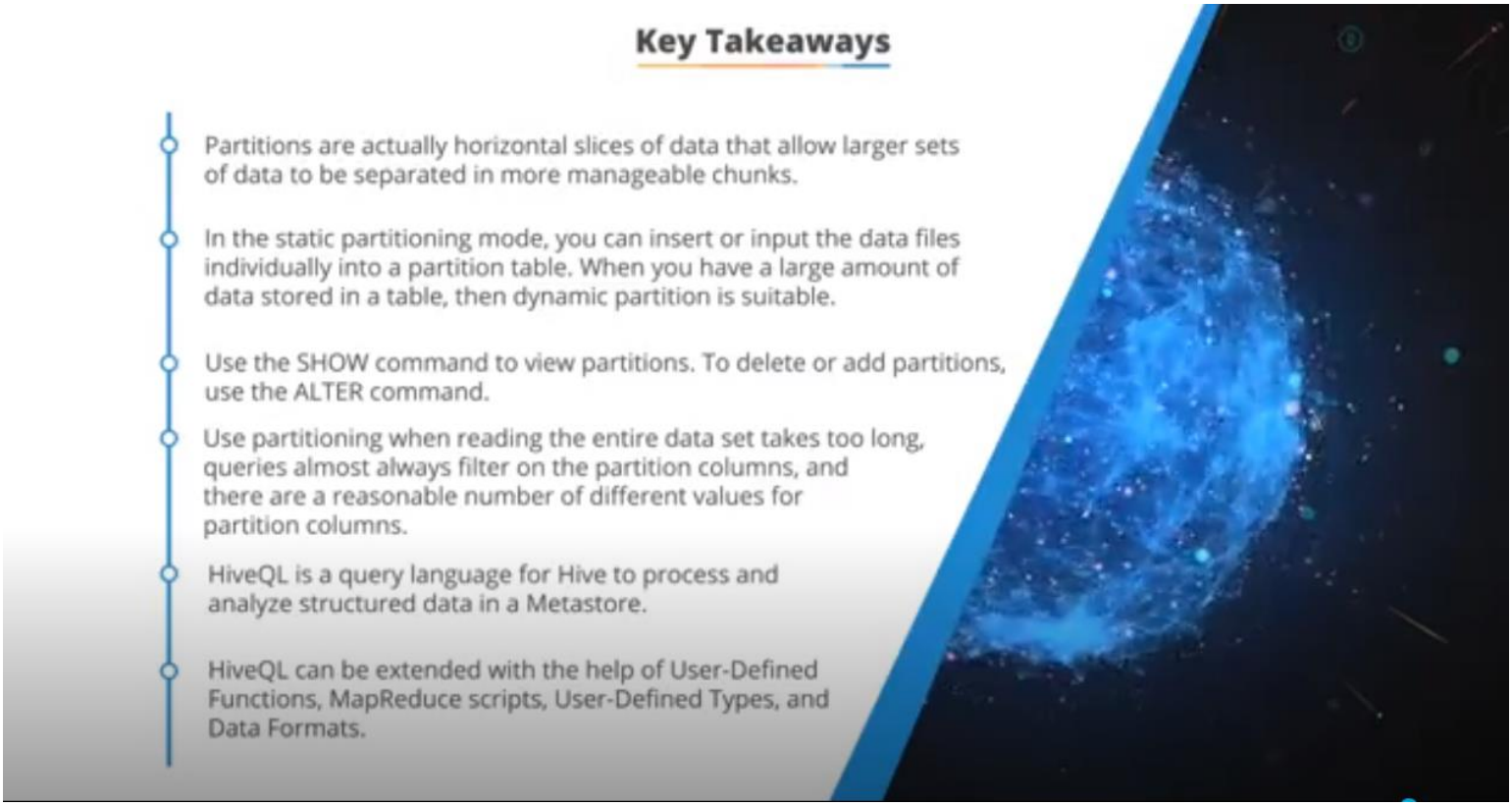
Play Video Content: Lesson 7.2

Content: HiveQL provides SQL type environment to analyze, query, and summarize data. A developer need not know Java or MapReduce to query the data. Besides the built-in functions of Hive, a developer can create User Defined Functions (UDFs). This is because built-in functions are not sufficient at times to meet the requirements. To develop UDFs, developers should know Java as Java classes need to be created.

Reference: Refer to the following links for more information:
https://dzone.com/articles/writing-custom-hive-udf-andudaf
https://www.youtube.com/watch?v=BDbMPfNw_Tc

Say: Let us see the key takeaways from the lesson.

KEY TAKEAWAYS



CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 7.3

FACILITATION GUIDELINES

Play Video Content: Lesson 7.3

Content: Write the key takeaways of the session on the whiteboard. Tell the students that they have learned the basics of HiveQL.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Play Video Content: Lesson 7.4. Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, continue the video and show the correct answer and explanation. Repeat this process for all the questions. If students have queries regarding the questions, discuss and resolve the queries.

Say: Let me test your understanding now by asking few more questions.

Questions:

1. Which is the table when deleted, deletes both the data and metadata from HDFS?

   Answer: Managed table

2. In a Hive partition, how is a table identified?

   Answer: Using partition keys

3. What are the tables called that are created from partitioned tables?

   Answer: Bucketed tables

4. Who can create UDFs in Hive?

   Answer: Developers proficient in Java language.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1:

Save a sample MySQL database from https://www.w3resource.com/sql/sample-database-of-sql-in-mysql-format.txt. Import it in HDFS in Avro format.

Session 7

Session Objectives:

- Explain Apache Flume along with its uses
- Explain the extensibility and scalability features in Flume
- Identify Flume data sources and their flow
- Explain the components in the Flume architecture
- Explain the Flume Agent Configuration file with an example
- Define HBase along with its uses
- Explain the characteristics of HBase
- Analyze the HBase architecture and components
- Discuss HBase data storage and data models
- Differentiate HBase and RDBMS
- Analyze the HBase shell commands

Video Content: Lesson 8

QUICK RECAP

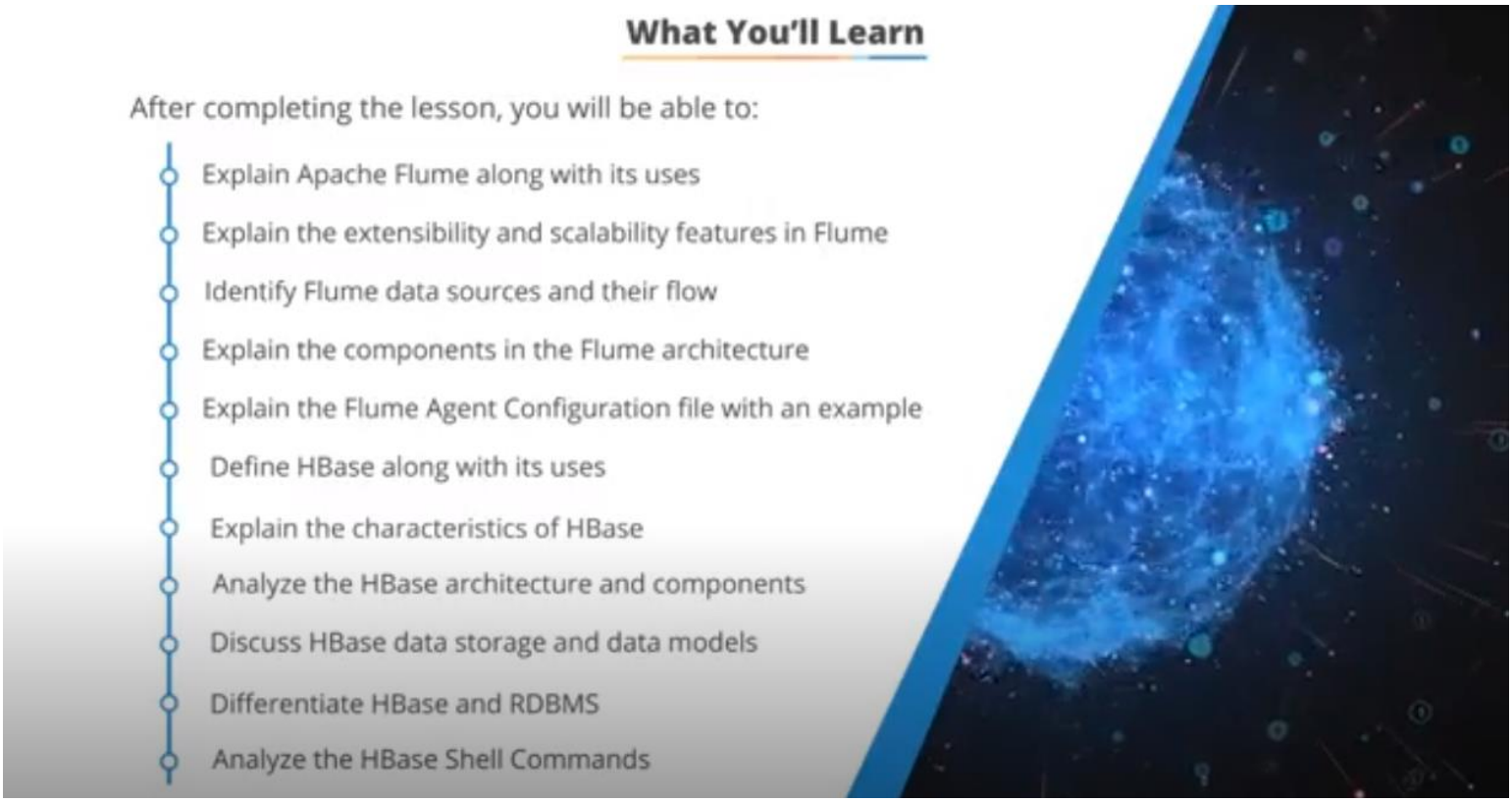
CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

INTRODUCTION



CLASS DURATION: 16 MINUTES

VIDEO DURATION: LESSON 8.1

FACILITATION GUIDELINES

Play Video Content: Lesson 8.1 till 01:00 minutes

Content: After playing the video, repeat all the points and tell the students that in this session they will learn about Apache Flume and HBase.

Play Video Content: Resume Lesson 8.1

Content: You know that all the application servers and Web servers generate log files. These files are crucial to data analysis as companies can use them to know about their customers. Ecommerce companies can use these log files to know about their customers' buying patterns. Essentially, we are talking about big data.

Say: Regarding Facebook and Twitter, data is continuously streaming from their servers. The question is how to get this streaming data into Hadoop? Can Sqoop solve the problem? (Allow students to respond.)

No. This is because Sqoop cannot handle unstructured data. This is where Apache Flume plays an important role. It takes the data from the server and writes the same in Hadoop.

Let us see the process that Flume follows. Consider a Web server as the source of data and Hadoop as the destination. In order to bring data from the source and write the data to Hadoop, we need to configure

a Flume agent. You can configure multiple Flume agents depending on the volume of data. If the data is humongous coming from different Web servers, you might consider configuring multiple Flume agents.

A Flume agent has three components, source, channel, and sink. A Flume agent cannot function without the three components. It is the Flume client that talks to the Web server and asks for data. The Flume client receives the data and sends it to the source. Source is the component that receives the data. It then pushes the data to the channel for the sink component to take the data and write it to Hadoop. Channel is the temporary storage area. It buffers the data until it is used by sink.

Resource: Ask students to refer to the following links to learn more:

https://www.youtube.com/watch?v=cSMhLBmmlUs

https://www.youtube.com/watch?v=OYFi5tBEyf8

https://www.youtube.com/watch?v=PcMHjQpLY2w

Say: Let us do a small activity.

ACTIVITY 1

CLASS DURATION: 24 MINUTES

Discuss the different use cases of Flume.

INTRODUCTION TO HBASE

CLASS DURATION: 25 MINUTES

VIDEO DURATION: LESSON 8.2

FACILITATION GUIDELINES

Play Video Content: Lesson 8.2

Content: HBase is a data store that stores unstructured data. It is a NoSQL database and hence, ideal for real-time data streaming. Companies such as Pinterest and Facebook use HBase for the 'Following' feed and 'Messaging' respectively.

HBase is built over HDFS file system. It can use the MapReduce computational framework to retrieve and store data. It is column family oriented database. Let us understand the architecture of HBase. It is similar to HDFS that follows the master-slave concept. Similar to this, HBase includes HMaster and Region servers. There can be multiple HMasters but only one is active at a time. In HDFS, if a NameNode fails, the whole cluster is inaccessible. In HBase, if a HMaster fails, another HMaster serves.

In the HBase architecture, you will find Zookeeper, HMaster, and region servers (Write this on the whiteboard.) It is the Zookeeper that keeps the HMaster and the region servers together. The region servers run on the data nodes in the HDFS architecture. When data is requested from HBase, the Zookeeper takes the request and routes it to HMaster or the region servers. In the same way, if HMaster wants to talk to any region server, the request must pass through the Zookeeper. Zookeeper has all the information about the region servers and the HMaster in the particular cluster.

Every region server is divided into blocks, Blockcache, MemStore, WAL, and Hfile. Blockcache is a read cache. It gives faster read access. MemStore is a write cache. It stores all the new information that is not written to disk. Write Ahead Log (WAL) is. It is a file where new data is stored and which is not

persisted to a permanent storage. It is used for data recovery. Hfile is the actual storage file. It stores row as a sorted key value on the disk.

There can be multiple regions in a region server that relates to a column family. Consider that you want to create a family called Products. This column family will have Blockcache and MemStore. There can be multiple column family in a region server.

Remember the HDFS architecture, where nodes are scalable. Here in HBase as well, region server can be scaled when required. Any read and write request that comes from the Zookeeper, HMaster routes them to the region servers. HMaster has additional responsibilities as well. It monitors any table that is created. It takes care of any failover of nodes and manages the cluster. If a client requests any changes such as schema change, HMaster updates the same in the region servers.

Reference: Refer to the following links to learn more:

https://www.youtube.com/watch?v=VRD775iqAko
https://www.youtube.com/watch?v=2Ci_QxJ1kiE
https://data-flair.training/blogs/features-of-hbase/

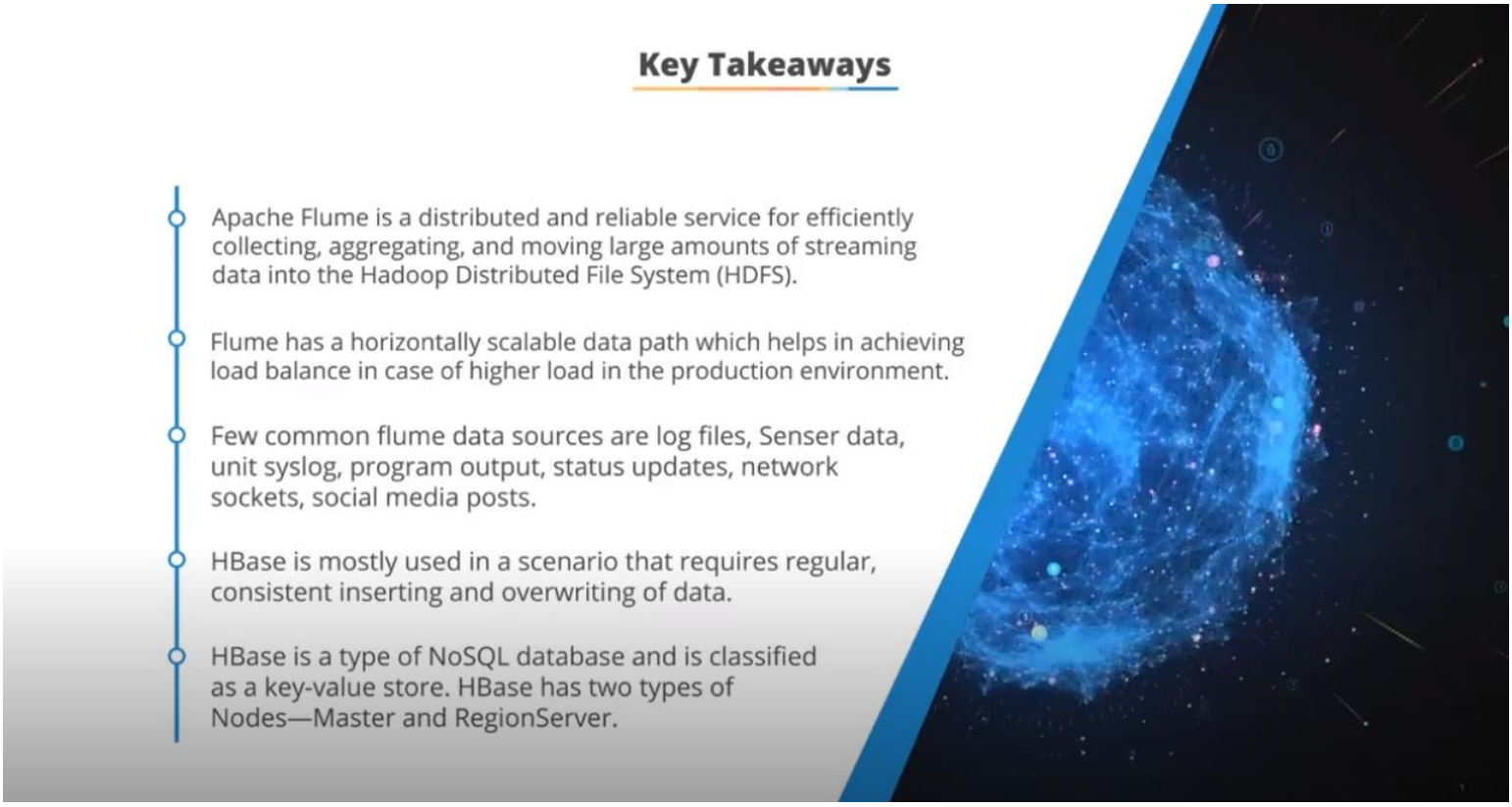
Say: Let us do an activity.

ACTIVITY 2

CLASS DURATION: 24 MINUTES

Divide the class into groups. Ask each group to show how data is processed in HBase using a diagram. The presentation should also include information on how data is stored in HBase as columns.

KEY TAKEAWAYS



CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 8.3

FACILITATION GUIDELINES

Play Video Content: Lesson 8.3

Content: Conclude the class by giving key takeaways of the session. Tell the students that they have learned about Apache Flume and HBase.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Play Video Content: Lesson 8.4. Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, continue the video, and show the correct answer and explanation. Repeat this process for all the questions. If students have queries regarding the questions, discuss and resolve the queries.

Say: Let me test your understanding now by asking few more questions.

Questions:

1. What are the three components of Flume agent?

Answer: Source, Channel, and Sink.

2. What is the function of source?

Answer: It receives the data from the Web or app server.

3. What is channel?

Answer: Channel is the temporary storage area. It buffers the data until it is consumed by sink.

4. What are the three prime components that you will find in an HBase architecture?

Answer: Zookeeper, HMaster, and region servers

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1:

Configure Flume and HBase in your Hadoop system.

Create a configuration file named flume-conf.properties and add appropriate configuration in that file.

Create a directory for Flume from where it can pick up data.

Create a directory in HDFS. Flume will put data here.

Run Flume.

Check the HDFS directory.

Session 8

Session Objectives:

- Explain the concept of Pig
- Understand the types of Data Models supported by Pig
- Differentiate Pig and SQL
- Understand the functionalities to perform Pig script operations
- Understand various commands in Pig

Video Content: Lesson 9

QUICK RECAP

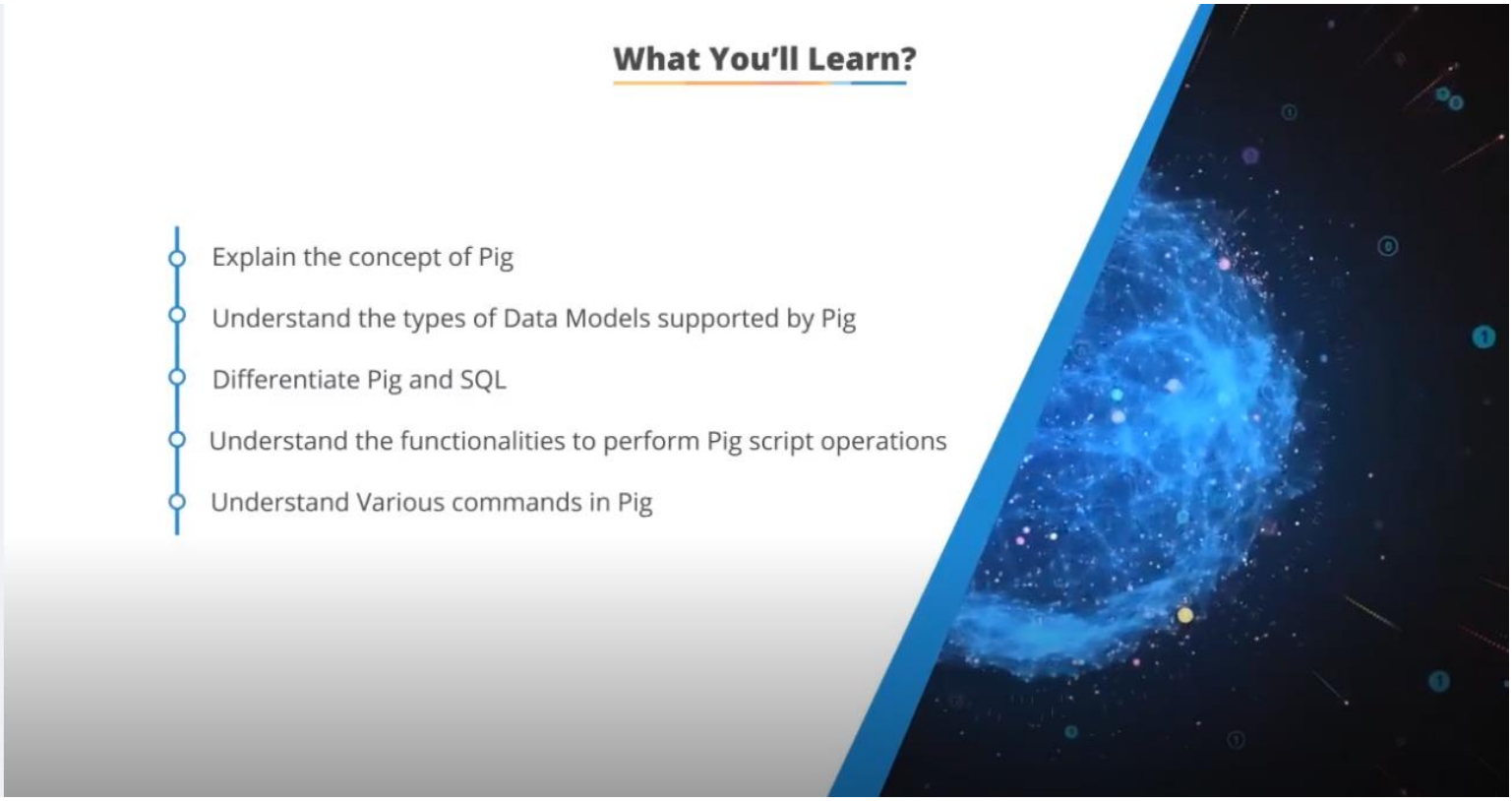
CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

LESSON OBJECTIVES



CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 9.1

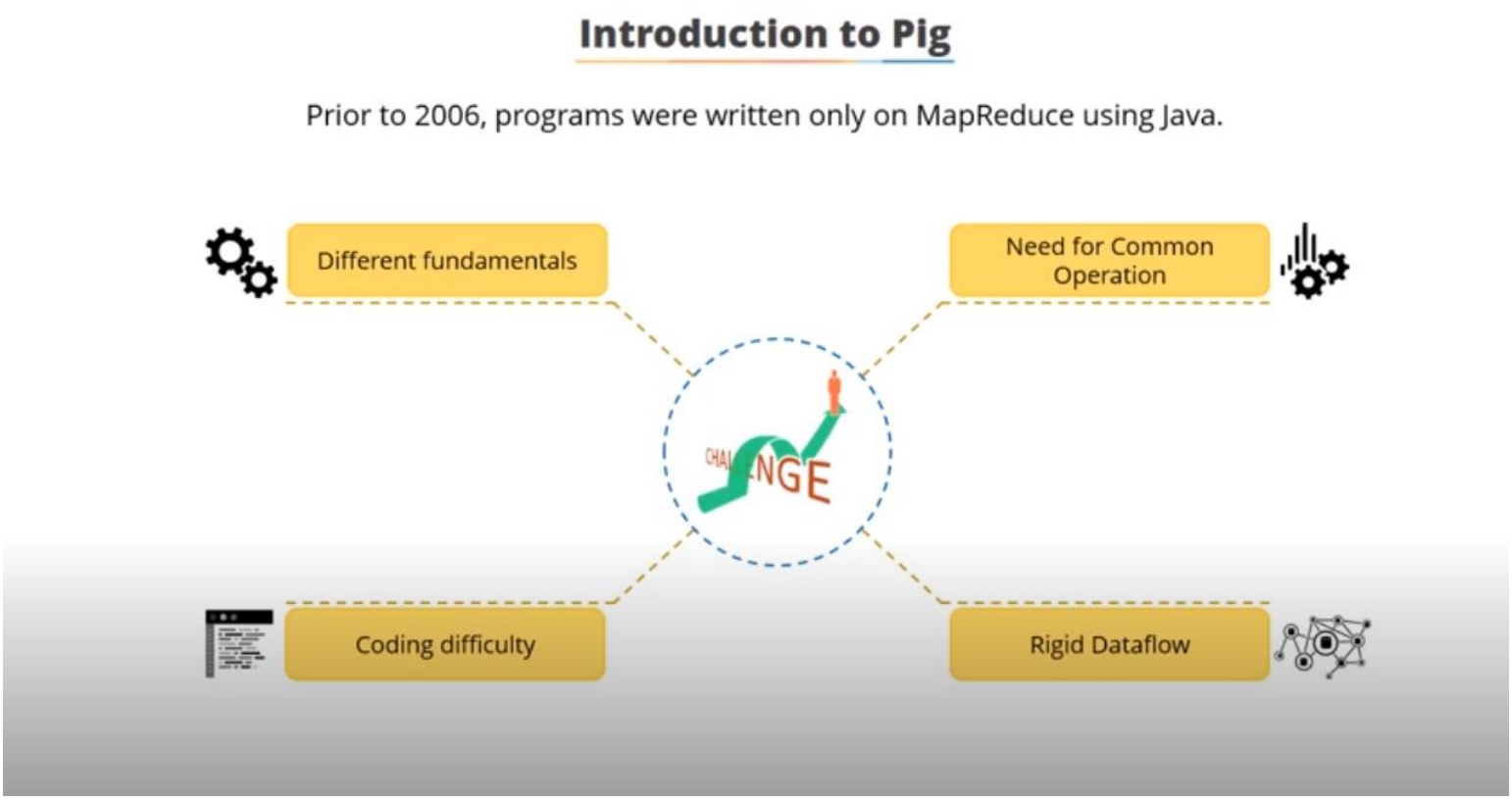
FACILITATION GUIDELINES

Play Video Content: Lesson 9.1

Content: After playing the video, repeat all the points and tell the students that in this session they will learn about Apache Pig.

