MONASH University

Session 01 Tutorial

- Introduction
- VM Installation and Setup
- Python Basics
- Spark Introduction and Tutorial Tasks

GROUP OF EIGHT AUSTRALIA

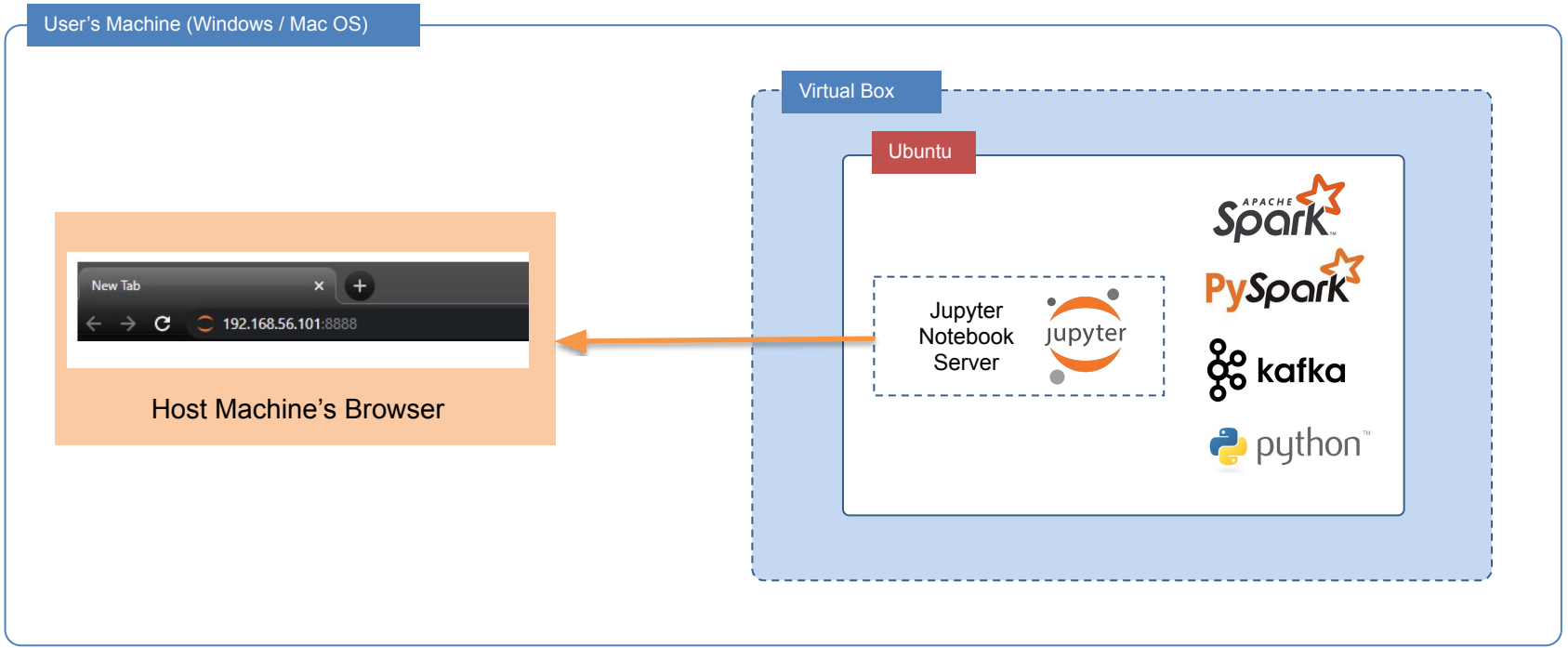
Introduction

MONASH University

General Information

- Tutorial Instructions
- Interactions and Breakout rooms
  - Asking questions and discussions
- Lab Tasks
  - Lab Task Submission
- Tutorial Attendance
  - Attending allocated tutorials

VM Download and Setup



What is Apache Spark?

MONASH University

- Cluster computing platform designed for *fast* and *general purpose.*
- Spark extends the popular MapReduce model to efficiently support interactive queries and stream processing.
- One of the main feature Spark offers for speed is the ability to run computations in memory.
- Spark is designed to be highly accessible, offering simple APIs in Python, JAVA, Scala, and SQL, and rich built in libraries.