Say: Let us begin.

INTRODUCTION



CLASS DURATION: 15 MINUTES

VIDEO DURATION: LESSON 9.1

FACILITATION GUIDELINES

Play Video Content: Lesson 9.1

Content: Pig was basically created because, for every MapReduce program, complex Java codes required to be written. This also meant that it was important that developers should know Java. To reduce this production time, Apache Pig was introduced.

Say: Let us now understand features of Apache Pig.

Ask: Ask students if they can tell key characteristics of the animal Pig. (Allow the students to respond.)

Pigs live anywhere and can eat anything. Similar to the eating habit of Pig, Apache Pig can process any type of data such as key/value stores. And just as Pigs live anywhere, Apache Pig is designed to work with any processing framework besides Hadoop. Besides these features, Apache Pig can process the data quickly and can be easily controlled and modified by the users. Developers also need not know Java in order to script in Pig. This makes scripting easy. Pig includes a number of operators. Developers can use them to introduce their own functions to read, write, and process data.

Coming to data types, Pig broadly supports four data types: Atom, Tuple, Bag, and Map. Atom is similar to the data type that you see in most programming languages, such as int and float. Tuple, Bag, and Map are categorized under complex data types. Tuple is an ordered set of fields, such as *(EmployeeID,*

Name_of_the_employee). Bag is a collection of tuples, such as {*(EmployeeID, Name_of_the_employee), (EmployeeID, Name_of_the_employee)}*.

What is a map? Map is a set of key value pairs. A map is a collection of data items and each item has an associated key. For example, area pin code can be a key and the area name can be its value. A hash is used between a key and a value, for example, *['area_code'#'area_name'].*

It is interesting to note that 10 lines of code in Pig are equivalent to 200 lines of code in Java. The language to write Pig programs is Pig Latin. Apache Pig includes two important components: Pig Latin programming language and Pig runtime environment. It is in this environment that Pig Latin programs are executed. In every Pig program, data is loaded, transformed, and dumped.

Do you know, the popular professional networking site LinkedIn includes features such as *Who Viewed My Profile*? It is Apache Pig, which is used to implement such features.

Reference: Refer to following links to know more:

https://beyondcorner.com/learn-apache-pig-tutorials/apache-pig-architecture/
https://www.dummies.com/programming/big-data/hadoop/the-pig-architecture-in-hadoop/
https://www.quora.com/What-is-the-internal-architecture-of-Apache-Pig
https://www.youtube.com/watch?v=YDmiC5yyWsE
https://www.youtube.com/watch?v=JFwgM247lPk

Say: You will now perform an activity.

ACTIVITY 1

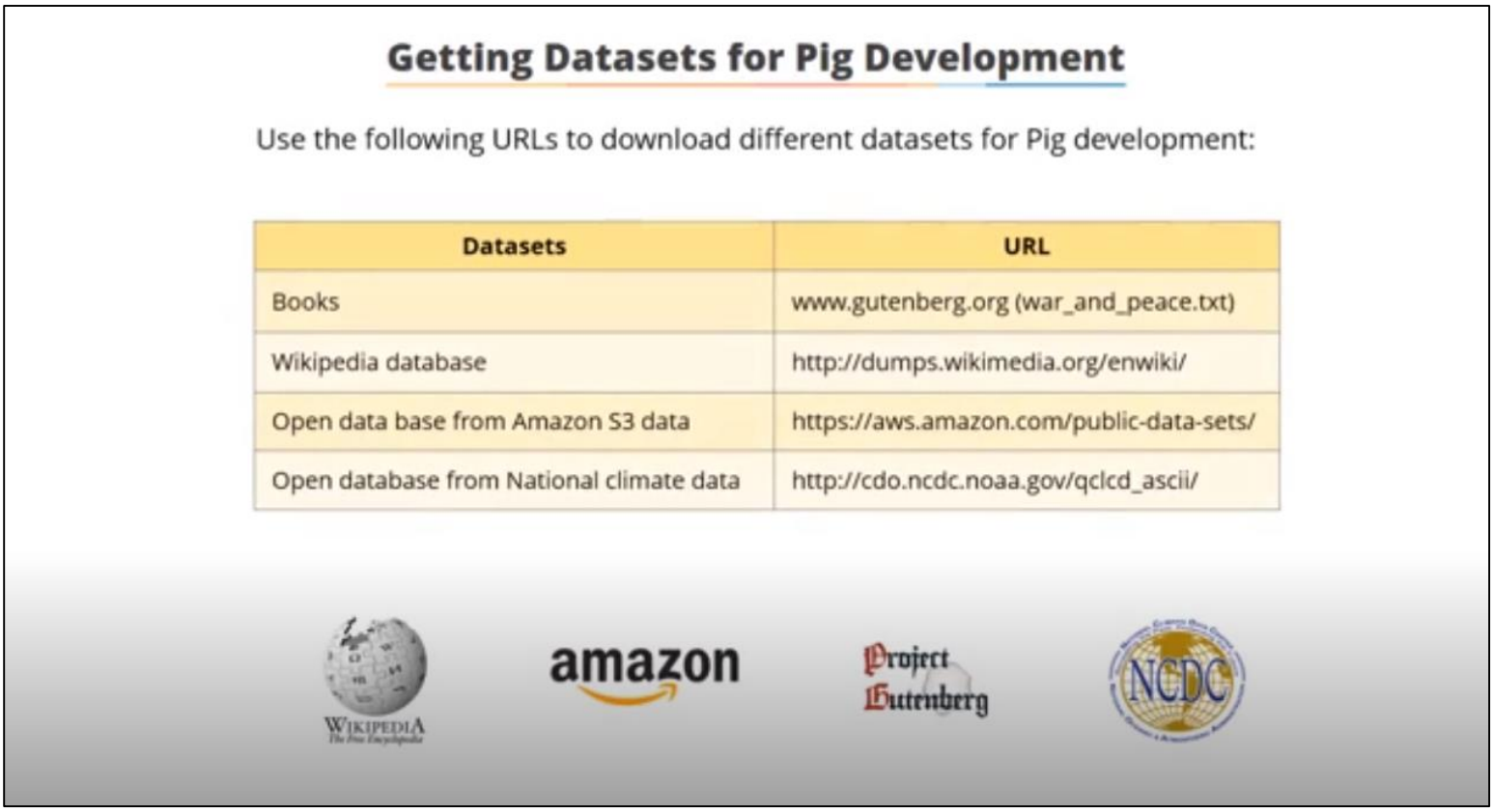
CLASS DURATION: 30 MINUTES

Pass sheets of blank paper to the entire class. Say, Apache Pig can be used for ETL Data Pipeline and iterative processing. Ask students to identify few real-life examples that suit each. Ask them to use the table format to fill in the examples. Once they finish, ask any two students to read aloud the results.

ETL Data Pipeline	Example 1
	Example 2
	Example 3
	Example 4
Iterative Processing	Example 1
	Example 2

	Example 3
	Example 4

GETTING DATSETS FOR PIG DEVELOPMENT



CLASS DURATION: 11 MINUTES

VIDEO DURATION: LESSON 9.2

FACILITATION GUIDELINES

Play Video Content: Lesson 9.2

Content: Remember that the Pig Latin script is converted into MapReduce jobs. Pig scripts, parser, optimizer, compiler, and execution engine are components in an Apache Pig execution environment.

Pig scripts are submitted to Apache Pig execution environment. The parser performs semantics checks and checks syntax of the script, and then parses each statement into a logical operator. Next, it outputs a Directed Acyclic Graph (DAG) as the logical plan. A logical plan is a collection of logical operators in the script. This plan is then passed to the optimizer. It performs the optimization activities such as merging and transforming data to reduce the amount of data processing. The compiler then converts the optimized logical plan into a physical plan that is a series of MapReduce jobs. This physical plan includes the physical operators that the Pig will use to execute the script. The execution engine, finally, executes the MapReduce jobs. The required result is then displayed.

Note that loading and storing functions are resolved in the physical plan.

Reference: Refer to following links to learn more:

https://data-flair.training/blogs/pig-architecture/

https://www.commonlounge.com/discussion/1c1d5878fd5041f38a12ba1f7dbd54f3

Say: Let us do another activity.

ACTIVITY 2

CLASS DURATION: 30 MINUTES

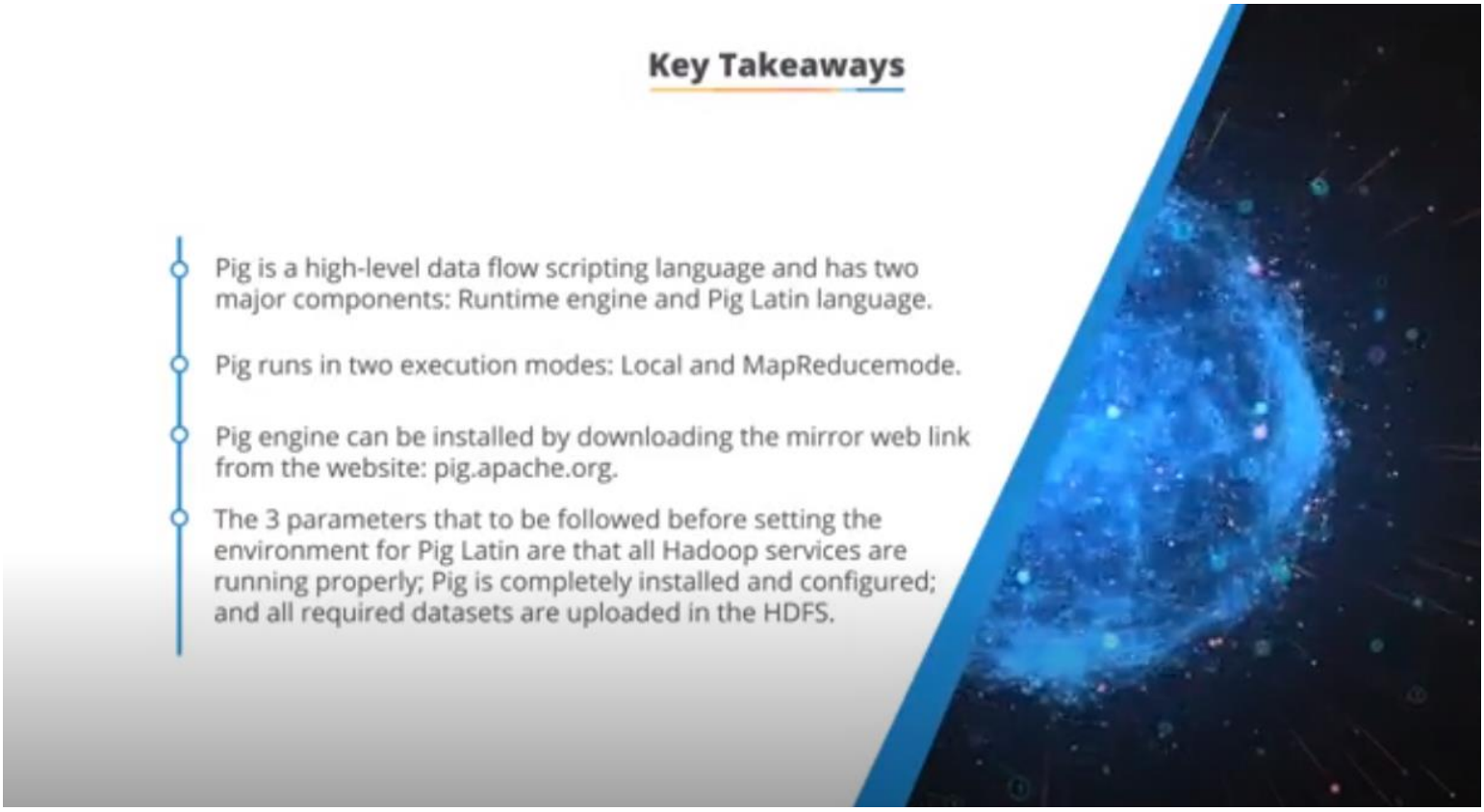
Do you know Apache Pig is used by Healthcare Industry also?

Identify the areas where Apache Pig is beneficial to the industry. Describe in details and present it to the class.

Self-study Activity:

Research on different components of Apache Pig architecture.

KEY TAKEAWAYS



CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 10.5

FACILITATION GUIDELINES

Play Video Content: Lesson 10.5

Content: Write the key takeaways of the session on the whiteboard. Tell the students that they have learned about Apache Pig.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Play Video Content: Lesson 9.4. Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, continue the video and show the correct answer and explanation. Repeat this process for all the questions. If students have queries regarding the questions, discuss and resolve the queries.

Say: Let me test your understanding now by asking few more questions.

Additional Questions:

1. What is a tuple?

Answer: It is a data type. It represents an ordered set of fields.

2. What are the four data types that Apache Pig supports?

Answer: Atom, Tuple, Bag, and Map

3. What is a map?

Answer: A map is a collection of data items and each item has an associated key.

4. What are two important components of Apache Pig?

Answer: The two important components are Pig Latin programming language and Pig runtime environment.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1:

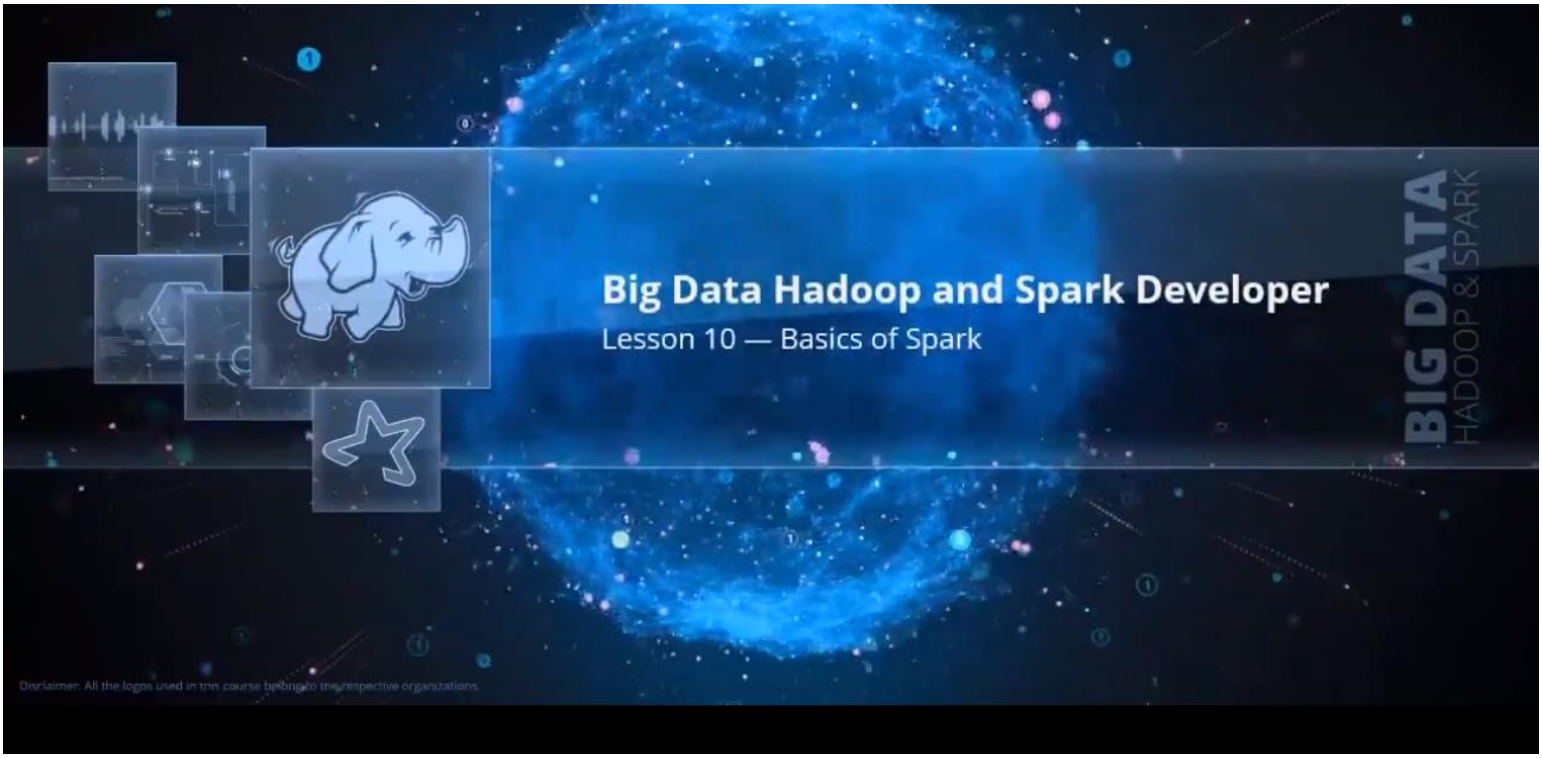
You have two datasets:

Student: This dataset contains name and roll number of students in a class.

Results: This dataset contains roll number and result (Fail or Pass) of students.

Write a Pig script to analyze the given datasets and print the student names who have successfully cleared the exam. You can only use Pig commands to achieve the desired result.

Session 9



Session Objectives:

- Describe the limitations of MapReduce in Hadoop
- Explain Spark, its architecture, and its advantages
- Understand Resilient Distributed Dataset Operations
- Compare Spark with MapReduce
- Understand functional programming in Spark

Video Content: Lesson 10

QUICK RECAP

CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

LESSON OBJECTIVES

CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 10.1

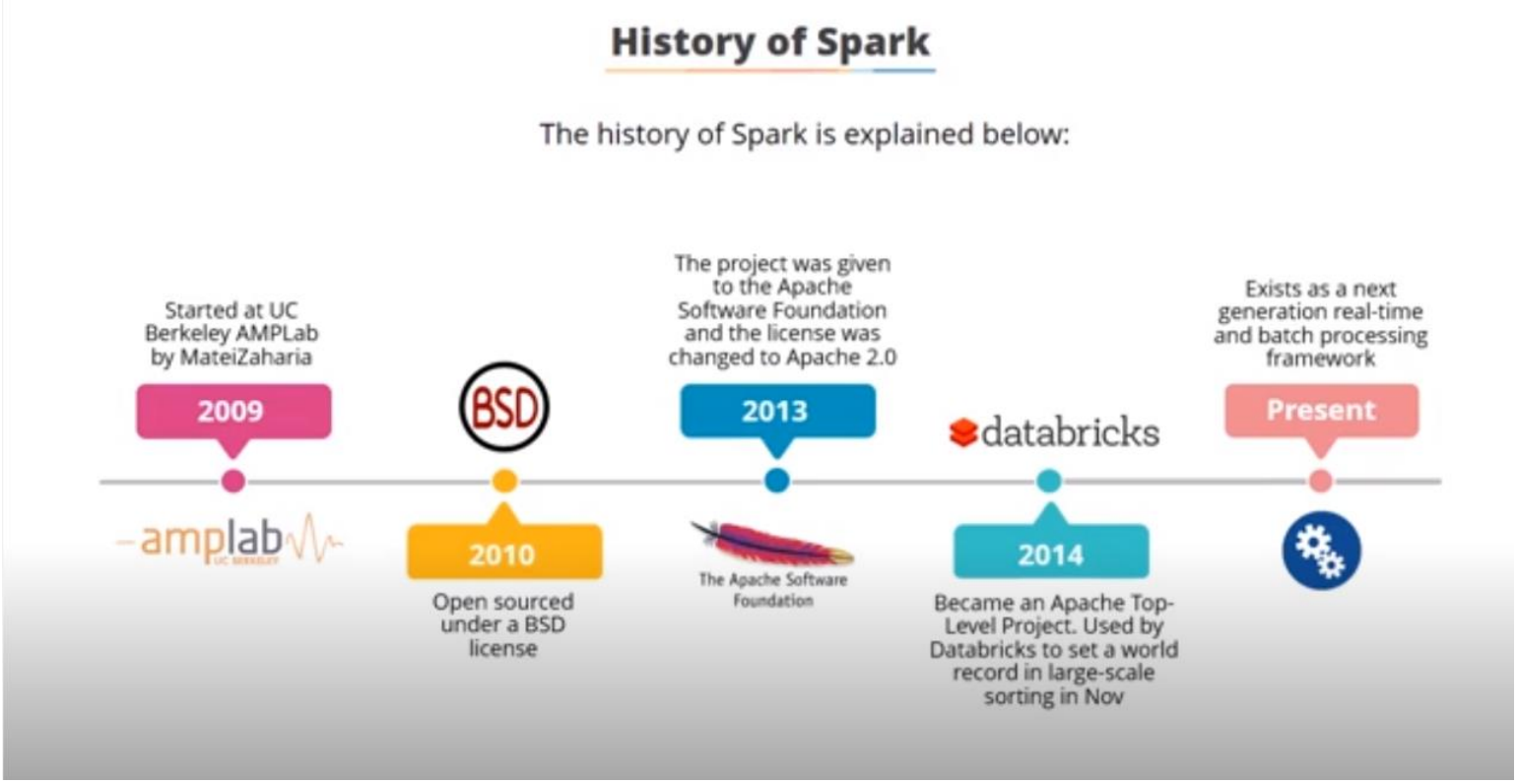
FACILITATION GUIDELINES

Play Video Content: Lesson 10.1

Content: After playing the video, repeat all the points and tell students that in this session they will learn about Apache Spark and Resilient Distributed Dataset Operations (RDD) in Spark.

Say: Let us begin.

BASICS OF APACHE SPARK



CLASS DURATION: 25 MINUTES

VIDEO DURATION: LESSON 10.1

FACILITATION GUIDELINES

Play Video Content: Lesson 10.1

Ask: Ask students what is meant by real-time and batch processing. (Give them time to respond.)

Their response may match with the following:

Batch Processing: It refers to the technique of processing or working with data in batches. This method ingests data in particular batches and then implements further transformations on it without human intervention. It supports data collection, entry, processing and generating results in batches. Hence, it works in an offline mode. Example: Customer service data, ATM data, and so on.

It is used in Hadoop ecosystem as well.

Real-time Processing: It deals with frequent data ingestion (within seconds) and gives responses within stringent constraints. Processing starts as data inputs and responses are generated continuously, thus making it a continuous stream of input→processing→output. Example: traffic control systems.

Content: So now the question arises, with the existence of MapReduce, why was Spark discovered?

It is because MapReduce only allows batch processing whereas spark allows both batch and real time processing. These days with the increasing complexity in data, real-time processing has become a necessity. Apart from this limitation, let us look at some more limitations of Hadoop's MapReduce over Spark.

- ***Unsuitable for trivial operations***: Since we know that MapReduce stores data in a key-value format, performing filter and join operations on data increases the complexity of the task. This is because you might have to rewrite the operations repeatedly on the entire data. On the other hand, Spark allows simple filter and join operation implementation on a particular subset of data.
- ***Unfit for large data on network***: MapReduce does not allow data shuffling (shuffling of data for performing different operations). This is because it works on the locality principle. Data locality refers to ability to move computation close to where actual data resides on the node, instead of moving large data to computation. Therefore, it is unable to handle large data on a network.
- ***Unsuitable with Online Transaction Processing (OLTP)***: OLTP is a method of processing data generated from online transactions. We know that this kind of data keeps on generating each second (real-time), therefore, MapReduce is unable to deal with it.
- ***Unfit for processing graphs***: We know that graphs increase understanding of any data, and MapReduce does not provide any package or library for plotting graphs, whereas Spark does.
- ***Unfit for iterative execution***: Many machine learning algorithms requires iterative steps to arrive at a decision. MapReduce does not allow this feature, whereas Spark itself has its own library for machine learning algorithms.

Let us learn about Spark in detail.

Spark is an open source library for cluster computing framework. (Show this image on the whiteboard and explain further.)

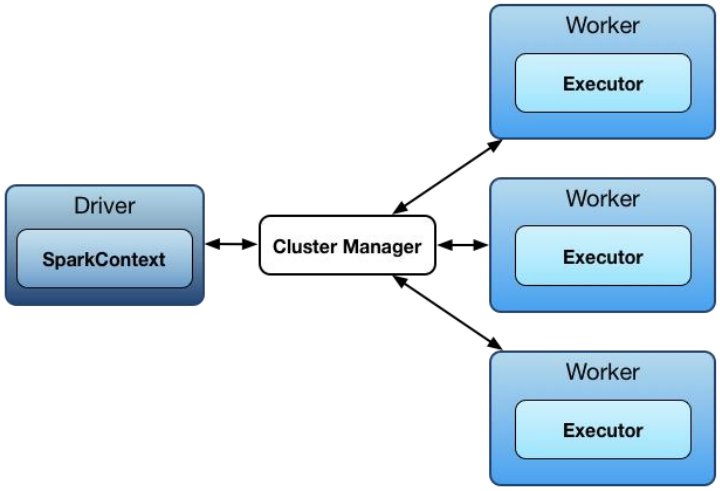


Image courtesy: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/spark-architecture.html

Spark receives data and stores it on a cluster, which has petabytes of memory space (even more than that). It then divides the data into several worker nodes to perform operations on the data. This kind of work division is called Data Distribution. Here, the number of worker nodes is specified by the user, otherwise Spark sets its default number. Data is processed in these worker nodes and results are sent back to the cluster, which in turn sends it to Spark. Spark then displays the output.

Data distribution method provides 100 times faster performance. This results in huge time saving. It is suitable for machine learning algorithms as it allows querying over data repeatedly.

Spark has five major components:

Spark Core and Resilient Distributed Datasets (RDD), Spark SQL, Spark Streaming, Machine Learning Library (MLlib), and GraphX. Let us put some emphasis on each of them.

Spark Core and Resilient Distributed Datasets (RDD)	RDD is a fundamental data structure of Spark. RDD works on parallel computing. Let us explore this in detail. We have a cluster of computers. One computer acts as master and three other computers as slaves. Master controls the slaves. Slaves do all the work and store all data. (Discuss the analogy given following this list).
Spark SQL	Spark SQL is a structured/semi-structured data structure of Spark which follows a particular schema, such as dataframes. A dataframe has a schema similar to a database. What the schema describes is name and type of each one of the columns. A dataframe is nothing more than an RDD of rows, including some high-level information that is the schema. A **database schema** is the skeleton structure that represents logical view of entire **database**. It defines how data is organized and how relations among them are associated. Ask students to read more about Spark SQL through this link: https://databricks.com/glossary/what-is-spark-sql
Spark Streaming	This component allows Spark to process real-time streaming data. It manipulates data stream with a particular set of API. API is an intermediary, which allows two applications to talk to each other. It allows users to understand the work and switch through the applications with API that manipulate the data and give real-time outcomes.
MLlib	This component of Spark consists of wide variety of machine learning algorithms.
GraphX	Spark also comes up with a library to plot and manipulate graphs and perform computations. Similar to Spark Streaming, GraphX also extends Spark RDD API, which creates a directed graph.

Analogy: Point to the diagram on the whiteboard. The head is Spark driver, which runs on the head node. Basically, it is the program that you wrote on Spark and this executes the main part of your code. The cluster manager runs on the head node and then, we have several worker nodes.

Consider a car rental company. You first action may be to call 1800 to talk with the general representative who tells you what cars are available. You go to the person that actually runs the particular agency and has cars to rent to you. This person will actually give you the car, which is the executor part in the diagram. What you are really looking for is the car but you have to go through all of these steps because they are part of the process and run in parallel.

The worker part is analogous to the local agency that knows how many cars they have. In Spark, the workers know how much CPU or memory they have on that machine. Each worker runs on one of the cores. If you have a multi-core machine, you will typically have one worker per core, but you can have more or less. RDDs are partitioned across workers and workers manage these partitions and executors. These workers are analogous to cars and executers are machine parts of the car that help move the car from one place to another. The executors operate on tasks given by workers.

SparkContext allows an application to enter into Spark infrastructure. Worker nodes manage resources in a single slave machine or in a single slave core. Worker nodes communicate with the cluster manager telling them about the amount of work done, and executers execute the task.

Spark is an in-memory processing architecture. The entire information is loaded into a memory, eliminating the requirement for indexes, schemas, and so on. It also uses compression algorithms to reduce the size of the data thus, saving the space. Caching is another feature of in-memory processing.

It refers to the process of storing data in cache for future use. It duplicates frequently accessed or important data in such a way that it can be accessed very fast thus saving time. Spark is also popular in its performance over MapReduce. It supports various languages such as Java, Python, Scala, and R and allows you to define in-line functions (functions which are not defined explicitly rather defined within the code). This reduces code length and complexity.

Reference: https://www.educba.com/mapreduce-vs-apache-spark/

Say: It is time for us to move to the next topic.

SPARK ARCHITECTURE EXECUTION AND RELATED CONCEPTS

CLASS DURATION: 15 MINUTES
VIDEO DURATION: LESSON 10.2
FACILITATION GUIDELINES

Play Video Content: Lesson 10.2

Content: Spark architecture includes a framework for launch of its application to the output from the application. Let us talk about components of Spark.

Spark Submit Script

It is the most vital component of the spark architecture because it manages Spark applications.

Spark Applications

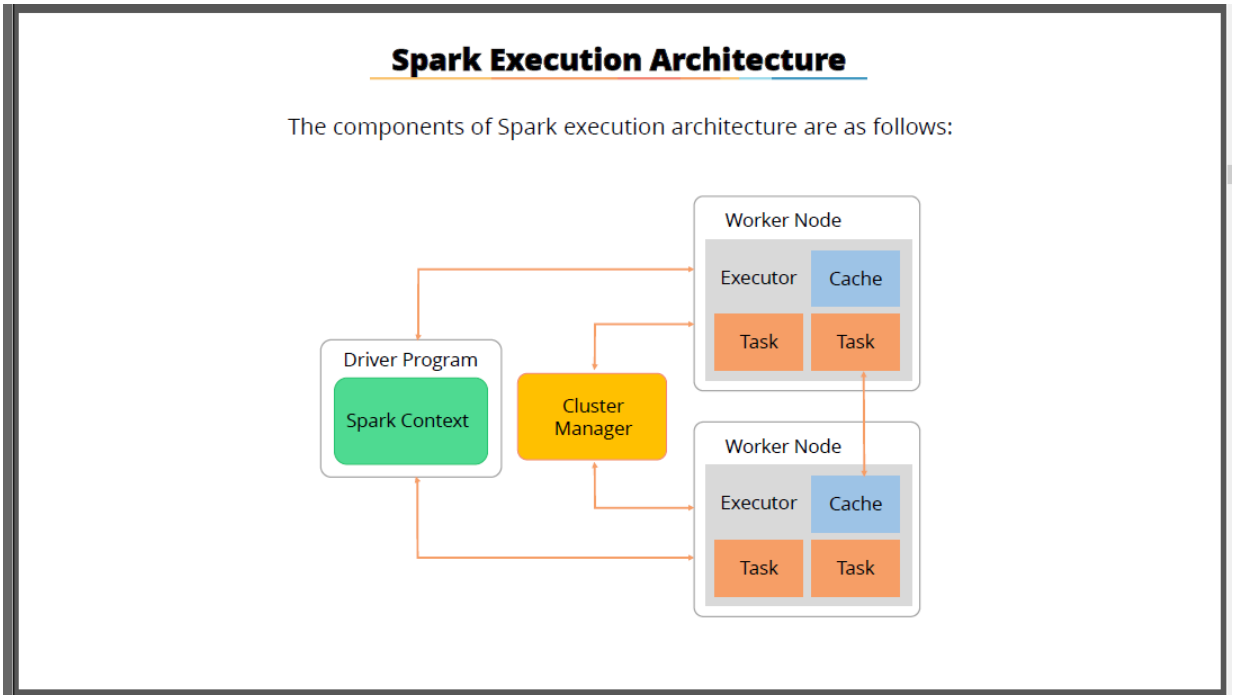
It runs on user-supplied code to get results. A Spark application performs complex tasks, such as completing a single task, an interactive session with multiple tasks, and running processes on its behalf even if no job is running, and so on.

Cluster Managers

Consider a monitor of a class, who is assigned the duty to manage the whole class and assigns tasks to different groups in a class. A cluster manager is similar to a monitor. It dispatches work for all clusters and provides useful resources for clusters. There are mainly three types of cluster managers; Spark Standalone (inbuilt with Spark), Apache Mesos, and Hadoop YARN.
Reference:https://data-flair.training/blogs/apache-spark-cluster-managers-tutorial/

Spark EC2 Launch Script

It launches, manages, and shuts down Spark clusters on Amazon Web Services (AWS) clusters. It automatically sets the Spark and HDFS on the cluster.



Use this analogy to explain.

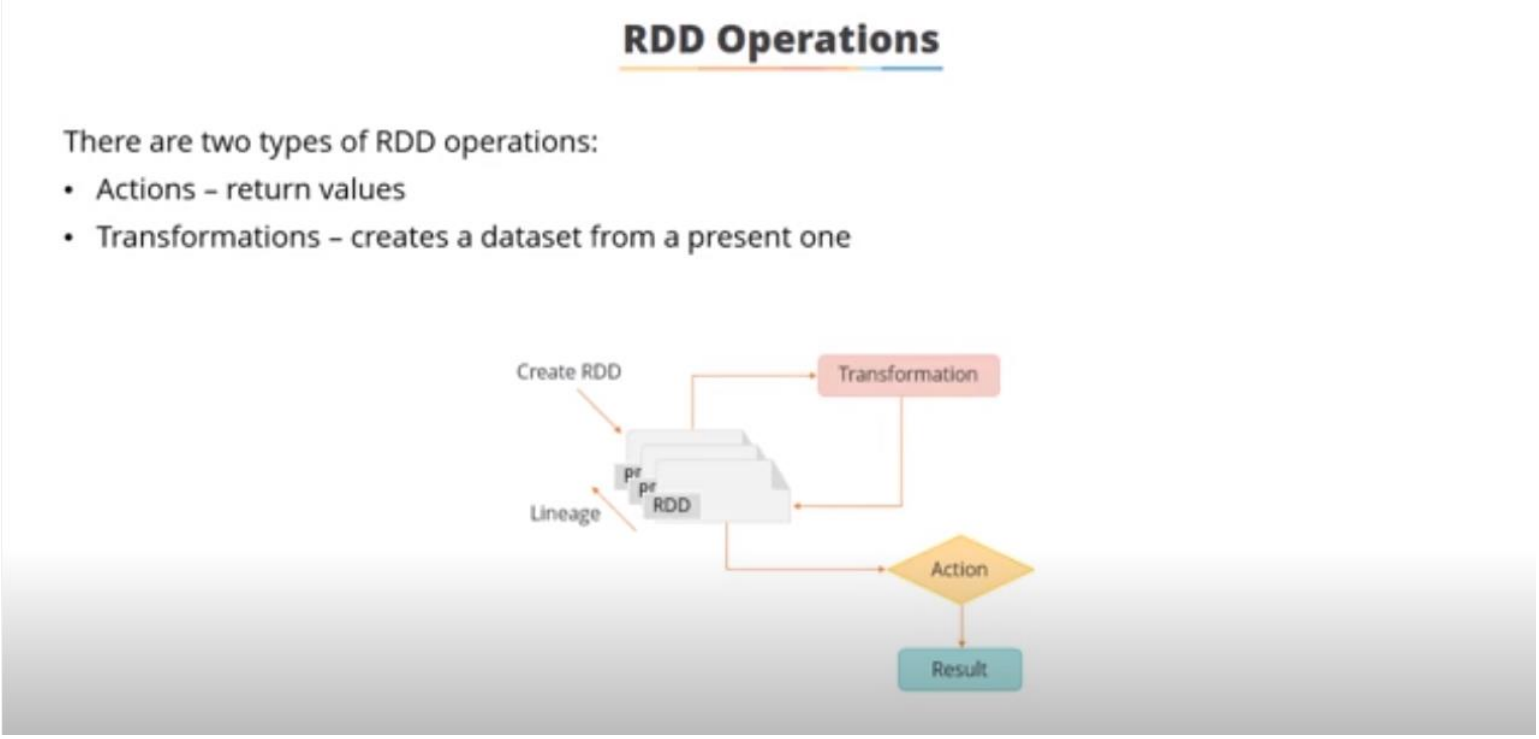
Analogy: Imagine a class where your class teacher is the Spark context, the monitor of the class is the cluster manager and worker nodes are students in the class. Your class teacher is your entry level, without her permission, you cannot enter the class and same is the work of Spark Context. It is the heart of the Spark environment and entry level for Spark execution. Now, the monitor of the class has a duty to manage the whole class and assign different tasks to students and provide the required resources to the students, same is the functioning of the Cluster manager. It supervises all worker nodes and checks for requirements and fulfills them. The students of the class are actual people who complete tasks given and report to the monitor. Similar to this, the worker node consists of executor, who executes the task, and reports it back to the cluster manager.

Spark jobs are done in a parallel way to reduce the complexity of a task and save time. The task is distributed in a parallel way to the worker nodes by the cluster manager so that one node does not get the entire burden. The series of individual tasks is expressed as a single program flow, which allows to parallelize the flow of operators automatically without any human intervention.

Reference: https://data-flair.training/blogs/how-apache-spark-works/

Say: RDD is the fundamental unit of Spark that stores the data and allows to apply functions on it to find the result. In the next topic, we will study about RDD and the operations performed on an RDD.

RDD OPERATIONS



CLASS DURATION: 17 MINUTES

VIDEO DURATION: LESSON 10.3

FACILITATION GUIDELINES

Play Video Content: Lesson 10.3

Content: We shall learn to initialize an RDD in both Python and Scala. RDD can be initialized in many ways. We can make an RDD from any external data, from data in a memory, or from an existing RDD. Let us now look at the syntax.

The data required for this is: 'learn.txt'.

Write the commands to initialize RDD in Python and Scala on the whiteboard and explain to them.

Python	Scala
`mydata = sc.textFile("learn.txt")` **Description**: Here `mydata` is the variable or name of the RDD, `sc.textfile` is the function to initialize the RDD, and `learn.txt` is the name of an external file from which we want to make the RDD.	`val mydata = sc.textFile("learn.txt")` **Description**: Here `mydata` is the variable or name of the RDD, `val` is written before initialization of every new variable, `sc.textfile` is the function to initialize the RDD, and `learn.txt` is the name of an external file from which we want to make the RDD.

An RDD operation has two main components: Action and transformation. A **Transformation** is a function that produces new **RDD** from the existing RDDs. When we want to work with the actual dataset, at that point **Action** is performed. When the action is triggered after the result, no new RDD is formed.

Some common actions performed on RDD are as follows:

count(): returns the number of elements in an RDD

take(n): returns an array of first n elements

collect(): returns a list of all elements of an RDD

saveAsTextFile(): saves RDD as a text file

Reference: https://spark.apache.org/docs/1.6.1/programming-guide.html#actions

Some common transformations performed on RDD are:

map(): applies a function to each record of an RDD and saves the result to a new RDD

filter(): filter elements according to some condition and includes/excludes the elements to the new RDD

Let us look at certain syntaxes here: (Write this on the whiteboard)

Python	Scala
`map(lambda line: line.upper())` **Description**: Here, `map()` is used to apply `upper()` function to the RDD created, `line` depicts each line of RDD. It converts all the lines to uppercase.	`map(line => line.toUpperCase)` **Description**: Here `map()` is used to apply `upper()` function to the RDD created, `line` depicts each line of RDD. It converts all the lines to uppercase.
`filter(lambda line: line.startswith('I'))` **Description**: Here, `filter()` is applied to filter those lines which starts with letter 'I'.	`filter(line => line.startsWith('I'))` **Description**: Here, `filter()` is applied to filter those lines which starts with letter 'I'.

Output of both the codes in Python and Scala is same. (Show the output to the students.)

```
IT IS BASED IN SAN FRANCISCO.
IT HAS TRAINED 450,000+ CUSTOMERS.
IT OFFERS 400+ PROFESSIONAL COURSES.
```

Reference: https://spark.apache.org/docs/1.6.1/programming-guide.html#transformations

One important term while performing operations on RDD is Lazy Evaluation. RDD works on the technique on Lazy Evaluation, which means that execution on RDD would not start until an action is triggered. Let us understand it more clearly with an example: (Write this code on the whiteboard.)

Consider the following code:

```
1. val mydata= sc.textFile('learn.txt')
2. val mydata_uc= mydata.map(line => line.toUpperCase())
3. val mydata_filter = mydata_uc.filter(line =>line.startWith("I"))
4. mydata_filter.count()
```

Explanation:

1. This creates an RDD `mydata` and links it to the file `learn.txt` but does not reads it.
2. This creates an empty RDD `mydata_uc` and links it to the previous RDD, but does not perform any operation.
3. This creates an empty RDD `mydata_filter` and links it to the RDD made in step 2, but does not perform any operation.
4. This processes the data steps mentioned in 1, 2 and 3 and outputs the result of counting the number of elements in the `mydata_filter` RDD.

We see that RDD performed all the operations only when the action `count()` was triggered in the last step.

To reduce the length and complexity of the code, it is not always necessary to make a new RDD in every step and store the results of the previous performed operations in it. Rather, we can use Pipelined RDD to write the code in a single line. Let us see an example.

(Write the code on the whiteboard.)

```
val mydata= sc.textFile('learn.txt').map(line => line.toUpperCase()).filter(line =>line.startWith("I")).count()
```

Here, instead of writing each function in different lines of codes and assigning them to separate RDD, we have used only one RDD `mydata` and have merged all the codes with '.' symbol. This is called pipelined RDD.

References: https://www.youtube.com/watch?v=NRo8TluH7KI

https://www.youtube.com/watch?v=9MeMWdILl5Q

Say: It is time for us to move to next topic, Functional Programming in Spark.

FUNCTIONAL PROGRAMMING IN SPARK

CLASS DURATION: 15 MINUTES

VIDEO DURATION: LESSON 10.4

FACILITATION GUIDELINES

Play Video Content: Lesson 10.4

Content: RDD can work with two types of function definitions:

- Function defined separately
- Function defined as anonymous functions inside the RDD transformation functions

We will study the syntax of both the ways separately.

Function defined separately:

Python	Scala

`def toUpper(s):`
`  return s.upper()`
`mydata= sc.textFile("lines.txt")`
`mydata.map(toUpper).take(2)`
 	
Description: Here, `toUpper(s)` function returns the string in uppercase. The next line indicates the initialization of the RDD `mydata`. Third line uses the `map` function, which takes the argument
	`def toUpper(s:String): String= {s.toUpperCase}`
	`mydata= sc.textFile("lines.txt")`
	`mydata.map(toUpper).take(2)`
	
	Description:
	Here, `toUpperCase()` function returns the string in uppercase. The next line indicates initialization of the RDD `mydata`. Third line uses the `map` function, which takes the argument as the

as the function defined in first line. This means, it maps each line of the `mydata` RDD using the `toUpper()` function and returns the lines in uppercase. `take(2)` prints the first two elements of the resultant RDD.	function defined in first line. This means, it maps each line of the `mydata` RDD using the `toUpper()` function and returns the lines in uppercase. `take(2)` prints the first two elements of the resultant RDD.

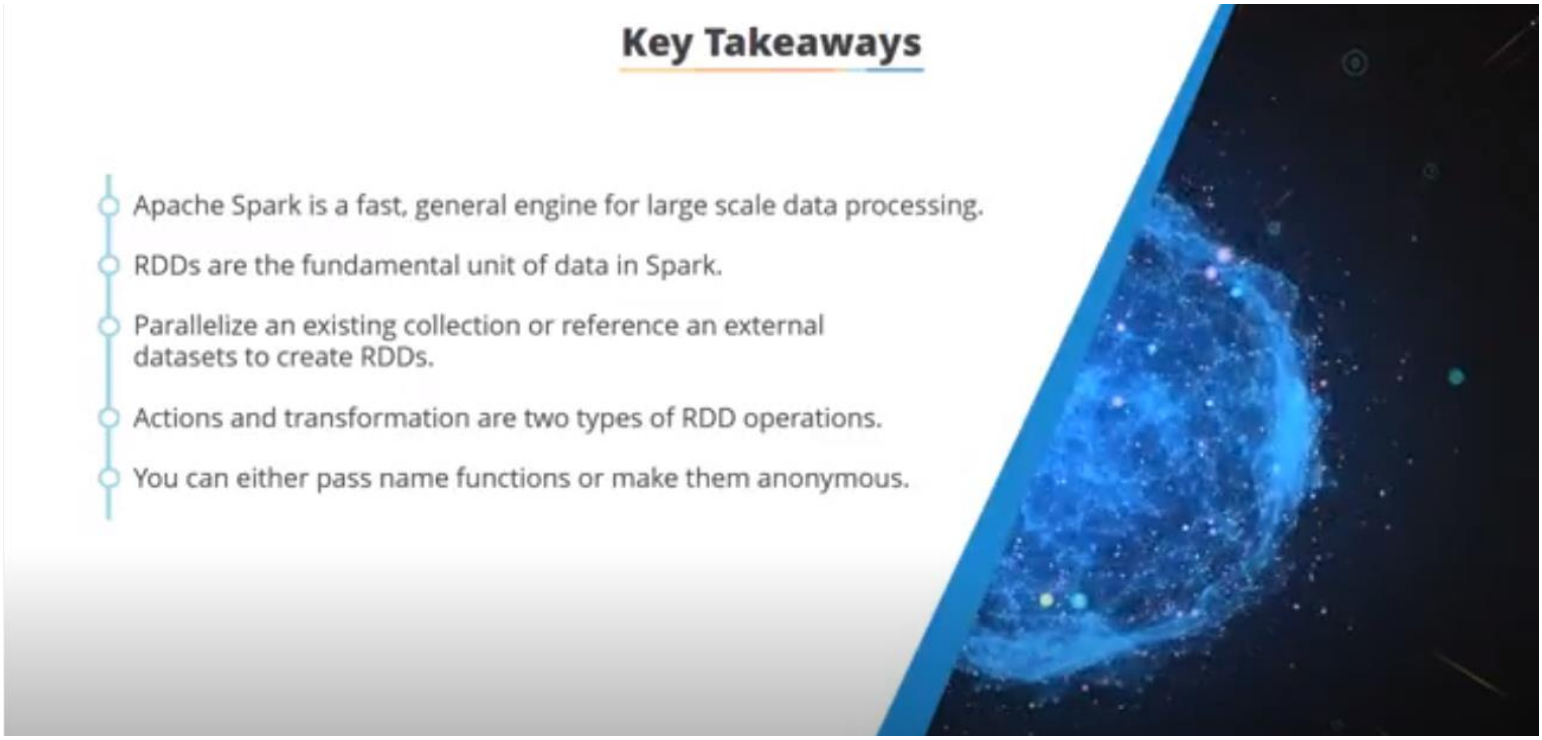
Functions defined anonymously: Anonymous functions are one-line functions, which are defined inside other functions. Anonymous functions are not global because they only stay for a short span.

They complete their job and cease to exist. Let us look at an example to understand more.

Python	Scala
`mydata.map(lambda line: line.upper()).take(2)` **Description**: Here, lambda is used to define an anonymous function inside the map function, which converts the elements of `mydata` temporarily identified with variable `line` to upper case. `take(2)` prints the first two elements of the resultant RDD.	`mydata.map(line => line.toUpperCase()).take(2)` **Description**: Here, `'line =>'` is used to define an anonymous function inside the map function, which converts the elements of `mydata` temporarily identified with variable `line` to upper case. `take(2)` prints the first two elements of the resultant RDD.

Say: With this, we conclude our session.

KEY TAKEAWAYS



CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 10.5

FACILITATION GUIDELINES

Play Video Content: Lesson 10.5

Content: Conclude the class by giving key takeaways of the session. Tell the students that they have learned about RDD and Apache Spark.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Play Video Content: Lesson 10.6. Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, continue the video and show the correct answer and explanation. Repeat this process for all the questions. If students have queries regarding the questions, discuss and resolve the queries.

Say: Let me test your understanding now by asking few more questions.

Questions:

1. What are the four limitations of MapReduce?

Answer:
- Unsuitable for trivial operations
- Unsuitable with OLTP
- Unfit for processing graphs
- Unfit for iterative execution

2. What is an RDD?

Answer: RDD is a fundamental data structure of Spark and works on parallel computing.

3. What are the four components of Spark?

Answer:
- Spark Submit Script
- Spark Applications
- Cluster Managers
- Spark EC2 Launch Script

4. What is action and transformation in an RDD?

Answer: A transformation produces new RDD from existing RDDs. When we want to work with the actual dataset, at that point, Action is performed.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1:

Download any text file from this link https://www.sample-videos.com/download-sample-text-file.php and perform following operations:

- Load the data into an RDD.
- Perform basic transformations operations learnt in the class.

Assignment #2:

Illustrate the Spark Execution Architecture with a diagram. Explain each component.