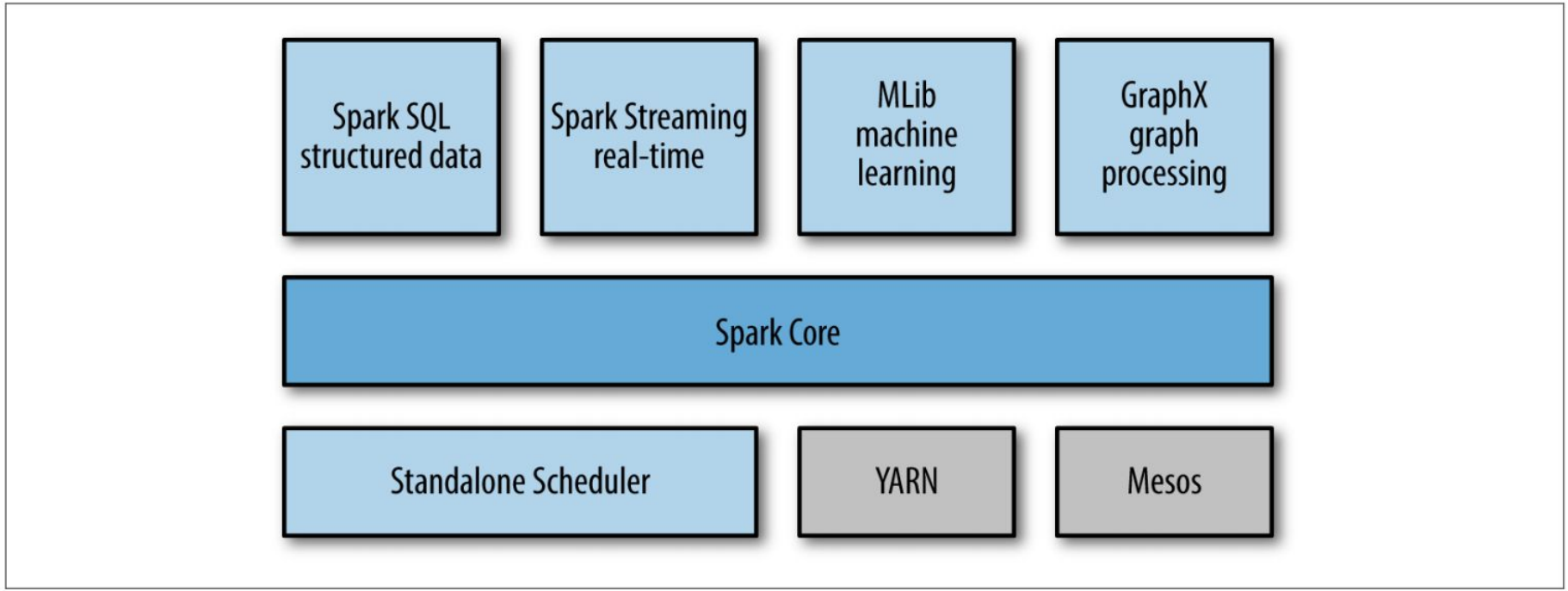
A Unified Stack

MONASH University

- Multiple closely integrated components.
- Core engine of Spark is both fast and general-purpose and it powers multiple higher-level components specialised for various workloads, such as SQL or machine learning.
- These components are designed to interoperate closely, that lets you combine these components like libraries in a software project.

A Unified Stack

MONASH University



A Unified Stack - Advantages

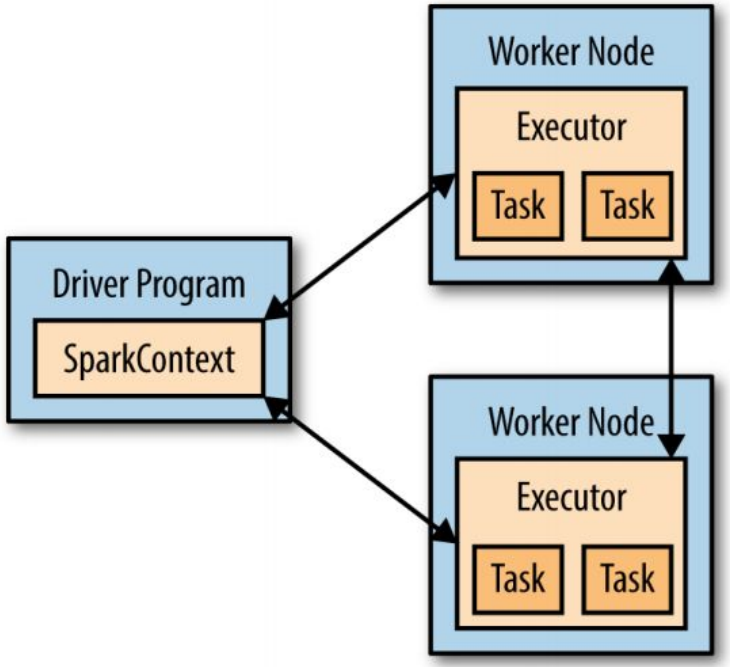
MONASH University

- **What are the advantages of Unified Stack?**
  - All libraries and higher level components in the stack benefit from improvements at the lower layers.
  - The cost associated with running the stack are minimized. Instead of running 5 -10 different independent software systems, an organization needs to run only one.
  - Provides an ability to build applications that can seamlessly combine different processing models.