Session 10

Session Objectives:

- Create RDDs from files and from collections
- List the data types supported by RDD
- Apply single-RDD and multi-RDD transformations
- Describe SparkContext and Spark Application Cluster options
- List the steps to run Spark on Yarn
- List the steps to execute Spark application
- Explain dynamic resource allocation
- Understand the process of configuring a Spark application

Video Content: Lesson 11 and Lesson 12

QUICK RECAP

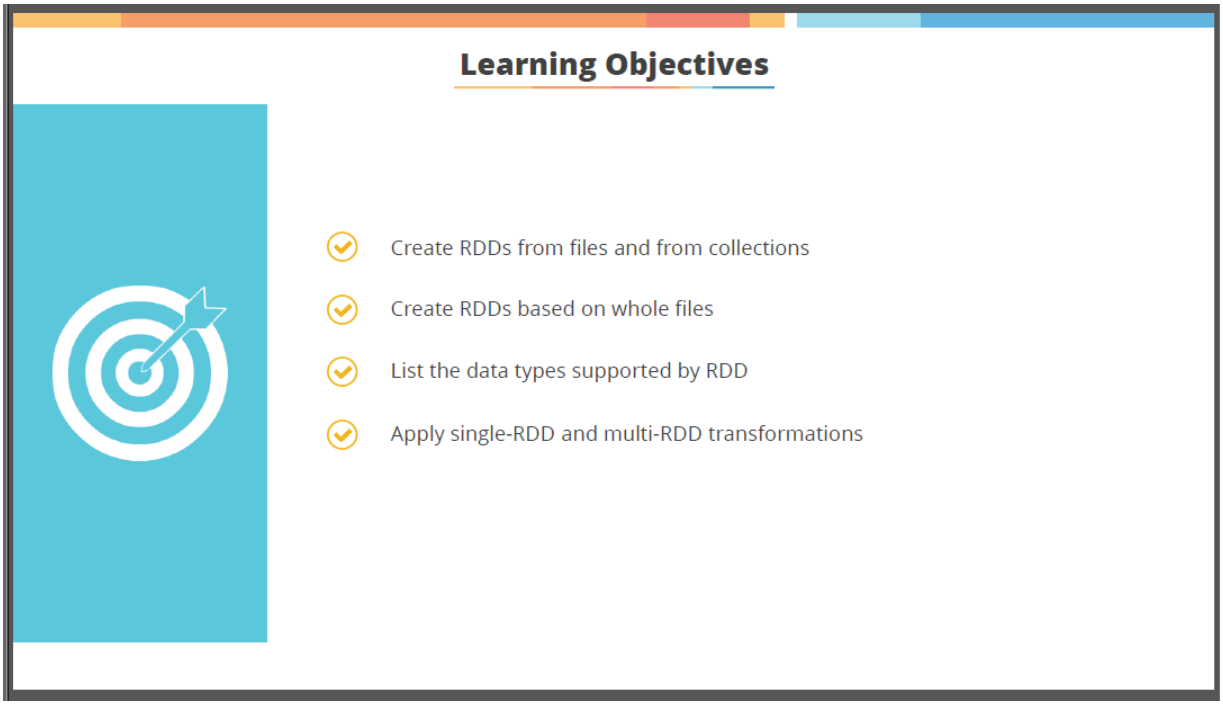
CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome the students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

INTRODUCTION



CLASS DURATION: 2 MINUTES

VIDEO DURATION: LESSON 11.1

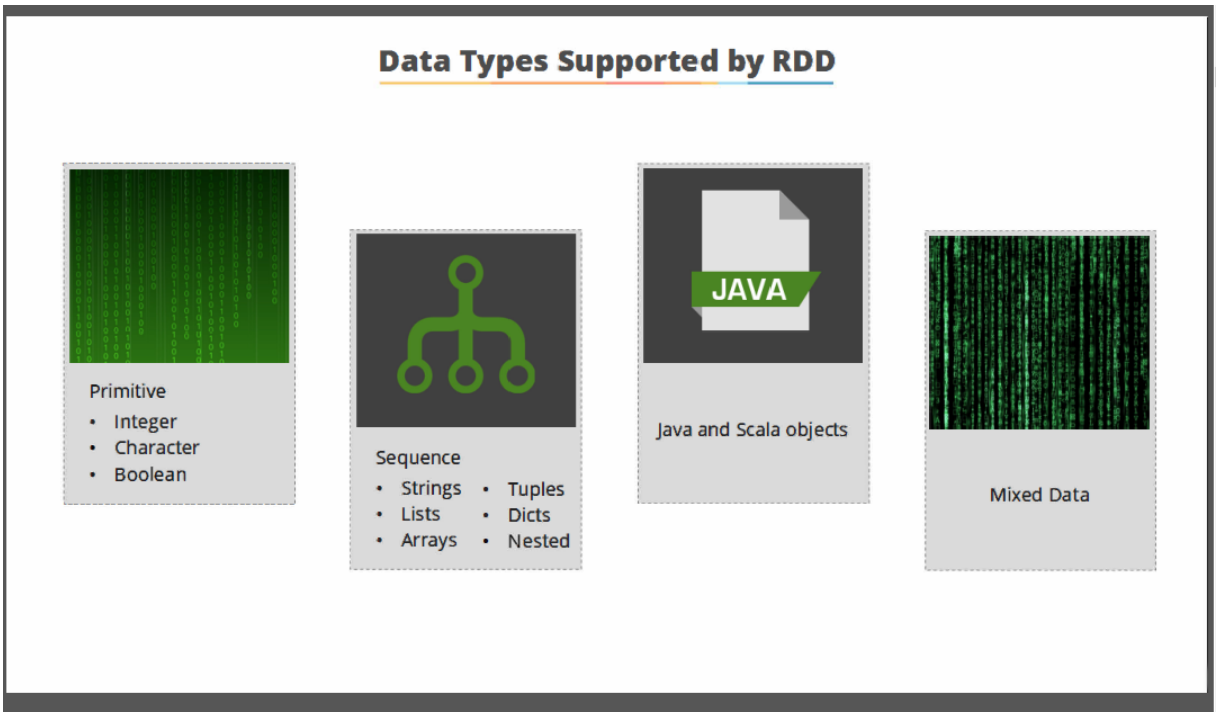
FACILITATION GUIDELINES

Play Video Content: Lesson 11.1

Content: After playing the video, repeat all the points and tell the students that in this session they will learn about RDD in Spark.

Say: Let us begin!

RDD DATA TYPES AND RDD CREATION



CLASS DURATION: 30 MINUTES

VIDEO DURATION: LESSON 11.2

FACILITATION GUIDELINES

Play Video Content: Lesson 11.2

Content: Let us focus on the different data types supported by RDD.

- ***Primitive data types***: Include common data types such as Integer (2, 3, 4, and so on.), Boolean (True or False), Character('a', 'd', 'G', and so on.)
- ***Sequence data types***: Include common sequence data types, such as String ("His name is John", "She ran because of rain", and so on.), Lists ([2,3,4,5], ['c', 'v', 't', 'p'], and so on.), Arrays ([[2,3][4,5],[6,7]]), Tuples((2,4,5,6,7,)), and Dicts ({'Steve': 3,'John': 7, 'Ria': 6})
- ***Java and Scala objects***
- ***Mixed data***

There also exists a different kind of RDD called (key, value) RDD. Every element of this RDD is of the form (key, value), where key and value can be of any data type. These are called **Pair RDD**. Example:

```
4, 'John'

6, 'Steve'

8, 'Shane'
```

Some operations of pair RDD are:

Activity 1

Duration: 15 MINUTES

Ask the students to go through the given link to see operations on (key,value) RDD.

https://intellipaat.com/blog/tutorial/spark-tutorial/working-with-keyvalue-pairs/

You already know that RDD can be initialized in various ways. Let us focus on some of the methods.

Creating RDD from a text file: RDD can be created from any text file. Each element of the text file is a separate element in an RDD. (Write the syntax on the board and explain.)

`sc.textFile('learn.txt')`, where `learn.txt` is the name of the text file.

RDD can also be created from more than one text files. (Write the syntax on the board and explain.)

`sc.textFile('learn1.txt', 'learn2.txt')`, where `learn1.txt` and `learn2.txt` are the names of the text files.

Creating RDD from collections: RDD can be created from collections given at the time of its initialization. (Write the syntax on the board and explain.)

```
data= ["learn", "is", "educational", "revolution"]

rdd1= sc.parallelize(data)
```

There is another way to initialize RDD:

```
rdd1= sc.parallelize("learn", "is", "educational", "revolution")
```

Creating RDD from whole files: RDD can be created from the whole text files as well. Suppose you have a folder containing 4 files and you want to create an RDD from all the four files, then this method forms a pair RDD with key as file name and its path, and value as the content of the file. (Write the syntax on the board and explain.)

```
rdd1= sc.wholeTextFile("path_to_folder"/*.csv) where * indicates to include all files in the folder to form an RDD
```

Creating pair RDD: Pair RDD can be created using the following syntax:

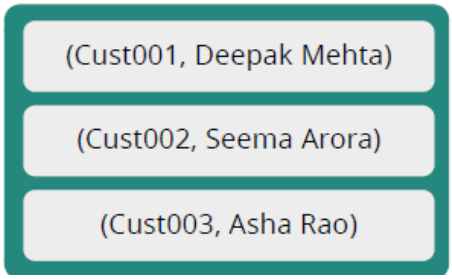
Suppose following is the content of a text file, **data.txt**: (Write the content and syntax on the board and explain).

```
Cust001 \t  Steve Johnson

Cust002 \t  John Watson

Cust003 \t  Andrea Warne
```

Python	Scala
`rdd1= sc.textFile(data.txt).map(lambda line: line.split('\t')).map(lambda row:(row[0], row[1]))` **Description**: Here `data.txt` is initialized as `rdd1`, the first map function splits each line according to separator `'\t'`, which splits each element of `data.txt` into two entries, and the second map function forms a pair (tuple) of both entries in each line of `data.txt`.	`val rdd1= sc.textFile(data.txt).map(line => line.split('\t')).map(row => (row[0], row[1]))` **Description**: Here `data.txt` is initialized as `rdd1`, the first map function splits each line according to separator `'\t'`, which splits each element of `data.txt` into two entries, and the second map function forms a pair (tuple) of both entries in each line of `data.txt`.

Output of both the syntax is same.

After playing the video, show the output of the code to the students.



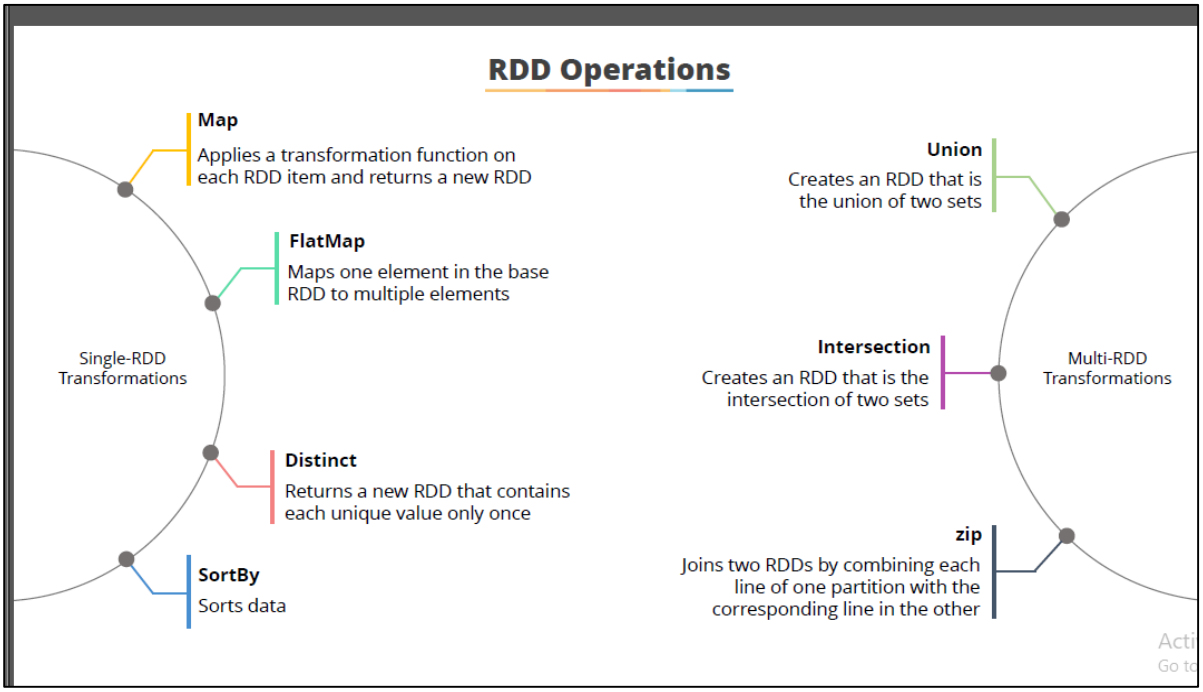
Reference: Refer to the following links for more information:

https://spark.apache.org/docs/latest/rdd-programming-guide.html#resilient-distributed-datasets-rdds

https://www.youtube.com/watch?v=eli2XiTRXF8

Say: Let us move to our next topic.

OPERATIONS IN RDD



CLASS DURATION: 15 MINUTES

VIDEO DURATION: LESSON 11.3

FACILITATION GUIDELINES

Play Video Content: Lesson 11.3

Content: Let us look at single RDD and multi RDD operations separately.

Operations on single RDD include:

map: Applies a transformation operation on each element of an RDD and returns a new RDD.

flatMap: Maps one element of the base RDD to many elements in the new RDD.

distinct: Returns unique elements of the base RDD to a new RDD.

sortBy: Sorts elements of an RDD.

We have already seen example of map function earlier. Let us now see flatMap and distinct functions in use.

Consider the text file, `learn.txt` has the following content:

```
Learn has several courses for children.

Learn has courses for adults.
```

Let us see the use of flatMap and distinct functions in Python and Scala. (Write the code on the whiteboard and explain.)

Python	Scala

`sc.textFile(learn.txt).flatMap(lambda line: line.split(' ')).distinct()` ***Description***: Here, `learn.txt` is initialized as RDD, `flatMap` is used to map one element of RDD to many elements in the new RDD by splitting each sentence according to spaces. Therefore, now, we have the new RDD as `[learn, has, several, courses, for, children, learn, has, courses, for, adults]`, which has some words repeated. The `distinct` function creates a new RDD of unique elements from the previous RDD. Therefore, now we have the final RDD as `[learn, has, several, courses, for, children, adults]`.	`sc.textFile(learn.txt).flatMap(line => line.split(' ')).distinct()` ***Description***: Here, `learn.txt` is initialized as RDD, `flatMap` is used to map one element of RDD to many elements in the new RDD by splitting each sentence according to spaces. Therefore, now, we have the new RDD as `[learn, has, several, courses, for, children, learn, has, courses, for, adults]`, which has some words repeated. The `distinct` function creates a new RDD of unique elements from the previous RDD. Therefore, now, we have the final RDD is `[learn, has, several, courses, for, children, adults]`.

Operations on multi-RDD:

Union: It creates a new RDD with elements from two RDDs similar to the union operation in sets.

Intersection: It creates a new RDD with common elements from two RDDs similar to the intersection operation in sets.

Zip: It deals with two RDDs simultaneously by combining each line of one partition with corresponding line in the other.

Let us look at the three operations in action. (Write these operations on the whiteboard and explain).

The two RDDs are:

Rdd1=[California, Ireland, Dublin, Australia, Austria]

Rdd2= [Ireland, Paris, France, Kenya, Australia]

Code:
`Rdd1.union(Rdd2).collect()`
 	
Output:
`[California, Ireland, Dublin, Australia, Austria, Paris, France, Kenya]`
	Code:
	`Rdd1.intersection(Rdd2).collect()`
	
	Output:
	`[Ireland, Australia]`
		Code:
		`Rdd1.zip(Rdd2).collect()`
		
		Output:
		`[(California, Ireland),(Ireland, Paris),(Dublin, France),(Australia, Kenya),(Austria, Australia)]`

The table explains certain general RDD operations.

Function	Syntax	Description
`collect()`	`rdd.collect()`	Returns a list of elements of RDD
`first()`	`rdd.first()`	Returns the first element of RDD
`foreach()`	`rdd.foreach(<function_name)`	Applies a function to each element of RDD
`sample()`	`rdd.sample(withReplacement=True,size,seed)`	Samples the RDD with replacement with size as fraction of the RDD's size, seed as the random number generator
`takeSample()`	`rdd.takeSample(withReplacement=True,size,seed)`	Samples the RDD with replacement with size of sample given by user, seed as the random number generator
`sampleByKey()`	`rdd.sampleByKey(withReplacement=True,size)`	Samples the RDD with replacement according to the key with size as fraction of the RDD's size
`mean()`	`rdd.mean()`	Computes the mean of elements of RDD
`stdev()`	`rdd.stdev()`	Computes the standard deviation of elements of RDD
`variance()`	`rdd.variance()`	Computes the variance of elements of RDD
`sum()`	`rdd.sum()`	Computes the sum of elements of RDD

Operations on (key, value) RDD:

countByKey()	Counts the number of elements of pair RDD according to the key.
groupByKey()	Makes groups of elements of pair RDD according to the key.
sortByKey()	Sorts the elements of pair RDD according to the key.
join()	Joins the two elements according to the key. Note that both the RDDs should have same keys.

Let us pay attention on the syntax of these.

Consider this example where RDD is `prdd1` that includes following data:

```
(00001, emp101)

(00002, emp102)

(00003, emp103)

(00003, emp131)

(00002, emp122)
```

Code:	**Code**:	**Code**:
`prdd1.sortByKey(ascending=False)` **Description**: Here, `sortByKey()` is used to sort the elements in descending order due to the parameter `ascending=False`. Therefore, the new RDD formed is: `[(00003, emp131), (00003, emp103), (00002, emp102), (00002, emp122), (00001, emp101)]`	`prdd1.groupByKey()` **Description**: Here, `groupByKey()` is used to group the elements according to key. Therefore, the new RDD formed is: `[(00003, [emp131, emp103]), (00002, [emp102, emp122]), (00001, emp101)]`	`prdd1.countByKey()` **Description**: Here, `countByKey()` is used to count the elements according to key. Therefore, the new RDD formed is: `[(00003,2), (00002,2), (00001,1)]`

For join operation, we will use two RDDs, `salesprofit.rdd` and `salesyear.rdd`

```
Salesprofit.rdd

(Cadbury's, $3.5M

(Nestle, $2.8M)

(Mars, $2.5M)

(Thorton's, $2.2M)

Salesyear.rdd

(Cadbury's, 2015)

(Nestle, 2014)

(Mars, 2014)

(Thorton's, 2013)
```

Code:

```
Salesprofit.rdd.join(Salesyear.rdd)
```

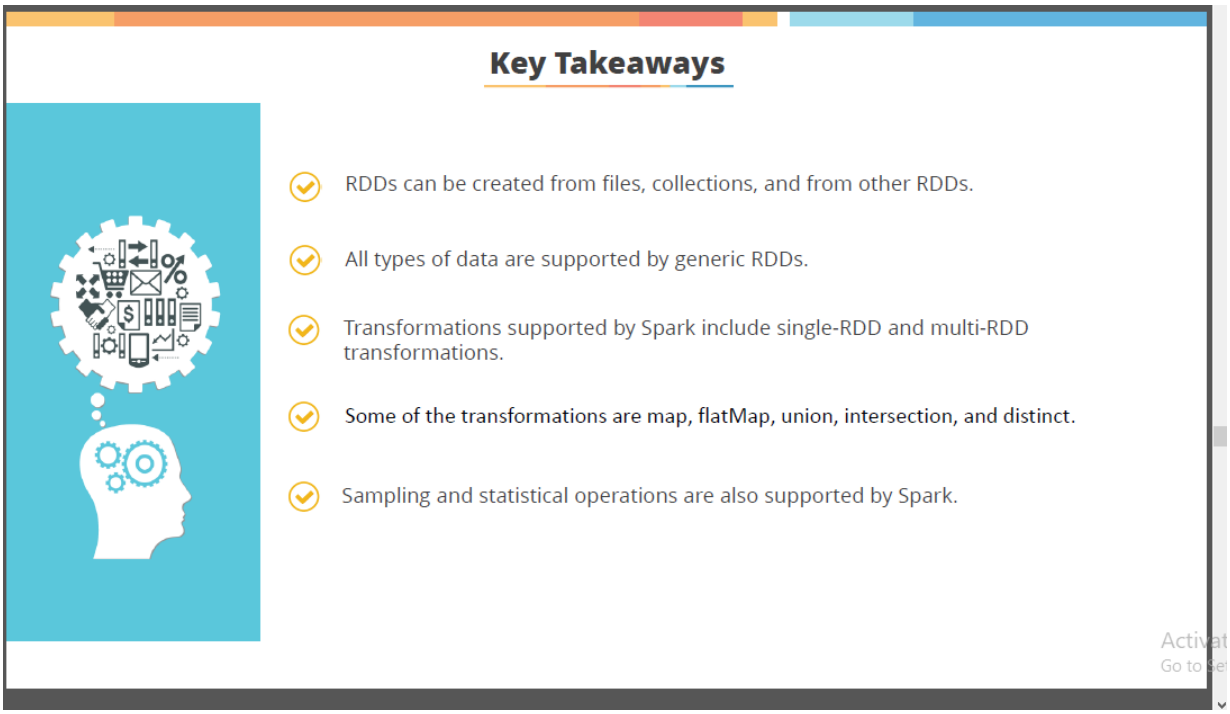
Description: Here, we have used the join operation to join the two RDDs. The new RDDs, thus formed will be merged with the values of RDD according to the key. Following output shows the company's profit and the year of the sale.

```
[(Cadbury's,[$3.5M, 2015]), (Nestle, [$2.8M, 2014]), (Mars, [$2.5M, 2014]), (Thorton's, [$2.2M, 2013])]
```

Reference: https://medium.com/@aristo_alex/how-apache-sparks-transformations-and-action-works-ceb0d03b00d0

Say: With this, we end our current topic.

KEY TAKEAWAY



CLASS DURATION: 2 MINUTES

VIDEO DURATION: LESSON 11.4

FACILITATION GUIDELINES

Play Video Content: Lesson 11.4

Content: Read aloud key takeaways of the session. Tell the students that they have learnt about single RDD and multiple RDD transformations.

Say: Let us go through a small quiz and test your knowledge.

QUIZ

CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 11.5

FACILITATION GUIDELINES

Content: Lesson 11.5 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions.

Questions:

1. What is a pair RDD?

Answer: RDD with key and value is called pair RDD.

2. What is a flatmap operation?

Answer: A flatmap operation maps one element of the base RDD to many elements in the new RDD.

3. Under which data type do integer, Boolean, and character fall?

Answer: They fall under primitive data type.

4. What data type does the following fall under?
   ({'Stella': 3,'Mary': 7, 'Ria': 6})

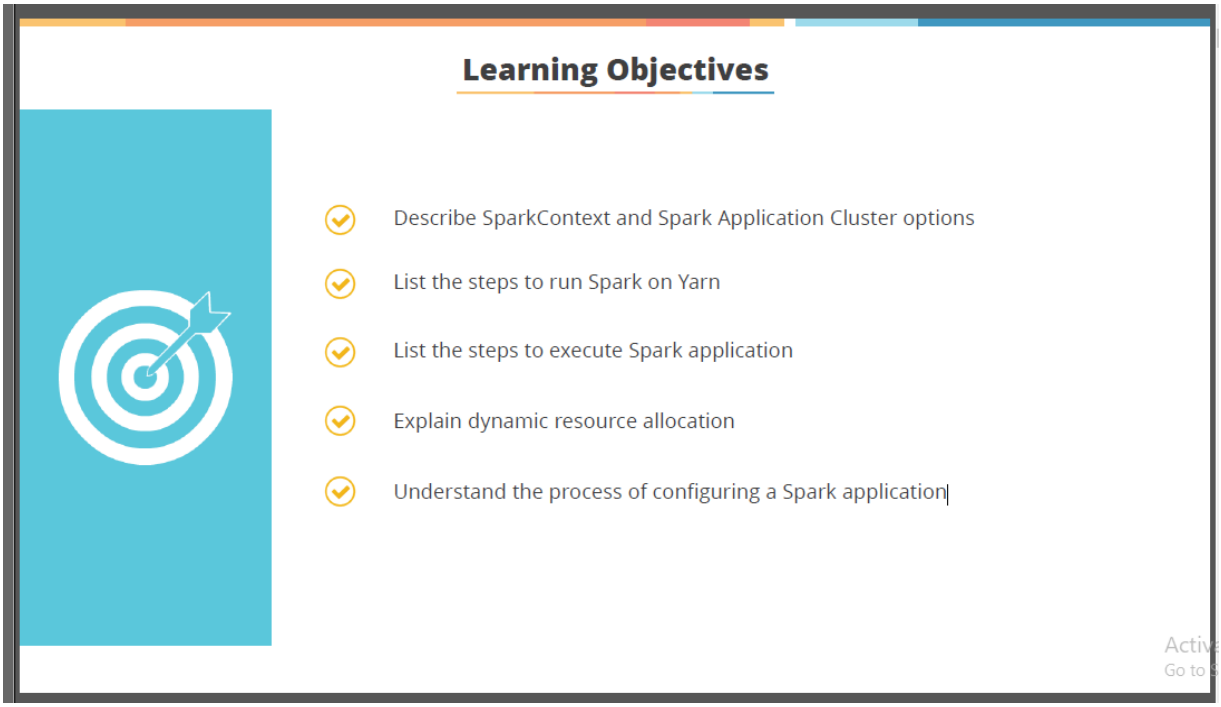
Answer: It is Dict. It is a sequence data type.

5. Mention some of the operations for pair RDD.

Answer: Some of the operations for pair RDD include sorting, joining, grouping, and counting.

Say: Let us move to out next topic, which is Lesson 12.

INTRODUCTION



CLASS DURATION: 8 MINUTES

VIDEO DURATION: LESSON 12.1

FACILITATION GUIDELINES

Play Video Content: Lesson 12.1

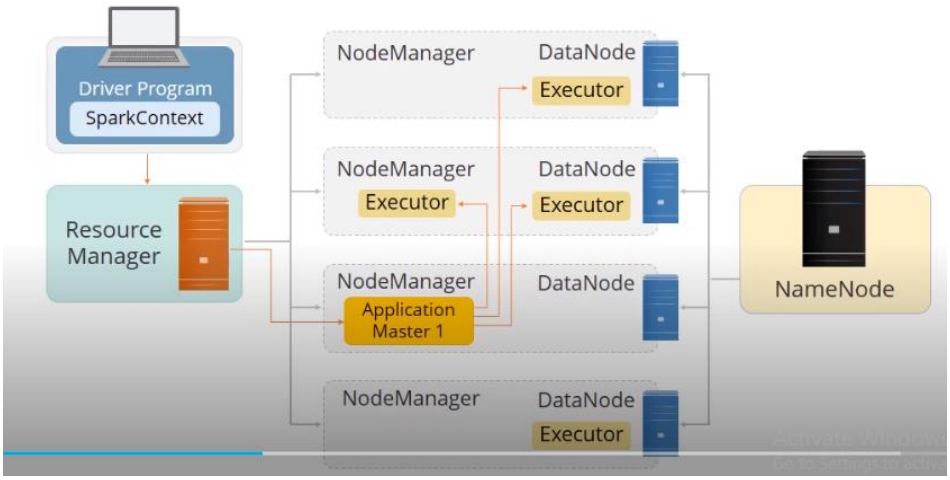
Content: Rewrite the objectives on the whiteboard and tell the students that in this session, they will learn about Spark Shell and Spark Applications.

Let us see the difference between Spark Shell and Spark application.

Spark Shell is a development environment that executes the Spark application. It allows you to interactively explore and manipulate data.

Spark application is a self-contained computation (this means this does not need human intervention to compute) that runs user supplied code to compute the result. It allows user to write interactive codes to operate on data.