Core Spark Concepts

MONASH University

- Driver Program
  - Launches various parallel operations on a cluster.
  - It contains your applications main function and defines distributed datasets on the cluster, then applies **operations** on them.
  - Driver programs access Spark through a `SparkContext(sc)` object, which represents a connection to a computing cluster.
- Worker Node (Executor)
  - To run the **operations**, driver programs typically manage a number of nodes called executors.



Programming with RDDs

MONASH University

- RDD Operations
  - RDDs support **two types** of operations:
    - *transformations* and
    - *actions*.
  - **Transformations** are operations on RDDs that return a new RDD, such as `map()` and `filter()`.
  - **Actions** are operations that return a result to the driver program or write it to storage, and kick off a computation, such as `count()` and `first()`.

Programming with RDDs: Transformations

MONASH University

- RDD Operations
  - **Basic RDD Transformation on an RDD containing {1, 2, 3, 3}**

Transformation	Purpose	Example	Result
map(*func*)	Apply a function to each element in the RDD and return an RDD of the result.	rdd.map(lambda x : x + 1)	{2, 3, 4, 4}
filter(*func*)	Return an RDD consisting of only elements that pass the condition.	rdd.filter(lambda x: x != 1)	{2, 3, 3}
flatMap(*func*)	Apply a function to each element in the RDD and return an RDD of the contents of the iterators.	rdd.flatMap(lambda x : range(x,4))	{1, 2, 3, 2, 3, 3, 3}
distinct([*numPartitions*])	Remove duplicates.	rdd.distinct()	{1, 2, 3}
sample(*withReplacement, fraction, seed*)	Sample an RDD, with or without replacement.	rdd.sample(false, 0.5)	Nondeterministic

See more here: https://spark.apache.org/docs/latest/rdd-programming-guide.html

Programming with RDDs: Actions

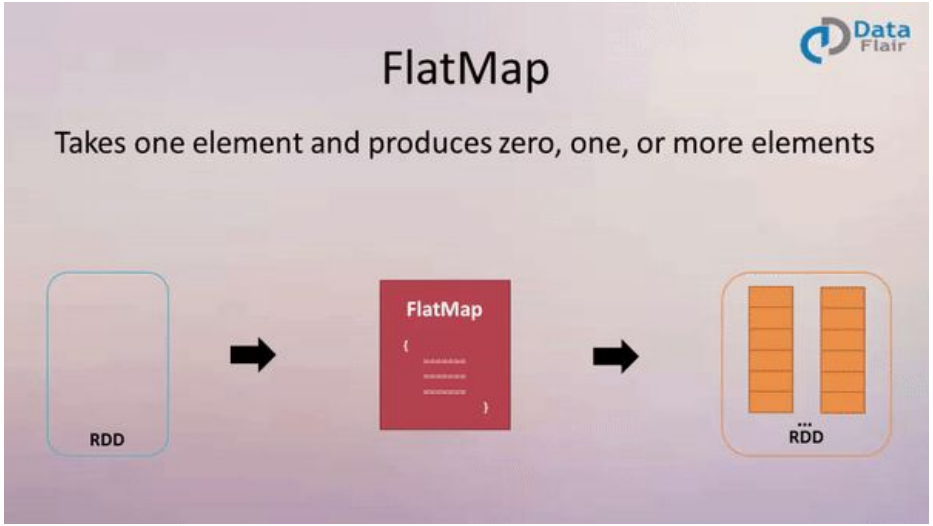
MONASH University

- RDD Operations
  - **Common Actions**
    - **Basic actions on an RDD containing {1, 2, 3, 3}**

Action	Purpose	Example	Result
reduce(*fucn*)	Combine the elements of the RDD together in parallel.	rdd.reduce(lambda x, y: x + y)	9
collect()	Return all elements from RDD.	rdd.collect()	[1, 2, 3, 3]
count()	Number of elements in the RDD.	rdd.count()	4
take(*num*)	Return number of elements from the RDD.	rdd.take(2)	[1, 2]
countByValue()	Number of times each element occurs in the RDD.	rdd.countByValue()	[(1, 1), (2, 1), (3, 2)]

See more here: https://spark.apache.org/docs/latest/rdd-programming-guide.html

Map vs FlatMap



Source : https://data-flair.training/blogs/apache-spark-map-vs-flatmap/

Thank You!

MONASH University

See you next week.