We have already studied the framework of Spark earlier but let us dig more deep into it. As specified earlier, Spark has an entry level called **Spark Context**, which is a door to Spark development environment. It fills the gap between the user and the Spark environment. It then sends the command to the **resource manager**, which is in charge of allocating resources to the node manager or worker node. It is the worker node that further sends the command to the executers who implements the code and sends the results back to the Spark context. (Show the image to the students.)



The worker node is divided into two nodes, **Data node** and **Name node**. Data node directs the name node to execute the functions. As the name suggests, data node contains the data and the name node executes the commands on the data.

After we are done with the complete execution of the program, we can type **sc.stop()** in the terminal to terminate the existing Spark application.

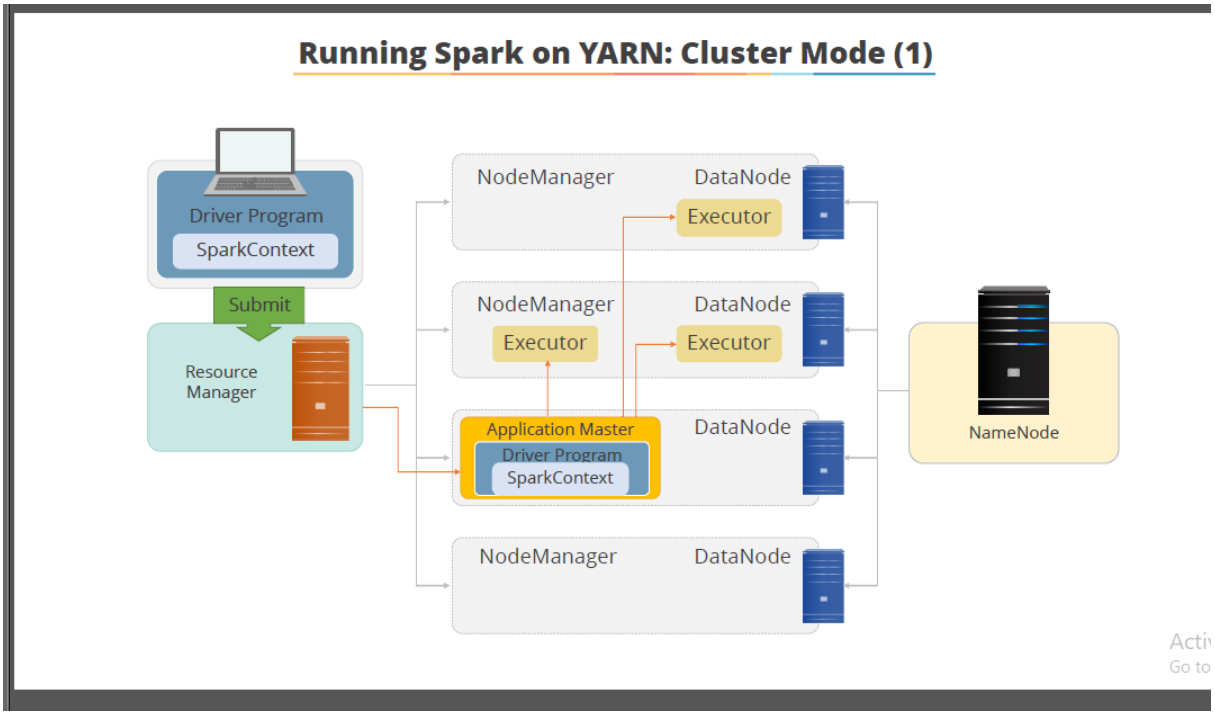
Spark Applications can run on one of two modes:

Spark Local Mode: In this mode, jobs run on a single machine and are executed in parallel using multi-threading approach. Therefore, it restricts parallelism (because no other nodes are required) to the number of cores in the single machine. This mode does not use any kind of resource manager. You can think of a local mode as executing a program on your system using single java virtual machine (JVM). This local mode can be Java, Scala or Python program where you have defined & used Spark Context object, imported Spark libraries and processed data residing in your system.
Spark Cluster Mode: It is the basic Spark-cluster architecture. Resource manager in a cluster mode can be of three types:

Hadoop YARN	YARN is Hadoop's resource manager. It has two components: schedulers and application managers. Scheduler checks whether the jobs are done whereas application manager accepts the job submission.
Spark Standalone	It is available as a part of the Spark distribution. It is resilient to worker failures and has capabilities of managing resources per application. It schedules the applications using first in first out (FIFO) approach, where each application uses all the nodes available in the cluster.
Apache Mesos	This resource manager comes along with Spark environment and works the same way as the cluster manager of the basic Spark architecture.

Say: Now, let us move to our next topic.

RUNNING SPARK ON YARN



CLASS DURATION: 6 MINUTES

VIDEO DURATION: LESSON 12.2

FACILITATION GUIDELINES

Play Video Content: Lesson 12.2

Content: There are two modes of deployment in Spark known as client mode and cluster mode. These modes direct where the driver program should run. It can run either on the worker node inside the cluster, which is the cluster mode or it can run on an external client, which is the client mode.

Let us see how Spark functions on YARN on the client mode.

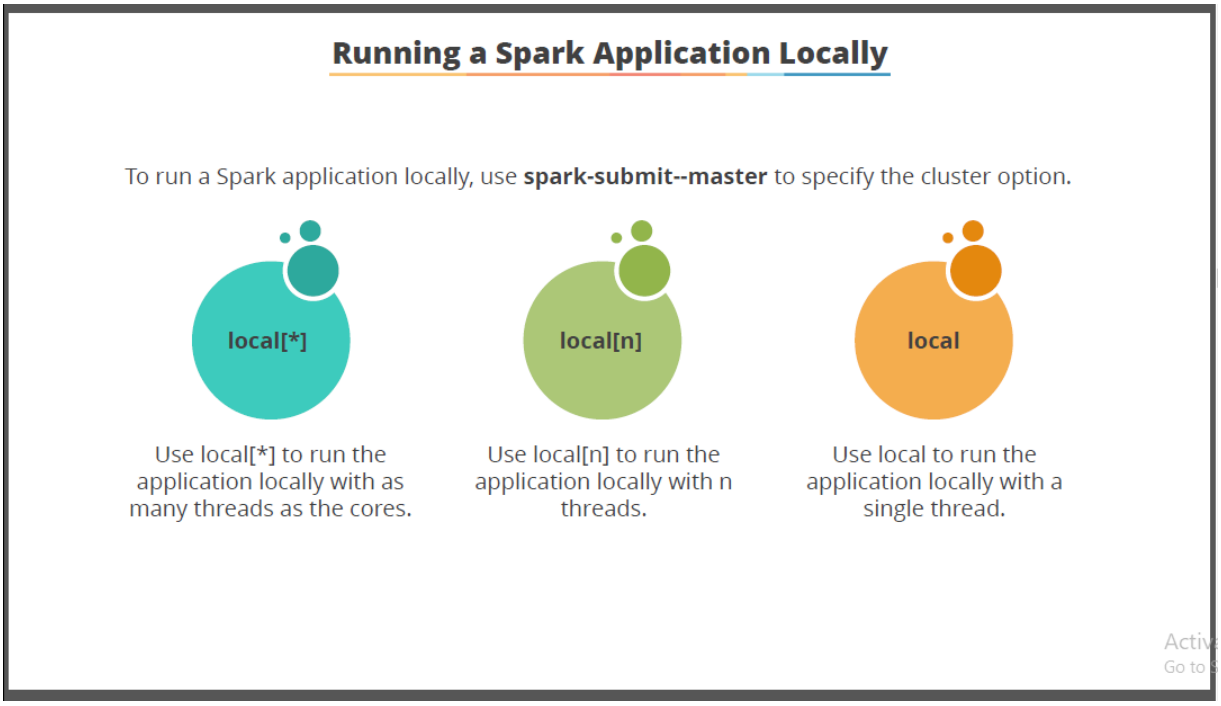
In the client mode, the Spark Context runs on the client machine externally. The Spark driver program sends the command to the resource manager, which distributes the work to the worker nodes. The worker nodes direct the executers to implement the work, who sends the result back to the Spark Context. Now, suppose another client has opened Spark Context, which has its own node manager. Spark Context follows the same process and gets back the result in its own driver. Now the Spark application currently has two clients running on it with their own Spark drivers. At a time, the Spark application can have more than one client running on it.

In the cluster mode, the Spark cluster has a Spark Context object already installed. Therefore, Spark Context directly commands the executors, which in turn returns the output to the Spark Context.

Reference: https://techvidvan.com/tutorials/spark-modes-of-deployment/

Say: Let us move to our next topic.

RUNNING A SPARK APPLICATION



CLASS DURATION: 6 MINUTES

VIDEO DURATION: LESSON 12.3

FACILITATION GUIDELINES

Play Video Content: Lesson 12.3

Content: To run a Spark application locally, use the command `spark-submit --master` (both in pyspark and Spark Shell) in the terminal and specify one of the thread options:

local[*]: To run the application locally, specify the threads equal to the number of cores in the Spark Context (because this is the default option)

local[n]: Specifies the number of threads equal to n.

local: Specifies the number of threads equal to one.

To run a Spark application on a cluster (which is done mostly) use the command `spark-submit --master` in the terminal and specify one of the cluster options from the following:

yarn-client: Use this option if Yarn runs on the client mode.

yarn-cluster: Use this option if Yarn runs on the cluster mode

spark://**masternode**:**port**: Use this option if Spark uses standalone resource manager

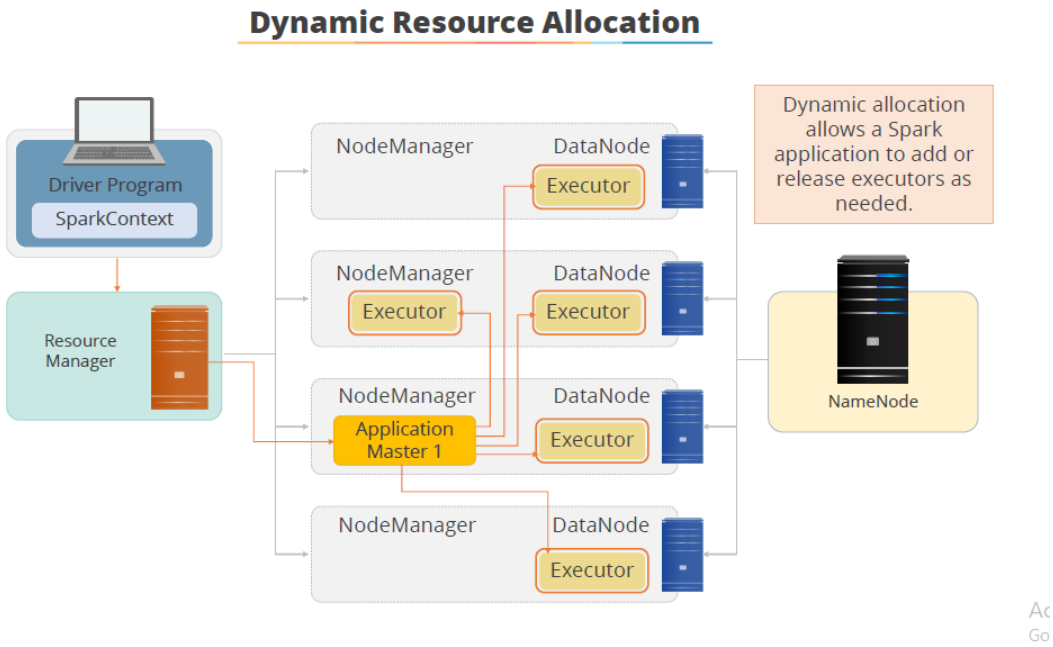
mesos://**masternode**:**port**: Use this option if Spark uses Apache Mesos resource manager

To start a Spark application on a cluster, follow the steps: (Write the steps on the whiteboard.)

1. Use the `--master` option to start the Spark locally or on cluster.
2. Start the Spark shell by using Spark or Apache Mesos cluster manager URL.
3. If Spark runs on the local mode, specify the thread option. If Spark runs on cluster, specify the number of nodes (or executers), you need to work.
   You are now ready to build your Spark application.

Reference: https://jaceklaskowski.gitbooks.io/mastering-apache-spark/spark-shell.html

DYNAMIC RESOURCE ALLOCATION



CLASS DURATION: 5 MINUTES

VIDEO DURATION: LESSON 12.4

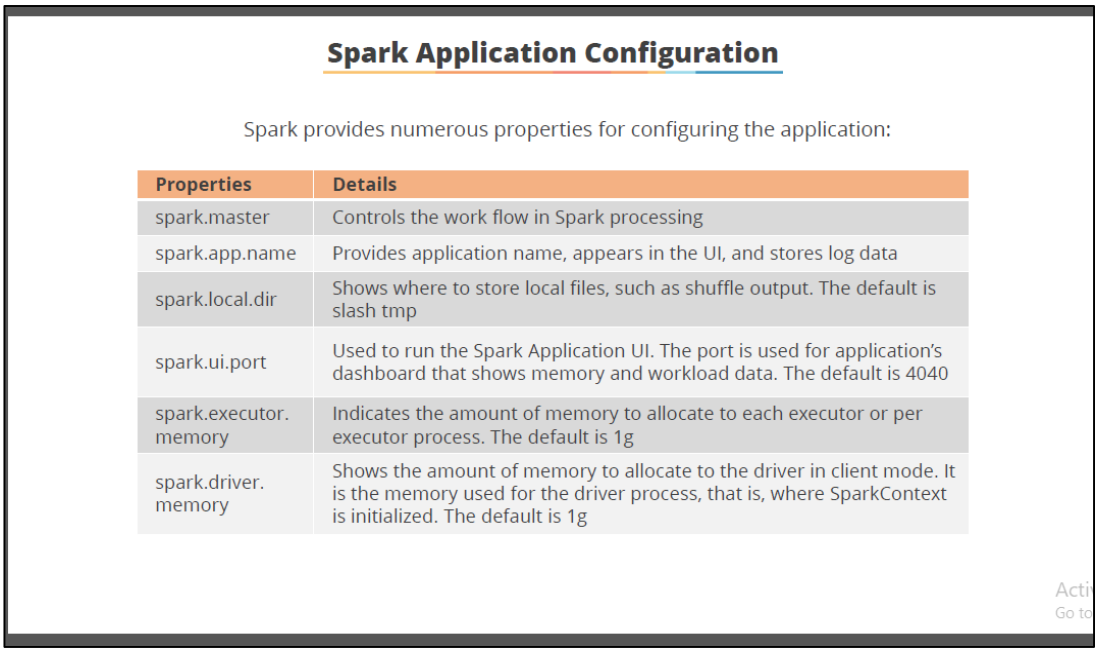
FACILITATION GUIDELINES

Play Video Content: Lesson 12.4

Content: Dynamic resource allocation refers to the instant entry and exit of resources in a Spark cluster according to the requirements. This feature allows adding or removing executors depending upon the length and complexity of the job assigned to the executers.

Suppose the numbers of executers specified in the start is 10, and the resource manager later realizes that some executers are free with no work, it automatically removes those executors. Suppose it realizes that the task is more cumbersome and load is increasing on the current working executers, it adds more executers and distributes the job amongst them.

CONFIGURING YOUR SPARK APPLICATION



CLASS DURATION: 7 MINUTES
VIDEO DURATION: LESSON 12.5
FACILITATION GUIDELINES

Play Video Content: Lesson 12.5

Content: To start a Spark application, you require to set up the Spark configuration environment. Basically, you need to set up the framework according to your requirements and then start your application.

To set up a Spark application environment, you should be familiar with Spark properties. Certain Spark properties are described here:

Properties	Description
`spark.master`	Acts as a supervisor who controls the work flow in the Spark architecture
`spark.app.name`	Gives name to each application that appears in the Spark UI and records data. Note that every application should have an identity.
`spark.local.dir`	Shows path to store the local files in the main memory. The default location is `'/tmp`.
`spark.ui.port`	Runs the Spark application UI and the port records data workload and memory. Note that every application has a UI. It refers to the outer layout of any application.
`spark.executer.memory`	Indicates the amount of memory required for each executer. Note that each executer needs some amount of memory to store the dynamic data on which it has to operate on.
`spark.driver.memory`	Shows the amount of memory to allocate to the driver in client mode. It is the memory used for the driver process, that is, where SparkContext is initialized.

There are two techniques by which we can set our Spark configuration.

- **By Declarative configuration options**: These are used for global configurations.
    - **Spark-submit script**: Initializes the starting driver memory and number of executers by the following commands:

      ```
      spark-submit --driver-memory 500M

      spark-submit --conf spark.executor.cores=4
      ```

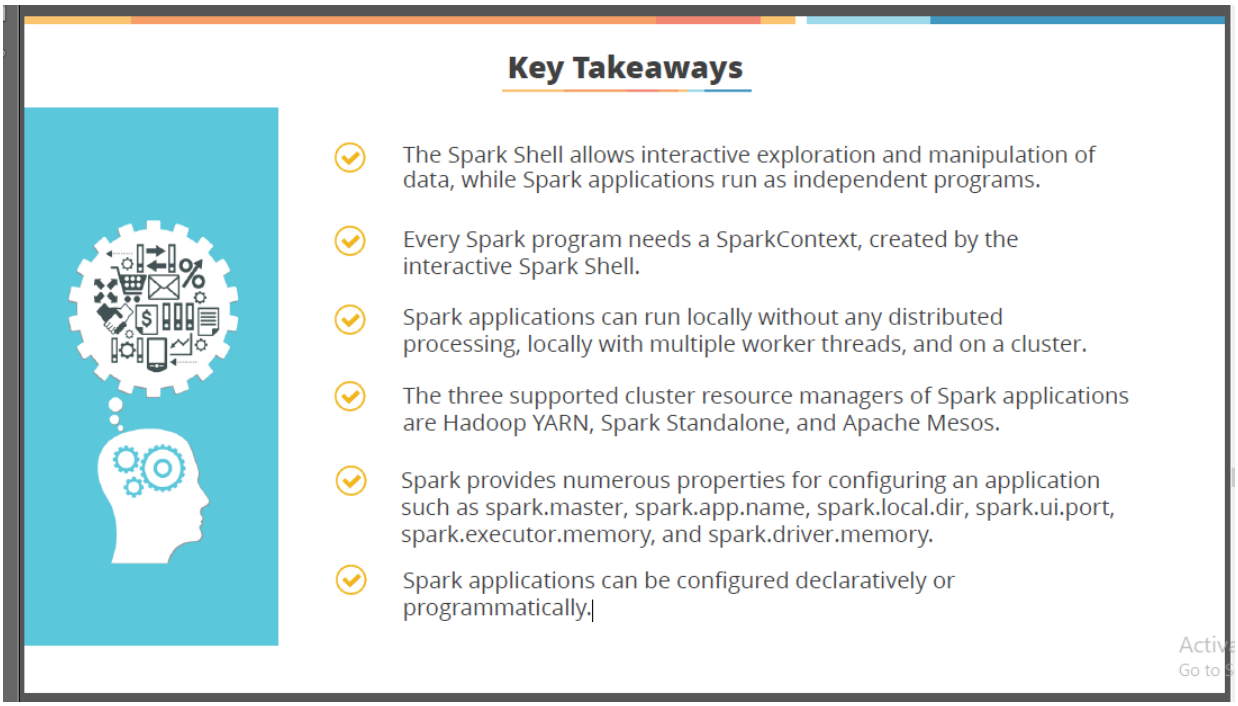
    - **Properties file**: Contains the properties of Spark applications. These properties can be loaded using the following command:

      ```
      spark-submit --properties-file filename
      ```

    - **Spark default properties file**: Configures the properties of all Spark applications with its default method.
- **By setting configurations programmatically**: Configurations are set in the programs by set functions:
    - `setAppName(name)`: Sets the name of the application in Spark context, provide the name of the app in parenthesis.
    - `setMaster(master)`: Sets the number of the executers and other details in the master, provide the details in the parenthesis.
    - `set(property- name,value)`: Sets the property name with its value, provide the property name and value in the parenthesis.

Say: With this, we conclude our session.

KEY TAKEAWAYS



CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 12.5

FACILITATION GUIDELINES

Play Video Content: Lesson 12.5

Content: Summarize using the key takeaways and tell the students that they have learnt about Spark shell, Spark program, and Spark applications.

QUIZ

CLASS DURATION: 5 MINUTES

FACILITATION GUIDELINES

Content: Lesson 12.7 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions.

Questions:

1. What are the three resource managers?

Answer:

Hadoop YARN: YARN is a Hadoop's resource manager.

Spark Standalone: It is available as a part of the Spark distribution.

Apache Mesos: It is a part of Spark environment.

2. What is dynamic resource allocation?

Answer: Dynamic resource allocation refers to the instant entry and exit of resources in a Spark cluster according to the requirements.

3. What are the four components of Spark?

Answer:

- Spark Submit Script
- Spark Applications
- Cluster Managers
- Spark EC2 Launch Script

4. What is the Spark local mode?

Answer: In this mode, jobs run on a single machine and are executed in parallel using multi-threading. The mode does not use any kind of resource manager.

5. What is a difference between a Spark shell and a Spark application?

Answer:

Spark shell is a development environment to execute the Spark application. It allows interactive exploration and manipulation of data.

Spark application is a self-contained computation that runs user supplied code to compute the result. It allows a user to write interactive codes to operate on data.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1: Download the text file through this link http://25.io/toau/audio/sample.txt and perform the following operations:

- ✓ Load the file as RDD.
- ✓ Use the `flatMap` function to form a new RDD with the words from text file.
- ✓ Use the `map` function to form a (key, value) RDD with unique words as key and their count as values.

Session 11

Session Objectives:

- Explain how RDDs are distributed across a cluster
- Analyze how Spark partitions file-based RDDs
- Explain how Spark executes RDD operations in parallel
- Explain how to control parallelization through partitioning
- Analyze how to view and monitor tasks and stages
- Explain how RDD lineage is created
- Explain how to mark persistence on an RDD
- Explain the features of RDD persistence
- List the storage levels in RDD persistence
- Describe how distributed persistence and fault tolerance help avoid data loss

Video Content: Lesson 13 and 14

QUICK RECAP

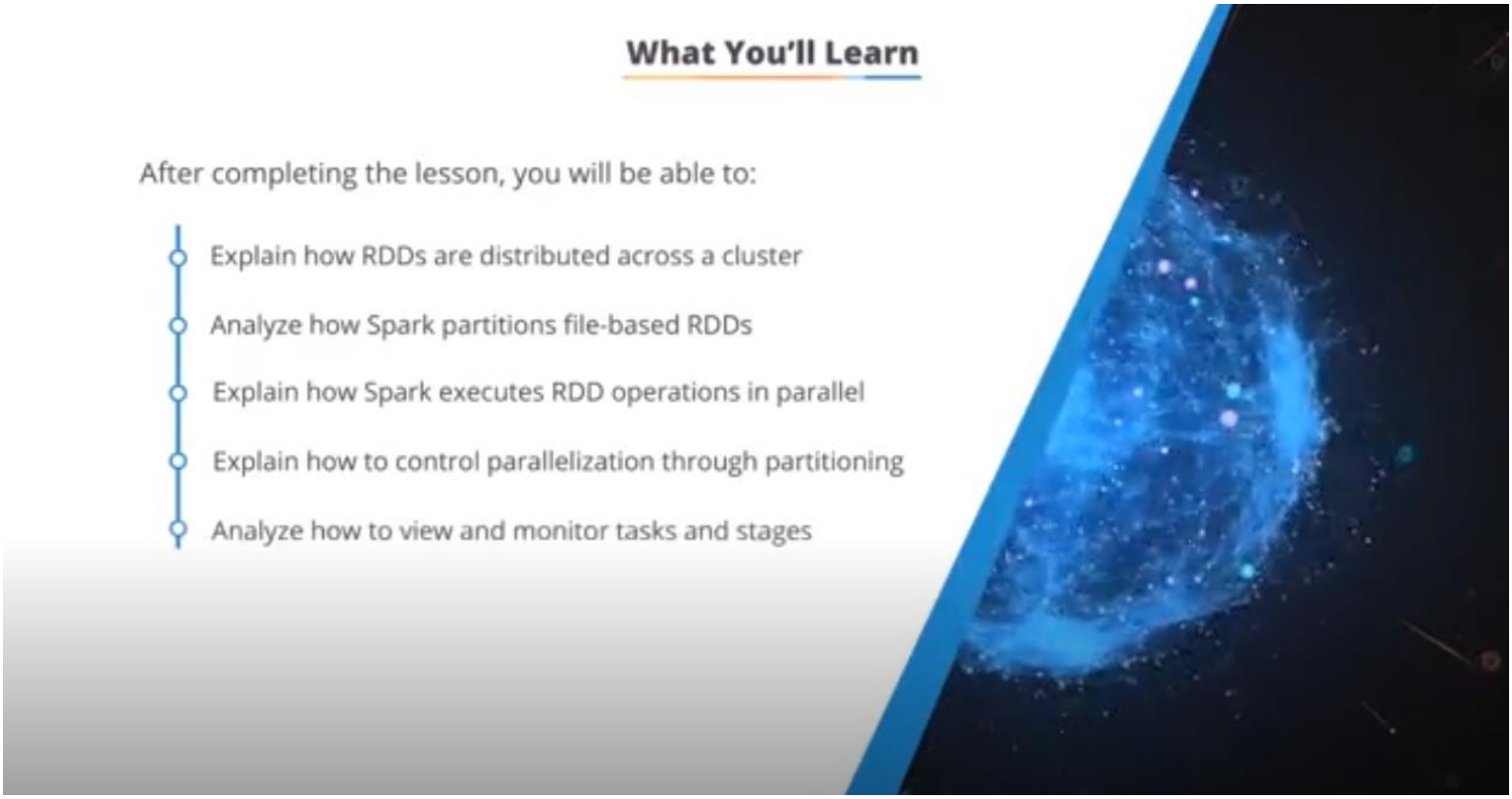
CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

LESSON 13: INTRODUCTION



CLASS DURATION: 2 MINUTES

VIDEO DURATION: LESSON 13.1

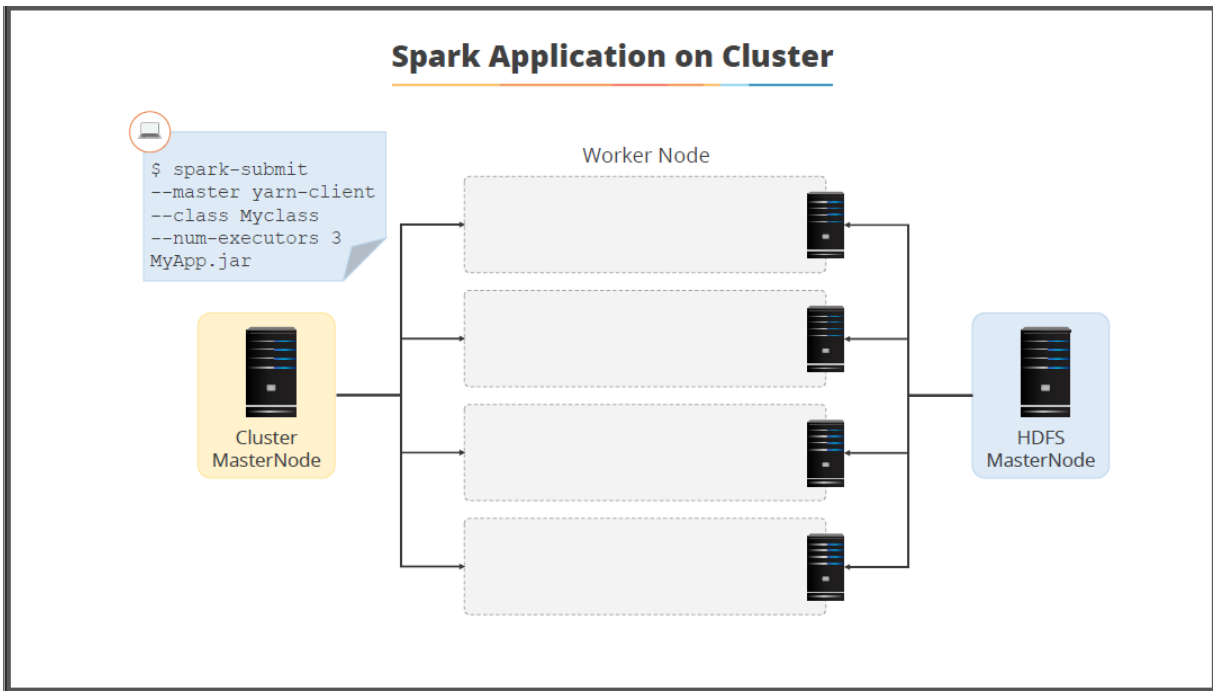
FACILITATION GUIDELINES

Play Video Content: Lesson 13.1 till 00:45 seconds

Content: After playing the video, repeat all the points and tell the students that in this session, they will learn about parallel processing in Spark.

Say: Let us begin!

SPARK CLUSTER



CLASS DURATION: 9 MINUTES

VIDEO DURATION: LESSON 13.1

FACILITATION GUIDELINES

Play Video Content: Continue Lesson 13.1

Content: This session focuses on Spark's method of parallel computing. As learnt earlier, parallel computation is a process of distributing work among several worker nodes to make the process faster and to optimize memory and time. Spark has a very simple functioning in the cluster mode. When a user submits a job using Spark-submit, the Spark Context comes into work. It then sends the job to the master-cluster node, which is responsible to distribute the job to the parallel worker nodes. As soon as the master-cluster node receives the job, executers are opened in the worker nodes that communicate with the Spark Context to send the result back.

When data is loaded in an RDD, it is partitioned across several worker nodes, which can be given by the user itself otherwise, Spark sets its default number of partitions that is 2. When data and job are distributed among the partitions, the executers complete the job and send the result back to the Spark driver program. Data can be partitioned from the single files as well as multiple files. We have seen the method of partition from a single file. Now, let us see the method of partitioning from multiple files.

In multiple files, the number of partitions is at least the number of total files. Suppose we have a directory named `mydir`, which contains 10 files that are to be loaded as RDD. Then, the number of partitions made by the Spark Context is at least 10, unless specified by the user.

Here is the syntax to load the file with the given number of partitions. (Write the syntax on the whiteboard.)

For single file:

```
Sc.textFile('file.txt', numPartitions= 12)
```

For multiple files:

```
Sc.textFile('path_to_folder_containing_files/*', numPartitions = 14)
```

Recall, we also studied loading files with `sc.wholeTextFiles`. It is used to create partitions from multiple small files and form a (key, value) RDD where key is the file name, and path and value are the content of the file.

RDD operations can also be operated on partitions. Let us focus on some basic operations on partitions.

foreachPartition: This operation calls a function to each partition of the RDD.

mapPartitions: This operation creates a new RDD by applying a function to each partition of the current RDD.

mapPartitionsWithIndex: This operation works similar to the mapPartitions but involves index of the partition.

Now, we will see the role of HDFS with Spark Context. We use the command `$hdfs dfs –put mydata` where `mydata` is the file we want to load in the HDFS. This file is then broken up into several data blocks. We can then use Spark Context and load the file `mydata` to an RDD. Following would be the command:

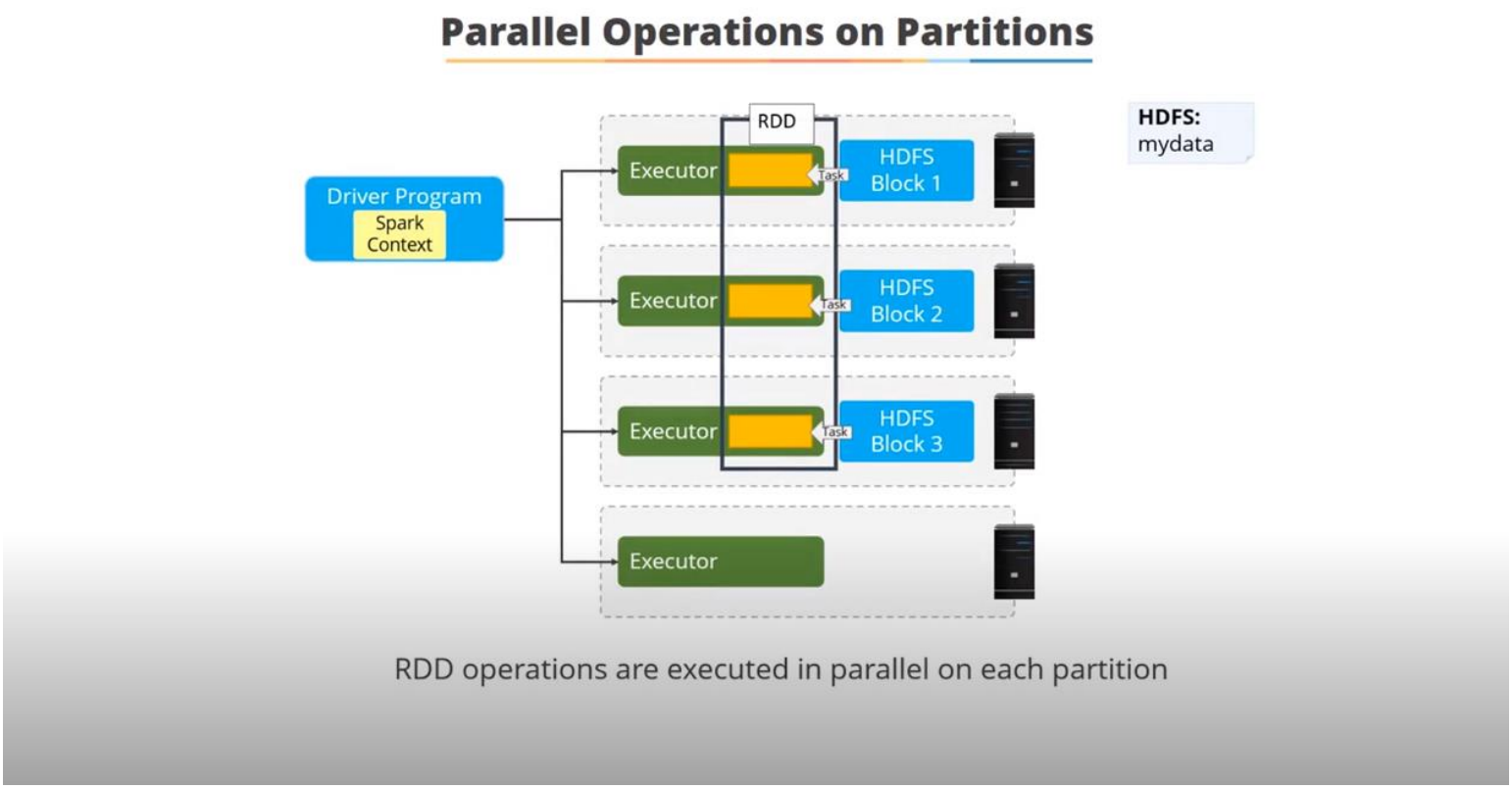
```
sc.textFile("hdfs://path/mydata.txt")
```

Reference: Refer to the following links for more information:

https://spark.apache.org/docs/latest/cluster-overview.html
https://intellipaat.com/blog/spark-executes-real-time-parallel-processing/
https://www.youtube.com/watch?v=3cnzpVDLBJg

Say: Let us move to the next topic, Parallel Operations on Partitions.

PARALLEL OPERATIONS ON PARTITIONS



CLASS DURATION: 8 MINUTES

VIDEO DURATION: LESSON 11.2

FACILITATION GUIDELINES

Play Video Content: Lesson 11.2

Content: Parallel operations on RDD are not that simple, as it seems. Selecting number of partitions is the most vital job for a developer, because the role is not only to code and find results but to keep in mind the time taken for the completion of each job. The time taken by Spark to complete any job mostly depends on the number of partitions. Certain operations on RDD work on the same partitions those provided at the start. Certain operations require repartition of data called **shuffling**. Shuffling is the process of redistributing data across partitions. It involves movement of data between stages.

Before understanding shuffling, let us understand some important terminologies of Spark.

Job: A set of tasks executed as a result of an action.

Stage: A set of tasks in a job that can be executed in parallel.

Task: An individual unit of work sent to one executer.

Application: It can contain any number of jobs managed by a single driver.

Operations such as reduceByKey, sortByKey, join, and groupByKey require shuffling of data across stages. Shuffling comes with a disadvantage. It increases the time of job completion. Therefore, developers try to use less of shuffling operations. When dealing with big data, its manipulation, and output comprise of lots of operations and jobs done by the executer. Therefore, sometimes, it becomes difficult for the developer to optimize tasks/operations manually. Spark has a feature called Directed Acyclic Graphs (DAG). It makes this graph indicating correct alignment of operations, which optimizes job time.

Resource: Ask students to go through the following links to learn more about DAG and shuffling operations:

https://intellipaat.com/community/7394/how-dag-works-under-the-covers-in-rdd
https://spark.apache.org/docs/latest/rdd-programming-guide.html#shuffle-operations

Additional Resources:
https://jaceklaskowski.gitbooks.io/mastering-apache-spark/spark-rdd-shuffle.html
https://www.youtube.com/watch?v=kbQmZiT1gnA

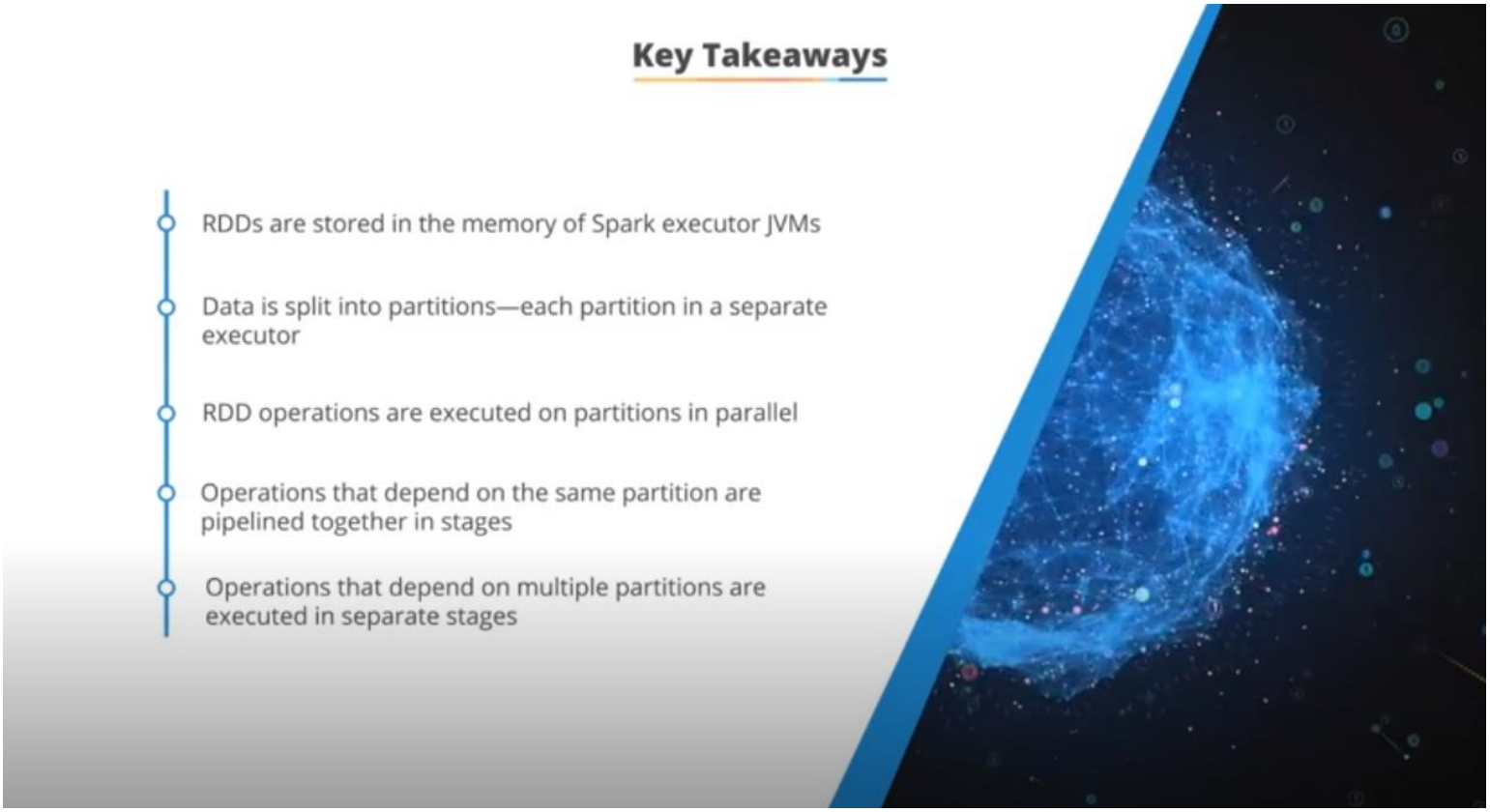
Say: Thus, this was all about parallel computation of RDD. We shall now move to the conclusion of this topic.

ACTIVITY 1

CLASS DURATION: 15 MINUTES

Research and read the use cases of Apache Spark.

KEY TAKEAWAYS



CLASS DURATION: 2 MINUTES

VIDEO DURATION: LESSON 13.3

FACILITATION GUIDELINES

Play Video Content: Lesson 13.3

Content: Rewrite the key takeaways of the session on the whiteboard. Tell the students that they have learnt about RDD operations.

Say: It is time to check our understanding!

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Content: Lesson 13.4 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions.

Questions:

1. What is the default number of RDD set by Spark?

Answer: 2

2. When data and job are distributed among the partitions, what completes the job and sends the result back to the Spark driver program?

Answer: The executers complete the job and send the result back to the Spark driver program.

3. What are some of the basic operations on partitions?

Answer: foreachPartition, mapPartitions, and mapPartitionsWithIndex.

4. What is a job?

Answer: A job is a set of tasks executed as a result of an action.

LESSON 14: INTRODUCTION

CLASS DURATION: 10 MINUTES

VIDEO DURATION: LESSON 14.1

FACILITATION GUIDELINES

Play Video Content: Lesson 14.1 till 00:53 seconds

Content: After playing the video, repeat all the points and tell the students that in this session, they will learn about optimization techniques in RDD.

Play Video Content: Resume Lesson 14.1

Content: What is **Lazy Evaluation**? (Allow the students to respond)

Lazy evaluation stands for the execution of operations performed on an RDD in a lazy pattern. That means until an action operation is performed on the RDD, Spark does not start the execution of transformation operations. This leads to the formation of **RDD lineage**. RDD lineage is a graph of all the parent RDDs of an RDD. For example, consider the below code: (Write the code on the whiteboard)

For this code, we will use the text file `learn.txt` having this text.

```
learn is an education provider. It is based in San Francisco.

it has trained 450,000+ customers.

it offers 400+ professional courses.
```

Code: (In Python)

```
mydata= sc.textFile("learn.txt").map(lambda s: s.upper()).filter(lambda s:
s.startswith('I')).count()
```

Description: In this code,

1. The sub-code `sc.textFile("learn.txt")` creates an empty RDD and links it to the file `learn.txt` but does not fill it with the contents of the file.
2. The sub-code `map(lambda s: s.upper())` creates a mapped empty RDD and links it to the previous RDD, but does not fill it with the result of map operation.
3. The sub-code `filter(lambda s: s.startswith('I'))` creates a mapped empty RDD and links it to the previous RDD, but does not fill it with the result of map operation.
4. The sub-code `count()` creates final RDD `mydata`, and starts executing the operations from the beginning to output the result of the last action.

Meanwhile, in steps 1, 2, and 3, the empty RDDs forms a lineage, where RDD in step 1 is the parent RDD of the RDDs in step 2, 3, and 4.

RDD in step 2 is the parent RDD of RDD in step 3 and 4; and RDD in step 3 is the parent RDD of the RDD in step 4.

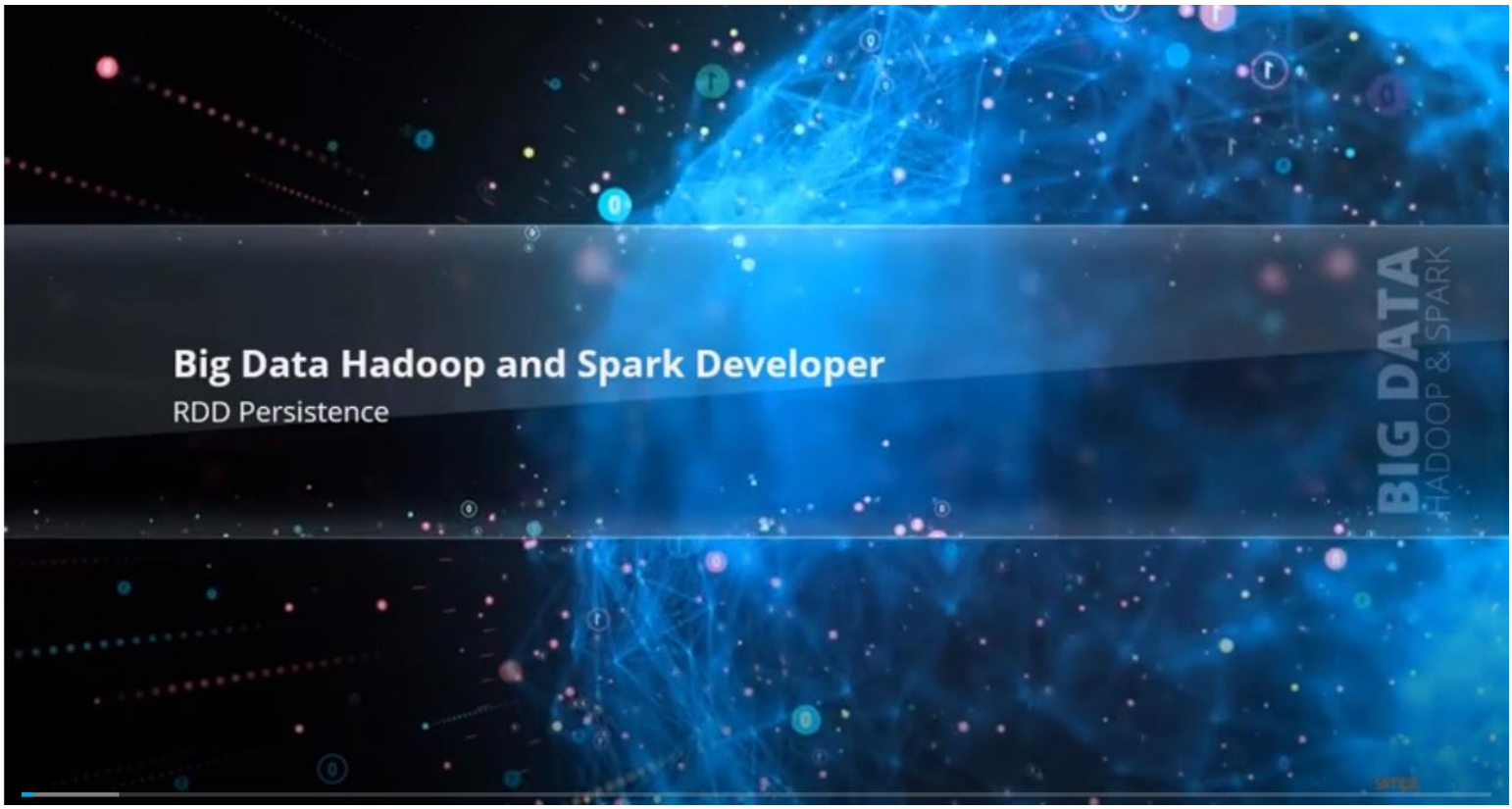
Suppose we call the last action operation again, Spark will go through the whole lineage again from the start because the RDDs in the steps 1, 2, and 3 are not stored in the memory.

(Keep the code on the whiteboard for the next topic.)

Reference: Refer to the following links for more information:
https://data-flair.training/blogs/rdd-lineage/
https://www.youtube.com/watch?v=GLE4mMuqCrY
http://cds.iisc.ac.in/wp-content/uploads/DS221.L6.Spark_.pdf

Say: Let us move to the next topic.

RDD PERSISTENCE



CLASS DURATION: 20 MINUTES

VIDEO DURATION: LESSON 14.2

FACILITATION GUIDELINES

Play Video Content: Lesson 14.2

Content: (Point to the code on the whiteboard.) This example code is very short and simple. Calling the last action `count()` again will lead to the whole computation again but will not make much difference in the speed and time of computation because of its small length and trivial nature. In actual, dealing with big data (approx. petabytes) involves lengthy codes and large memory to accommodate data as well as the code, therefore the computation of whole code again and again might lead the system to hang or more time in processing, which is a sheer waste of time. Therefore, to overcome this, there is a method called **RDD persistence**. This method when called on a particular RDD saves it in the memory at that instance. Therefore, calling the action operation does not involve computation from the beginning. Rather, it starts from the saved RDD, thus optimizing speed and improving performance. (Point to whiteboard) Let us consider the same example to understand the notion of RDD persistence better.

For this code, we will use the text file, `learn.txt`

```
learn is an education provider. It is based in San Francisco.

it has trained 450,000+ customers.

it offers 400+ professional courses.
```

Code: **(In Python)**

```
1. mydata= sc.textFile("learn.txt")
2. myrdd1= map(lambda s: s.upper())
3. myrdd1.persist()
4. myrdd2= filter(lambda s: s.startswith('I'))
5. myrdd2.count()
```

Description:

1. This creates an empty RDD `mydata` and links it to the file `learn.txt`.
2. This creates an empty mapped RDD `myrdd1` and links it to the RDD `mydata`.
3. This saves `myrdd1` in the memory.
4. This creates an empty mapped RDD `myrdd2` and links it to the RDD `myrdd1`.
5. This action counts the elements in `myrdd2`. The execution starts from this step and the RDDs built starts to fill and output is displayed. Now, if the last operation `myrdd.count()` is called again, Spark does not need to start the execution from step 1, it can start from step 4 because the linked RDD is saved in step 3, thus saving time and improving speed of the code.

Let us see few features of RDD persistence:

- RDD partitions can be saved in the memory and reused in later steps whenever needed.
- If any partition is lost, it is automatically restored using the original transformation steps.
- Each persisted RDD is stored on a different storage level, that is determined by the storage level object, which is passed on the persist method.

Let us now see some different kinds of storage levels for RDD persistence:

MEMORY_ONLY: It allows storage of deserialized Java objects (objects, which are restored from its sequence of bytes). If the RDD does not fit in the memory, it recomputes it.

MEMORY_AND_DISK: It allows storage of deserialized Java objects and stores the RDDs on disks.

MEMORY_ONLY_SER: It allows storage of serialized Java objects (objects, which are converted into a byte sequence). It enables better space efficiency because of storage in the form of bytes.

MEMORY_AND_DISK_SER: It allows storage of serialized Java objects and spills those partitions, which do not fit in the memory.

DISK_ONLY: It allows storage of RDDs only on disks.

MEMORY_ONLY_2, MEMORY_AND_DISK_2: Here, 2 stands for replication of every partition on two cluster nodes. Rest, it is similar to DISK_ONLY.

OFF_HEAP: Allows storage of RDD in serialized format. By storing the memory, the crash of an executor no longer leads to memory loss in cache.

Syntax on how to specify the storage level is as follows: (write on the whiteboard.)

```
rdd.persist(pyspark.StorageLevel./<name_of_the_storage_level>)
```

If storage level is MEMORY_AND_DISK_SER, it will be written as follows:

```
rdd.persist(pyspark.StorageLevel.MEMORY_AND_DISK_SER)
```

It is very crucial to decide the best storage level, therefore:

- If RDD fits with default storage level, leave it as it is.
- If default storage level cannot be left as is, use **MEMORY_ONLY_SER**.
- If fast fault recovery is needed, use replicated storage levels, **MEMORY_ONLY_2, MEMORY_AND_DISK_2**
- If environments have high memory or multiple applications, use **OFF_HEAP**.

Suppose you want to remove the persisted RDD from the memory, use the code `rdd.unpersist()`. (Write the code on the whiteboard.) Suppose you want to change the storage level, first unpersist the RDD and then, persist it again with the required storage level.

Reference: Refer to the following links for more information:
https://data-flair.training/blogs/apache-spark-rdd-persistence-caching/
https://www.javatpoint.com/apache-spark-rdd-persistence
https://www.youtube.com/watch?v=bmbFqTg7lck

Say: With this, we conclude our session.

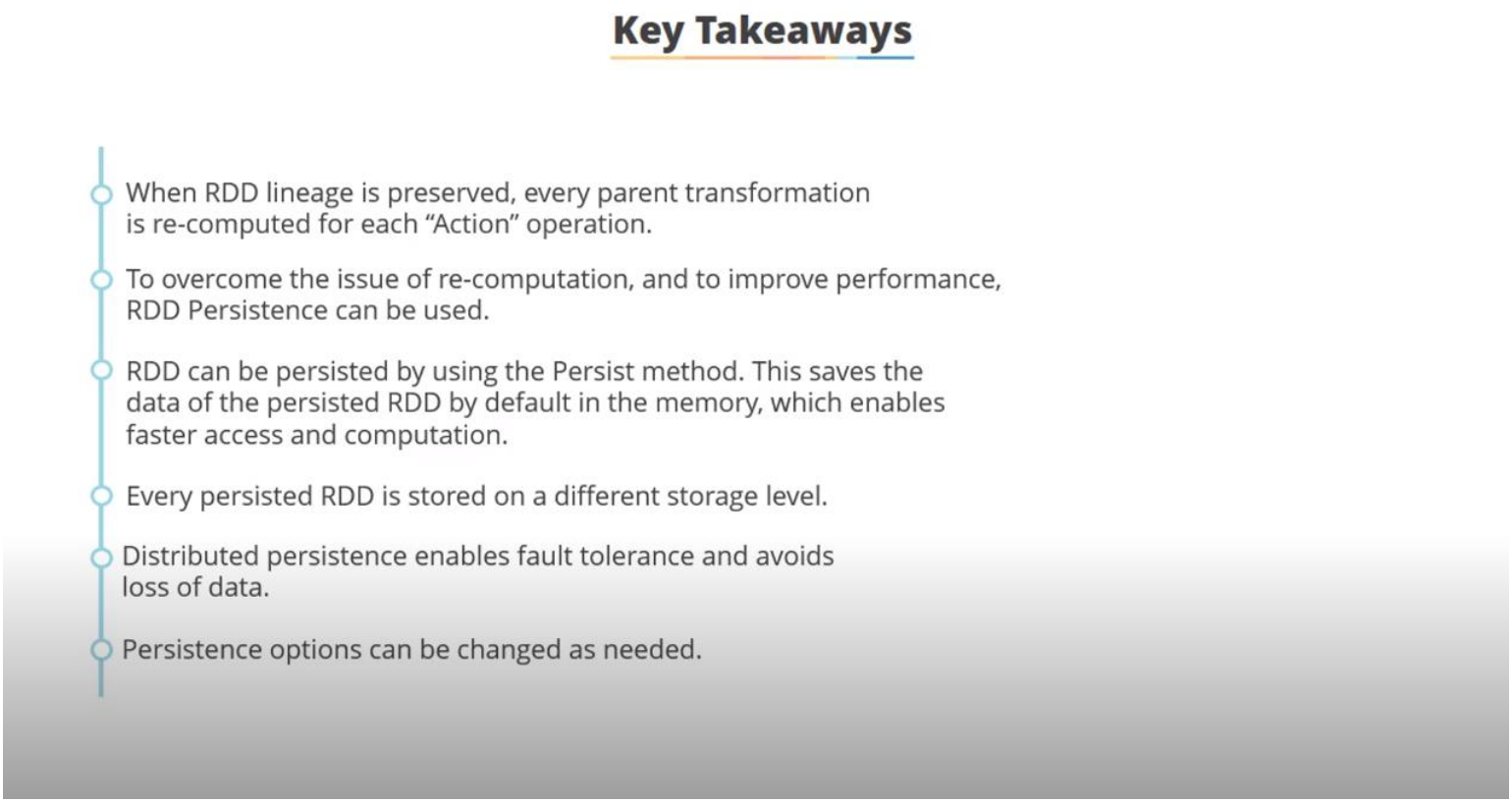
ACTIVITY 2

CLASS DURATION: 15 MINUTES

Ask the students to perform transformation operations on the given text, and write a clear pipelined code.

Current usage of the term big data tends to refer to the use of predictive analytics, user behavior analytics, or certain other advanced data analytics methods that extract value from data, and seldom to a particular size of data set. "There is little doubt that the quantities of data now available are indeed large, but that's not the most relevant characteristic of this new data ecosystem." Analysis of data sets can find new correlations to "spot business trends, prevent diseases, combat crime and so on."

KEY TAKEAWAYS



CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 14.3

FACILITATION GUIDELINES

Play Video Content: Lesson 14.3

Content: Rewrite the key takeaways of the session on the whiteboard. Tell the students that they have learnt about RDD lineage and RDD persistence.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Content: Lesson 14.4 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions.

Questions:

5. What is meant by parallel computation in Spark?

Answer: Parallel computation is the process of distributing work amongst several worker nodes to make the process faster, and optimize memory and time.

6. What is shuffling?

Answer: Shuffling is the process of redistributing data across partitions. It involves movement of data between stages.

7. What is lazy evaluation and what does it lead to?

Answer: Until an action operation is performed on the RDD, Spark does not start the execution of transformation operations. This is lazy evaluation. This leads to the formation of **RDD lineage.**

8. What is the method of RDD persistence?

Answer: When RDD persistence is called on a particular RDD, the RDD is saved in the memory at that instance. It does not involve computation from the beginning. It starts from the saved RDD, optimizing speed and improving performance.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1: Download a text file from the link: http://25.io/toau/audio/sample.txt and perform the following operations:

- Load the file into an RDD.
- Use the `map` function to split the RDD according to '.'
- Persist the RDD.
- Count the number of elements in the RDD.
- Find the lineage of this RDD.

Assignment #2: Large companies such as Netflix, Facebook, and Uber are using Apache Spark in a big way.

Here are few interesting links that you should read to get a closer look on how such companies are using Apache Spark.

https://medium.com/netflix-techblog/netflix-at-spark-ai-summit-2018-5304749ed7fa
https://www.knowledgehut.com/blog/big-data/spark-use-cases-applications
https://www.youtube.com/watch?v=IGfvVd-v3P8

Session 12

Session Objectives:

- Describe the kinds of processing and analysis that Spark supports
- Describe the applications of machine learning
- Explain how GraphX and MLlib works

Video Content: Lesson 15

QUICK RECAP

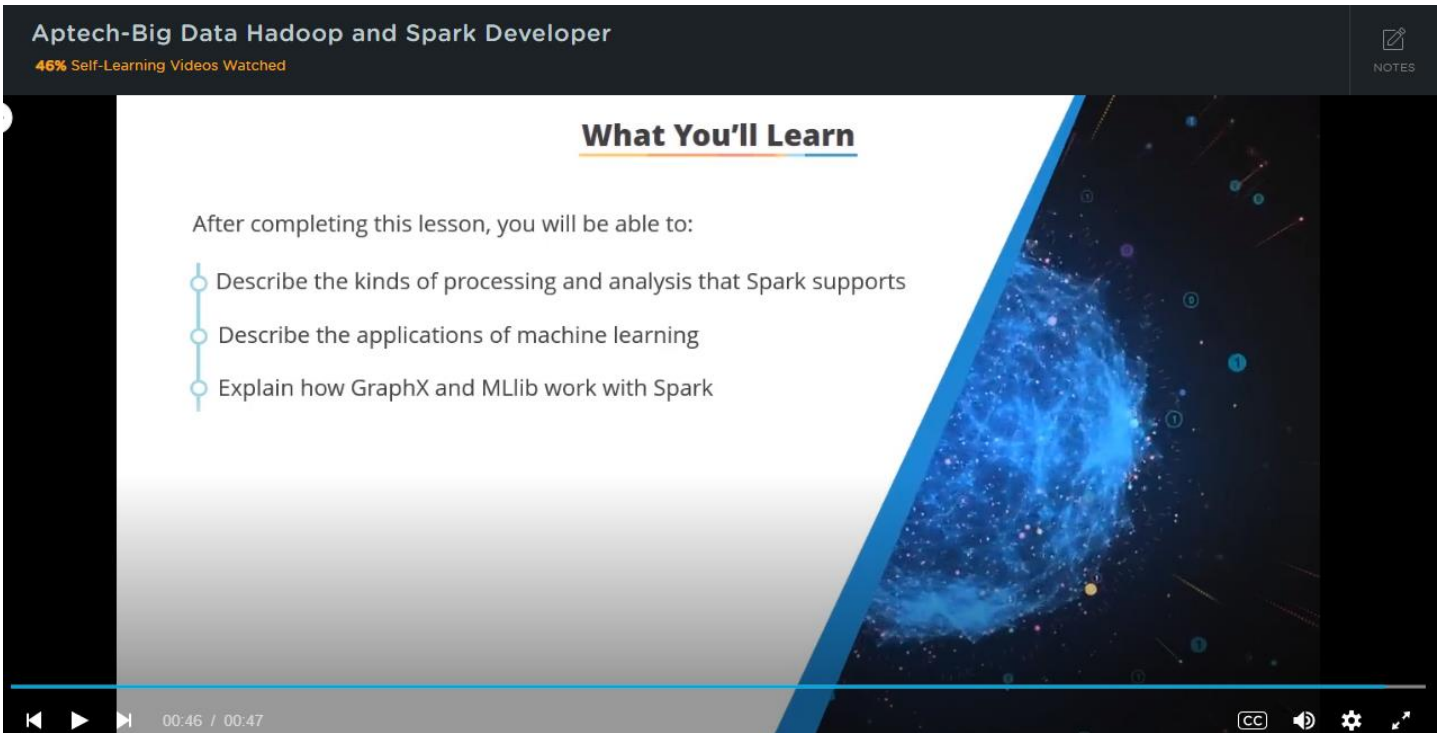
CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

LESSON OBJECTIVES



CLASS DURATION: 2 MINUTES

VIDEO DURATION: LESSON 15.1

FACILITATION GUIDELINES

Play Video Content: Lesson 15.1

Content: After playing the video, repeat all the points and tell students that in this session, they will learn about machine learning and graph component of Spark.

Say: Let us begin.

SPARK: AN INTERATIVE ALGORITHM

CLASS DURATION: 15 MINUTES

VIDEO DURATION: LESSON 15.2

FACILITATION GUIDELINES

Play Video Content: Lesson 15.2

Content: Spark is an open-source cluster computing architecture. It provides 100 times faster performance than Hadoop's MapReduce. Spark includes partitioned framework and provides faster computation. In addition, because it allows iterative querying over data, it is suitable for implementing machine learning algorithms on big data. Spark can predict outcomes such as 'how likely is it that a person will repay a loan?', 'spam detection of email', 'product requirement of customers', and so on.

Let us look at an example of machine learning algorithm known as **Page Rank** algorithm. This algorithm is used to rank Web pages based on number of other Web pages to which it is linked.

Consider four Web pages henceforth called as A, B, C, and D for the purpose of our example. Our approach in this algorithm is to assign numerical weights (weightage) to each element of hyperlinked set of documents. This is done with the purpose of measuring its relative importance within the set. This means if you have a set of hyperlinked Web pages (in our case A, B, C, and D), then you have to measure importance or ranking of each Web page in relation to other Web pages. This algorithm outputs a probability distribution to see how many times a Web page is linked to another page. Hence, let us continue our example of a small set of universe of four Web pages A, B, C, and D. The approach of this method is that pages that have links to other pages transfer their weights (which were assigned to them at the first iteration) to those pages equally. Let us see this in action.

If the only links in the system are from pages B, C, and D to A, that means pages B, C, and D have hyperlinks to page A. Assume probability distribution between 0 and 1, therefore, initially in the first iteration, every page has a probability of 0.25 (divided equally amongst the four pages). Now, since B, C, and D are linked to A, therefore, in the second iteration, all the three pages transfer their probabilities to A, that means:

P(A) = P(B)+P(C)+P(D) = 0.25+0.25+0.25 = 0.75

(P stands for probability.)

Now, suppose B has a link to A and C, and C has a link to A, and D has a link to A, B, and C, then to calculate probability of A and B would transfer half of its probability to A and half to C. Then, C would transfer its whole probability to A. D would transfer its one-third probability to A, one-third to B, and one-third to C. Therefore,

P(A)= (P(B)/2) + P(C) + (P(D)/3)= 0.125+0.25+0.083= 0.458

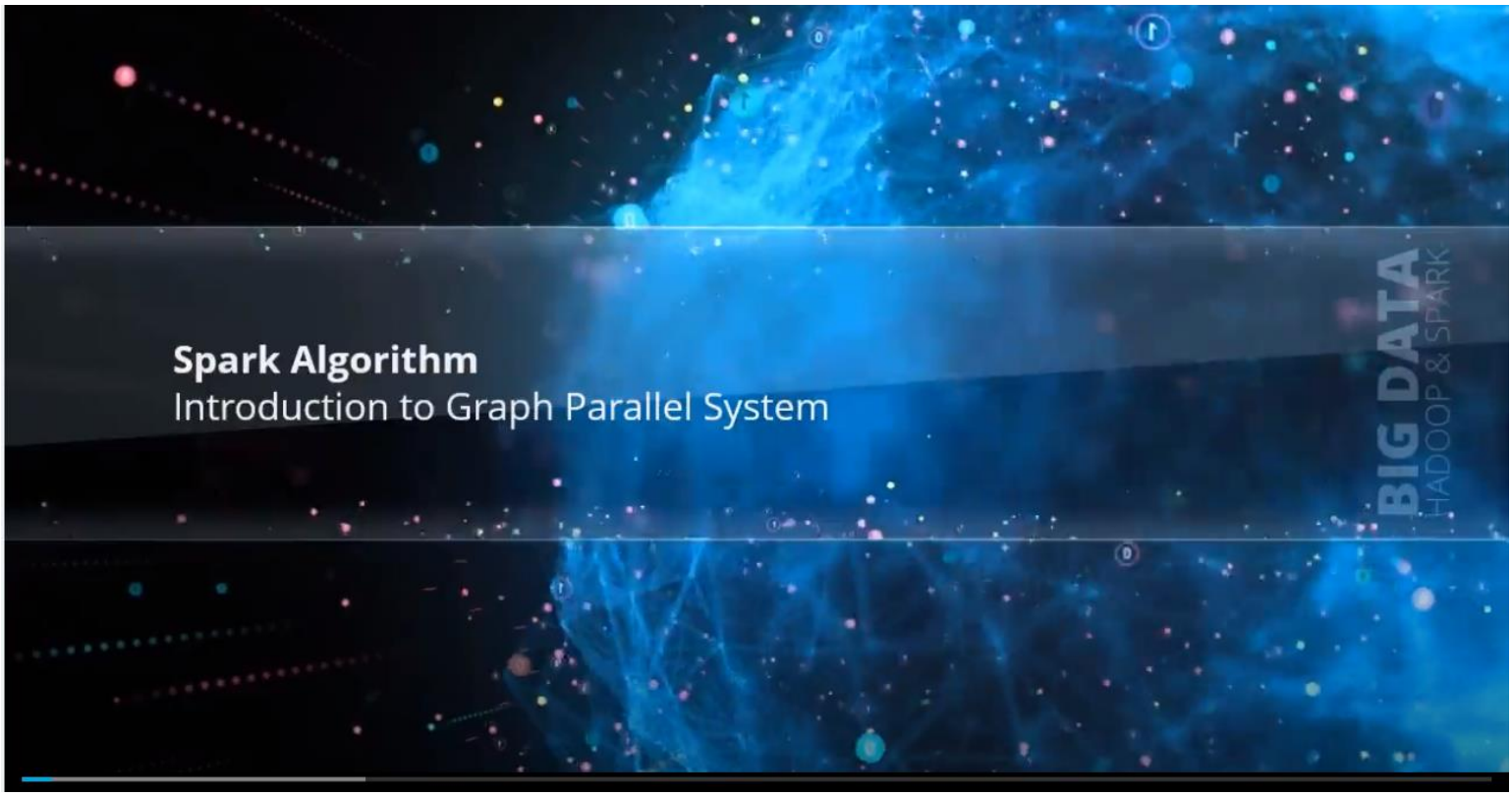
Similarly,

P(B)= P(D)/3 = 0.083

P(C)= (P(B)/2) + (P(D)/3) = 0.125 + 0.083= 0.208

Say: It is time to move to the next topic.

INTRODUCTION TO GRAPH PARALLEL SYSTEM



CLASS DURATION: 12 MINUTES

VIDEO DURATION: LESSON 15.3

FACILITATION GUIDELINES

Play Video Content: Lesson 15.3

Content: Graph parallel processing system solves large-scale parallel computation. Graph is an abstract data type that is meant to implement the undirected graph and directed graph, known as graph theory. It helps in easy understanding of problem's framework. Graphs help in targeted advertising, identifying communities, and deciphering meaning of documents. Targeted Advertising refers to placing advertisements according to demographics and customer behavior. You can see these ads shown by Google while surfing the Internet.

Give an example to illustrate targeted advertising: If a user is browsing something related to real estate on one Web page, when the user opens a new tab or scrolls down on the same page, he/she might be shown ads for real estate. Another example illustrating demographics based ads is that of a Canadian bank that targets Asian-origin Canadians with specific ads.

Deciphering the meaning of documents refers to transforming complex documents into graphs to understand them in an easier way.

Some large-scale distributed graph-parallel frameworks include the following:

GraphLab: It is an open-source project using Apache license. It is capable of doing machine learning tasks and broad range of data mining tasks. As the amount of collected data and computing power grows, datasets nowadays do not fit in single core machines. Efficient parallel algorithms for handling large-scale data are required. The GraphLab framework is a parallel programming abstraction targeted for iterative graph algorithms.

Giraph: Apache Giraph is an open-source project of Apache to perform graph processing on big data. It utilizes Apache Hadoop's MapReduce method to process graphs. Its functionality is almost same as GraphLab.

PowerGraph: PowerGraph abstraction uses the internal structure of graph programs to address the challenges, which cannot be solved by GraphLab or Giraph. The graph parallel system used by GraphLab and Giraph rely on each vertex of graph having small neighborhood to maximize parallelism. However, graphs that are derived from real world phenomena such as social networks typically have small subsets of vertices, which connect to large fraction (part) of graph, thus making the computation difficult. Therefore, PowerGraph was introduced to exploit the structure of these graphs and increase the computation speed.

GraphX: It is a graph computation system that runs on the framework of parallel computation. It is a real-time processing framework. It extends the RDD into Resilient Distributed Graph (RDG), a graphical form of RDD). It analyzes the process step-by-step. It is the fastest graph systems that retains Spark's flexibility.

Some limitations of graph parallel system are as follows:

- Each graph parallel system framework presents a different graph computation.
- These frameworks depend on different runtimes.
- These frameworks cannot resolve the data Extract Transform Load (ETL) (that is data ingestion, data transformation, and loading back of data) and cannot decipher process issues.

Reference: Refer to the following links for more information:

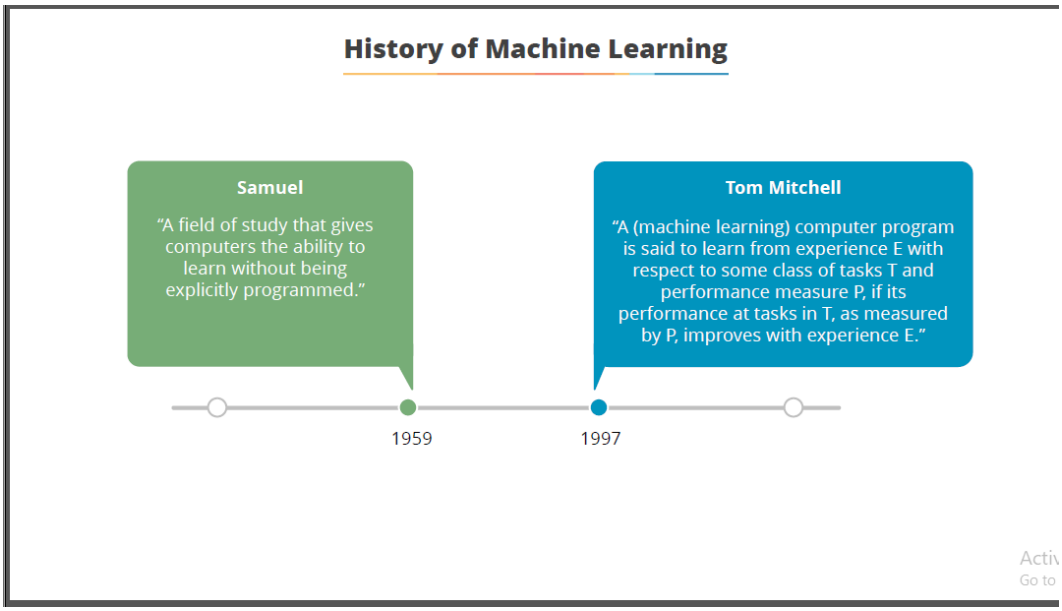
https://www.usenix.org/system/files/conference/osdi12/osdi12-final-167.pdf

https://en.wikipedia.org/wiki/Graph_(abstract_data_type)

https://spark.apache.org/graphx/

Say: It is time to move to the next topic.

INTRODUCTION TO MACHINE LEARNING



CLASS DURATION: 20 MINUTES

VIDEO DURATION: LESSON 15.4

FACILITATION GUIDELINES

Play Video Content: Lesson 15.4

Content: Machine Learning now a days has become a vital part of information technology. It refers to a trivial word called **prediction**. It is a field of study that gives computers the ability to learn from past experience and predict the future outcomes. Machine Learning comprises several algorithms, which are applied on data to ascertain predictions. For example, machine learning is used for credit card fraud detection. A model is trained from the past data and is applied on the new data to get the outcome. It is also used in health care system. Cancer can be predicted using the algorithms or how likely is the patient prone to cancer.

Reference: https://www.expertsystem.com/machine-learning-definition/

Let us look at certain common terminologies of machine learning.

Vector Feature: This includes numbers to describe an object. For example, consider you have a dataset of a school record. Here, names of students are the objects. Then, all these columns, which describe the attributes of a student such as age, class, weight, height, marks, and so on will be features. A vector consisting of its entries is called vector feature. This is also known as feature space.

Samples: Sample is a subset of any data set drawn when the processing on it is required.

Labeled Data: This refers to the data that includes the classification results such as data of credit card transactions can include categories such as fraud or not fraud, data of email transactions with a label column comprising spam or not spam, and so on.

Now, let us see a small example to understand functioning of machine learning.

Consider that you have a bucket of oranges and apples. (Write them on the whiteboard.)

Features of an orange are:

- Color-orange
- Shape-round
- Diameter-5 cm

Features of an apple are:

- Color-red
- Shape-disturbed round
- Diameter- 6 cm

Data given can be considered as a framework of any past data of oranges and apples having features as Color, shape, and Diameter, which have varied entries. To classify the fruits as orange or apple, a model is built on the past data of oranges and apples, making it learn through these features and then new data is fed into the model, which gives the outcome as orange or apple for each entry.
Machine learning has wide range of applications. These days, you cannot think of any new technology that does not make use of machine learning. Let us highlight a few applications.

Speech Recognition	Many users are now familiar with Google Assistant, Apple's Siri, and Amazon's Alexa. All these are examples of speech recognition products, which take a human's voice as input and predict speech.
Effective Web Search	Google is the largest search engine in the market. If you plan on to search on any topic, it automatically shows you recommendations of what your topic could be as you type 2-3 letters.
Recommendations System	Amazon, Netflix, and Etsy are all companies that use their recommendation systems for the customers for a better shopping/viewing experience. Amazon and Etsy keep track of items bought by a customer and recommends them similar items that can be bought. Netflix keeps track of shows or movies watched by a user and recommends similar genre to them. (Ask students if they have experienced this in Netflix.)
Computer Vision	Image processing or image detection is the best example for computer vision. It is fed with hand-labeled images. A training dataset is built and a model is developed based on this. It then predicts the labels to new images being tested.
Information Retrieval (IR)	IR is the activity of obtaining information system resources that are relevant to information needs of users. For example, search engines.
Fraud Detection	Fraud detection is a system to detect frauds related to any service. For example, credit card fraud detection, email fraud detection, fraud calls detection, and so on.

Applying machine learning on big data brings Spark MLlib into action. Spark's MLlib component allows one to apply machine learning on a continuous stream of data with numerous iterations thus, enabling it to work fast. The process is the same as for the normal data, that is, the data which is not

continuous and is fully available once at a time. First, data is fed into the system that requires cleaning and manipulation, then machine learning algorithms are applied on this data, and a model is trained. Then, this model is tested on future data and is deployed. The feedback is then sent to the user.

Reference: https://www.ubuntupit.com/top-20-best-machine-learning-applications-in-real-world/

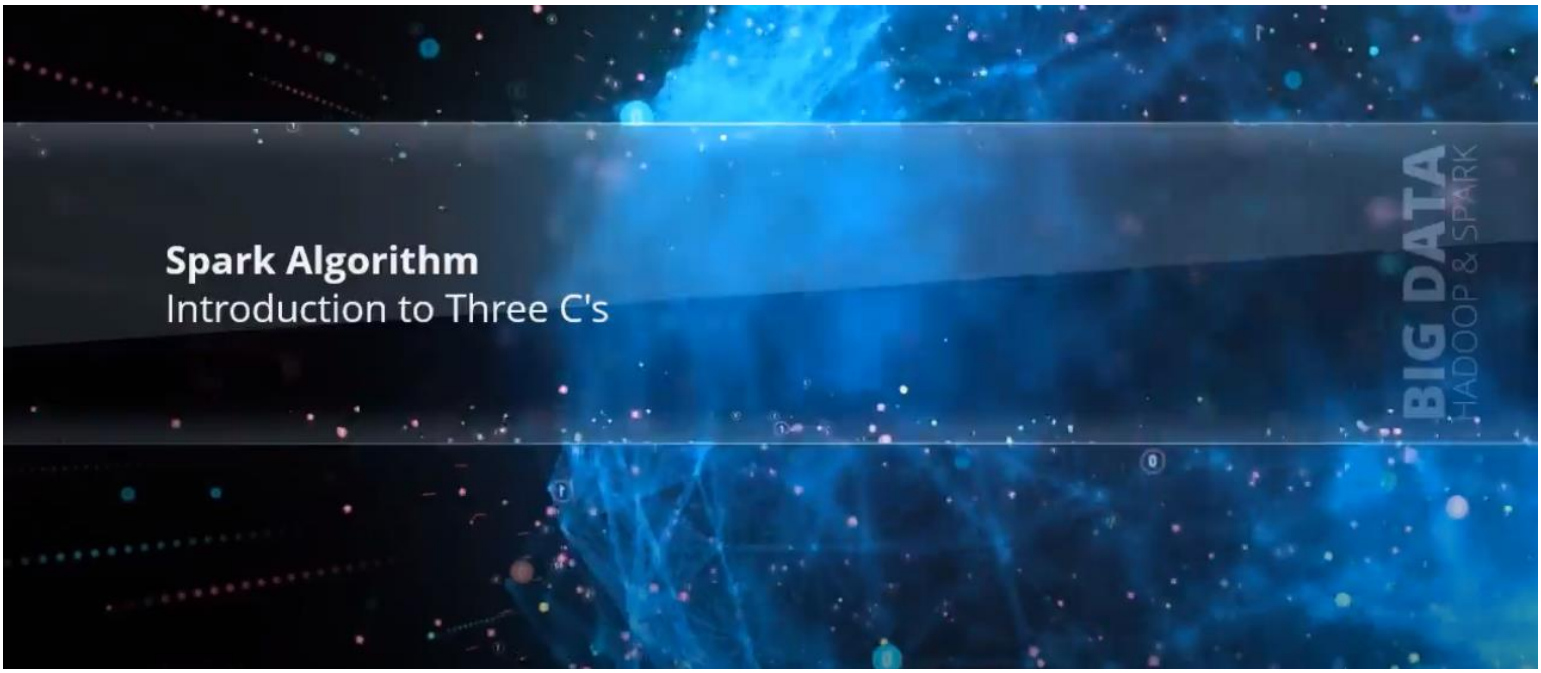
Say: Let us now move to the next topic.

ACTIVITY 1

CLASS DURATION: 12 MINUTES

Netflix is world's leading Internet entertainment service. Tell students to research on how Netflix implements machine learning. Ask any two students at random to come in front of the class and present results of their research.

INTRODUCTION TO THREE C'S



CLASS DURATION: 14 MINUTES

VIDEO DURATION: LESSON 15.5

FACILITATION GUIDELINES

Play Video Content: Lesson 15.5

Content: Spark's MLlib consists of three main algorithms, Classification, Clustering, and Collaborative Filtering. Let us look at them one by one.

Classification: Classification is a method to classify objects into categories such as Yes, No, Rich, Poor, Male, Female, and so on. Classification can be of three types:

- ***Binary Classification***: This means classification in two categories. Methods to do this include logistic regression, linear support vector machines, decision trees, random forests, and so on.
- ***Multiclass Classification***: This means classification in more than two categories. Methods to do this include logistic regression, decision trees, Naïve bayes, and so on.
- ***Regression***: This means assigning continuous real values to objects. Method to do this include linear least squares, lasso, ridge regression, gradient boosted trees, and so on.

Clustering: This is a method of classifying the objects on the basis of similarities. Retail stores groups their customers according to common likes or dislikes so that it can predict what its customer is likely to buy. Amazon groups the data and buying patterns of its customers. This is why it is able to predict product suggestions that a customer would likely to buy. Similarly, Facebook groups its users' data that is why you see the section of friend's suggestion in your profile.

The clustering approach works on the formation of groups of similar objects or the objects that are close to one another. One of the popular methods to do clustering is called k-means.

K-means assigns the data points to its nearest centroid and forms the cluster.

1. It first chooses k centroids randomly from a dataset, where k is given by a user such as 3, 4, 6, and so on (the number of clusters users want).
2. It then assigns each point to its nearest centroid by calculating distance between points and centroids.
3. It then calculates mean of all data points for each cluster and resulted mean becomes the new centroid.
4. It then starts from step -1 again and keeps on iterating until the distance of all the points from their centroid is minimized and distance between the clusters is maximized.

Collaborative filtering: Collaborative filtering is used by recommender systems for customer benefits. It is a method for making automatic predictions about the interests of users by keeping track of preferences or their tastes. Specifically, it is to predict user preference for a set of items based on past experience. This method uses customers' historical preference on a set of items. The main assumption is that customers, who have agreed in the past trend, will also agree in the future. It is usually expressed as two categories, explicit rating, and implicit rating. **Explicit rating**, as the name suggests is the rating given to any item by the customer. **Implicit rating** refers to indirect preferences for items, such as page views, purchase records, music preferences, and so on.

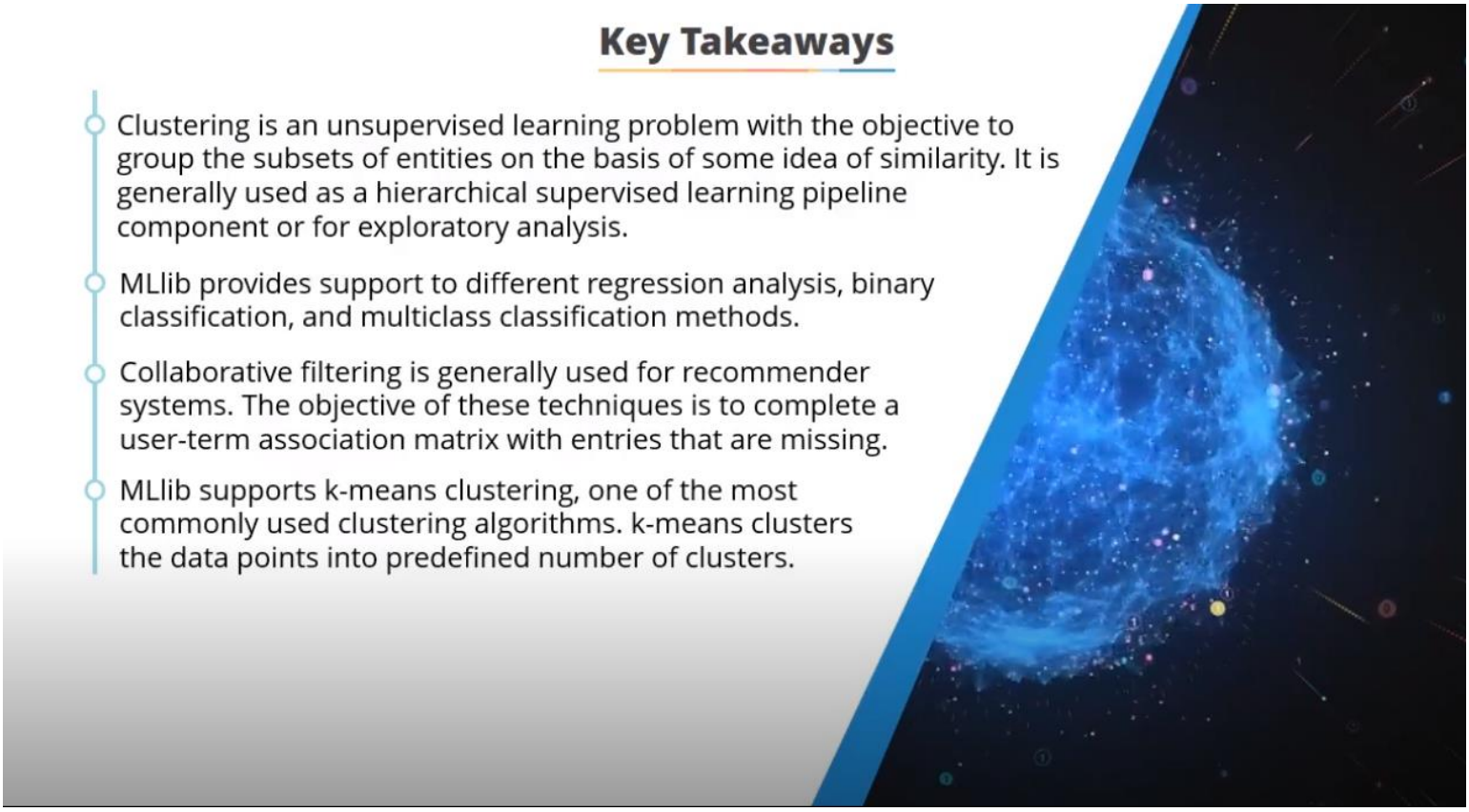
Say: With this, we conclude our session.

ACTIVITY 2

CLASS DURATION: 15 MINUTES

Divide the students into three groups. Assign each group one of the three Cs of machine learning, that is, Classification, Clustering, and Collaborative filtering. Ask them to think and discuss about some real life examples where the technique given to them can be applied.

KEY TAKEAWAYS



CLASS DURATION: 4 MINUTES

VIDEO DURATION: LESSON 15.6

FACILITATION GUIDELINES

Play Video Content: Lesson 15.6

Content: Rewrite the key takeaways of the session on the whiteboard. Tell the students that they have learnt about machine learning in this session.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Content: Lesson 15.7 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions.

Questions:

1. What is page rank algorithm?

Answer: This algorithm ranks Web pages on the basis of the number of other Web pages it is linked to.

2. What is graph parallel processing?

Answer: Graph parallel processing system is an approach to solve large-scale parallel computation.

3. What is machine learning?

Answer: Machine learning gives computers the ability to learn from past experience and predict future outcomes. Note: If student gives a different but correct answer, you can accept that as well.

4. What is GraphX?

Answer: It is a graph computation system that runs on the framework of parallel computation. It is a real-time processing framework.

5. What is collaborative filtering?

Answer: Collaborative filtering is a method for making automatic predictions about the interests of user by keeping track of preferences or their tastes.

6. What is clustering?

Answer: It is a method of classifying the objects on the basis of similarities.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1: Elaborate on the ten different areas where machine learning is used extensively.

Assignment #2: Read more about k-means algorithm through this link:

https://towardsdatascience.com/clustering-using-k-means-algorithm-81da00f156f6

.

Session 13

Session Objectives:

- Explain the importance and features of Spark SQL
- Describe the methods to convert RDDs to DataFrames
- Explain a few concepts of spark SQL
- How to load existing data into a DataFrame
- How to convert from DataFrames to Pair RDDs

Video Content: Lesson 16

QUICK RECAP

CLASS DURATION: 6 MINUTES

VIDEO DURATION: NONE

FACILITATION GUIDELINES

Greeting: Welcome the students back to the course. Ask them if the previous session was a great learning experience, then, give them a brief recap of topics covered in the last session. Ask them if they have any doubts regarding these topics. After clearing the doubts, inform them about the topics that would be covered in this session.

INTRODUCTION

CLASS DURATION: 20 MINUTES

VIDEO DURATION: LESSON 16.1

FACILITATION GUIDELINES

Play Video Content: Lesson 16.1 till 00:37

Content: After playing the video, rewrite all the points on the whiteboard and tell the students that in this session, they will learn about an important component of Spark, namely, Spark SQL.

Say: Let us begin.

Play Video Content: Resume Lesson 16.1

Content: Spark Structured Query Language (SQL) is yet another very important component of Spark. SQL is a language used in programming and designed for database management system to store data in the form of rows and columns. It allows to query over data and work with pattern of rows and columns called DataFrame. Consider a small data named 'Record'.

(Write this data on the whiteboard.)

Name	Age	Gender	Rank
Maven	12	M	1
Sandria	18	F	2
James	11	M	3
Sam	10	F	4

A SQL query can be like-

```
SELECT * FROM Record WHERE Age>11;
```

The output will be: (Write this data on the whiteboard.)

Name	Age	Gender	Rank
Maven	12	M	1
Sandria	18	F	2

Description: In the example query, SELECT is used to query over the data, the fields which the user needs; in our data we selected * which stands for 'all'. That means we want to select the whole data according to the age condition, that is >11.

Reference: Refer to following links for more information:
https://www.codecademy.com/articles/sql-commands
https://www.tutorialspoint.com/spark_sql/spark_sql_quick_guide.htm
https://intellipaat.com/blog/what-is-spark-sql/

Spark SQL provides DataFrame API (commands) that can be used for data manipulation and transformation. It allows to run query on any data and returns a DataFrame as a result. It can be used with languages such as Python, R, Java, Scala, and so on.
Let us see some features of Spark SQL:

- **Compatibility with HIVE**: Spark SQL is compatible with HIVE queries and its user-defined functions.
- **SQL queries support**: It mixes SQL queries with Spark programs.
- **Components support**: It includes columnar storage.
- **Spark Engine Inclusion**: It allows you to apply many queries on thousands of nodes.
- **Comprehensive**: It does not require a different storage engine to store historical data.

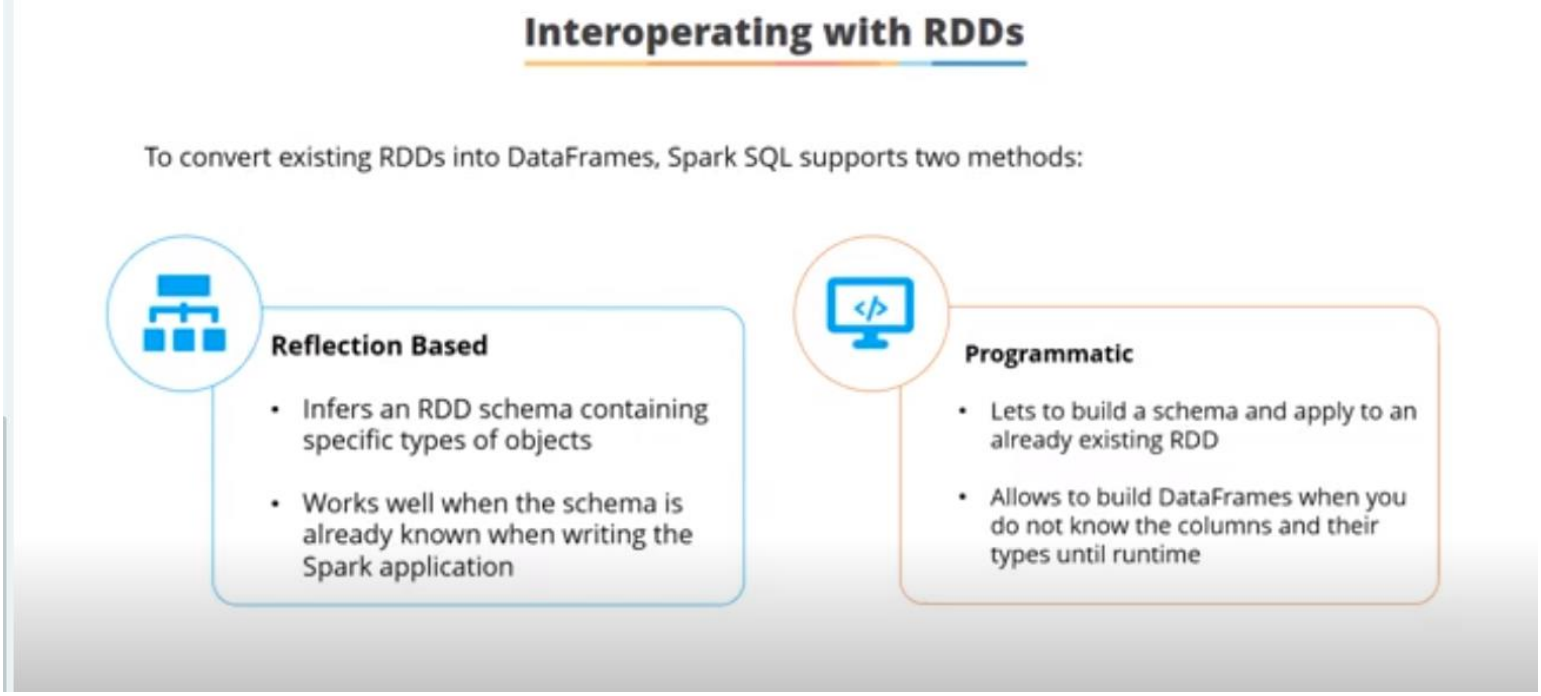
As already mentioned, Spark SQL query returns DataFrame, which is basically two-dimensional data and they are very similar to spreadsheets. They have rows and columns. In general, the columns have names. Each column has a different data type, but the whole column keeps the same type and each row contains a record.

SQL context is similar to Spark context. It lives essentially underneath the Spark context. It is the SQL context that handles DataFrame. You can print schema of every DataFrame. A DataFrame has a schema similar to a database. The schema describes the name and the type of each column. A DataFrame is nothing more than an RDD of rows and you have some high-level information that is the schema. If you perform the collect operation on the RDD or the DataFrame, you basically get the same result. It is often easier to define the schema explicitly, so instead of creating a lot of rows and then making it into a DataFrame, we can just define one schema and then give it a standard RDD.

Reference: Refer to following links for more information:
https://spark.apache.org/docs/1.1.0/sql-programming-guide.html
https://medium.com/@mrpowers/adding-structtype-columns-to-spark-dataframes-b44125409803
https://www.youtube.com/watch?v=1aCpLUFTja0

Say: Let us move to the next topic.

INTEROPERATING WITH RDD



CLASS DURATION: 20 MINUTES

VIDEO DURATION: LESSON 15.2

FACILITATION GUIDELINES

Play Video Content: Lesson 15.2

Content: In this topic, we will specifically learn methods to convert RDD to DataFrame using the schema. There are two methods for this:

Reflection-based: It results into the schema containing specific types of objects. It works well when the schema of an RDD is already known. It automatically converts RDD containing case classes into a DataFrame. Case class is a regular class built in Spark and is good for modeling data, which is immutable. This means that data cannot be edited.

Ask students to read more about case classes through this link:

https://data-flair.training/blogs/scala-case-class/

Let us see this method in action. Consider the following code snippet: (Write the snippet on the whiteboard.)

```
1. val SqlContext = new org.apache.spark.sql.SQLContext(sc)
2. import sqlContext.implicits._
3. case class Person(name: String, age:Int)
4. val people = sc.textFile("path_to_the_text_file").map(_.split(",")).toDF()
5. people.registerTempTable("people")
6. val teenagers= sqlContext.sql("SELECT name, age FROM people WHERE age>=13 AND age <=19")
```

Description:

1. Here, a variable `SqlContext` is defined which stores the entry of spark SQL.
2. Here, `implicits` library of spark SQL is loaded.
3. The `case class` is defined as `Person` containing two fields, `name` and `age`.
4. Here, `people` is initialized as an RDD with some text file and a `map` function is applied for some transformation on RDD (if required). `toDF()` is the function to convert the RDD into a DataFrame.
5. Here, the DataFrame created is converted or registered as a table.
6. Here, a simple SQL query is run on the table and the result is assigned to the variable `teenagers`.

***Programmatic-based**:* It is used when the case class cannot be created ahead of time that is when the schema is specified at the run time. Here are some steps to do this: (Write these steps on the whiteboard.)

- Create an RDD of rows from the original RDD.
- Create a schema matching the structure of rows of the RDD created.
- Apply the schema to the RDD of rows.

Let us see this in action. Consider the following code: (Write the snippet on the whiteboard.)

```
1. val SqlContext = new org.apache.spark.sql.SQLContext(sc)
2. val people = sc.textFile("path_to_the_text_file")
3. val schemaString = "name age"
4. import org.apache.spark.sql.Row;
5. import org.apache.spark.sql.types.{StructType, StructField, StringType};
6. val Schema = StructType( schemaString.split("").map(fieldname => StructField(fieldname,
   StringType, true)))
7. val rowRDD = people.map(_.split(""))
8. val peopleDF= sqlContext.createDataFrame(rowRDD, schema)
```

Description:

1. Here, a variable `SqlContext` is defined that stores the entry of Spark SQL.
2. `people` is initialized as an RDD with some text file.
3. `schema` is encoded as a string and assigned it to the variable `schemaString`.
4. Here, the Spark SQL library `Row` is imported.
5. Here, the Spark SQL necessary libraries are imported.
6. Schema is generated based on the string of schema.
7. Here, the records of the RDD people are converted into RDD of rows.
8. Schema is applied to the RDD and the DataFrame is created.

Reference: Refer to following links for more information:

https://spark.apache.org/docs/latest/sql-getting-started.html#interoperating-with-rdds
https://www.tutorialspoint.com/spark_sql/spark_sql_dataframes.htm

Say: Time for some activities!

ACTIVITY 1

CLASS DURATION: 15 MINUTES

Divide students in three groups and assign one of the three tasks to each group.

Task 1: Ask students to write four operations on single RDD.

Task 2: Ask students to write four operations on multi RDD.

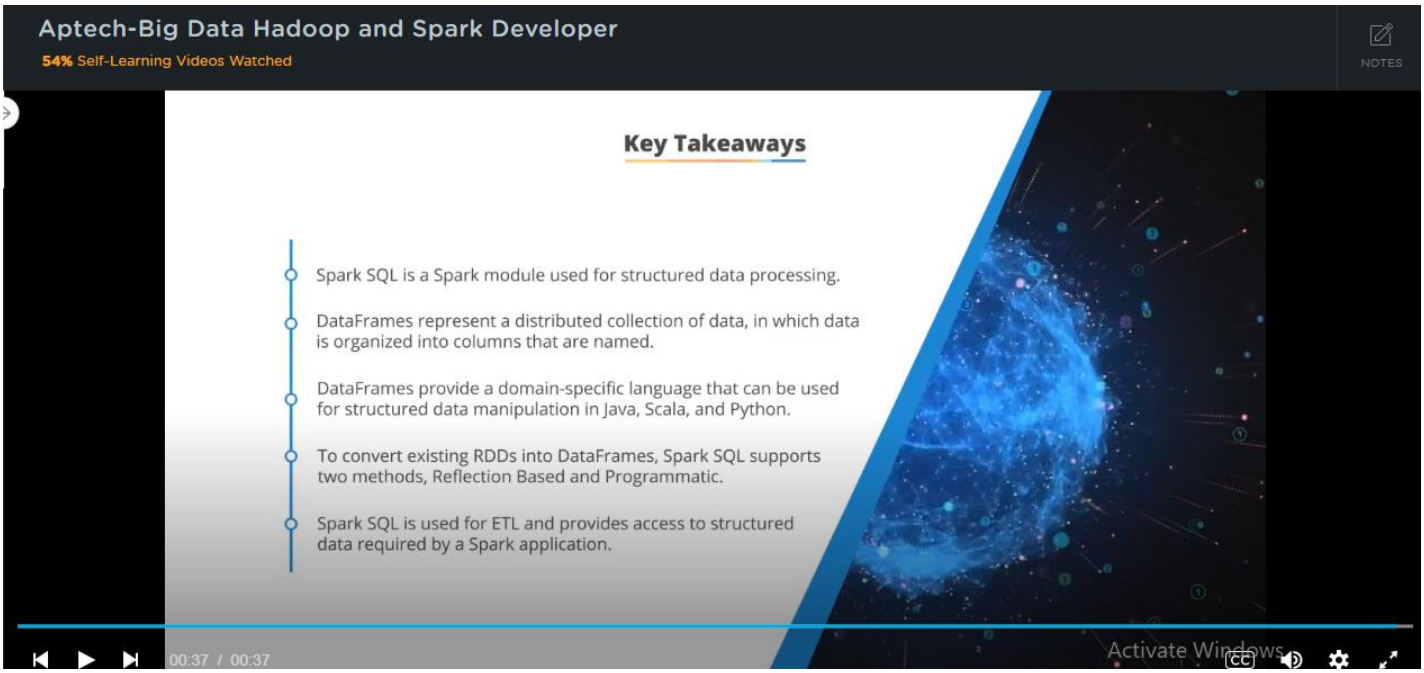
Task 3: Ask students to write four operations on paired RDD.

ACTIVITY 2

CLASS DURATION: 36 MINUTES

Illustrate with a diagram how Uber uses Apache Spark to optimize customer experience.

KEY TAKEAWAYS



CLASS DURATION: 3 MINUTES

VIDEO DURATION: LESSON 16.3

FACILITATION GUIDELINES

Play Video Content: Lesson 16.3

Content: Rewrite the key takeaways of the session on the whiteboard. Tell the students that they have learnt about Spark SQL in this session.

QUIZ

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Content: Lesson 16.4 (QUIZ). Show each question and give students some time to recall the concepts and attempt the question. Once they have finished answering the question, show them the correct answer and explanation. Repeat this process for all the questions in the QUIZ. If students have queries regarding the QUIZ questions, discuss and resolve the queries.

You may choose to ask few more questions:

Questions:

1. What is Spark SQL?

Answer: Spark SQL is a component of Spark. It provides DataFrame API (commands) that can be used for data manipulation and transformation.

2. What are the features of Spark SQL?

Answer: Spark SQL is compatible with HIVE. It supports SQL queries and components. It is comprehensive and includes Spark engine.

3. What are the two ways for converting RDD to DataFrame?

Answer: Reflection-based and programmatic-based are the two ways of converting RDD to DataFrame.

QUERIES REGARDING SESSION:

CLASS DURATION: 10 MINUTES

FACILITATION GUIDELINES

Ask students if they have any queries regarding the session and resolve the queries.

Assignment Sheet

Assignment #1: Go to link https://support.oneskyapp.com/hc/en-us/articles/208047697-JSON-sample-files. Download the file 'example_2.json'. Use this to load into the Spark and convert it to a DataFrame. You can use either of the two methods learnt.