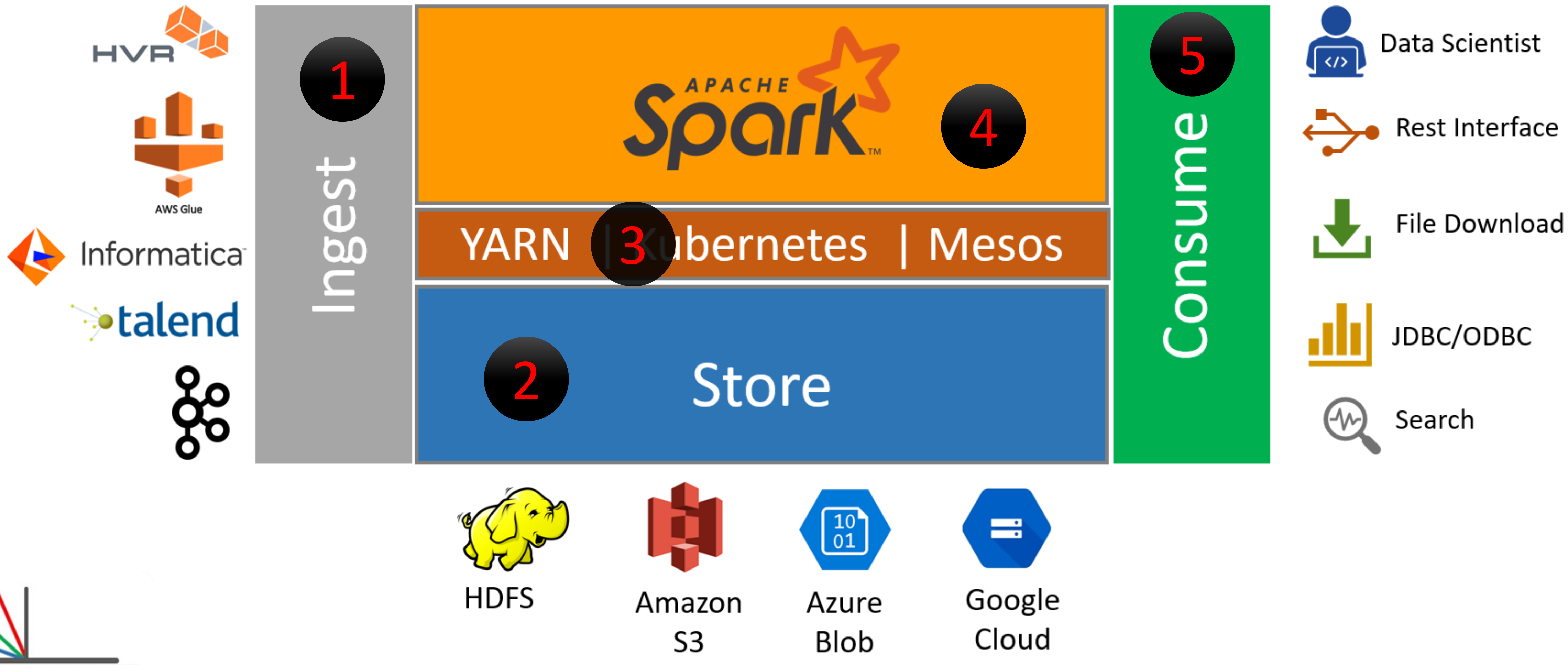
Apache Spark

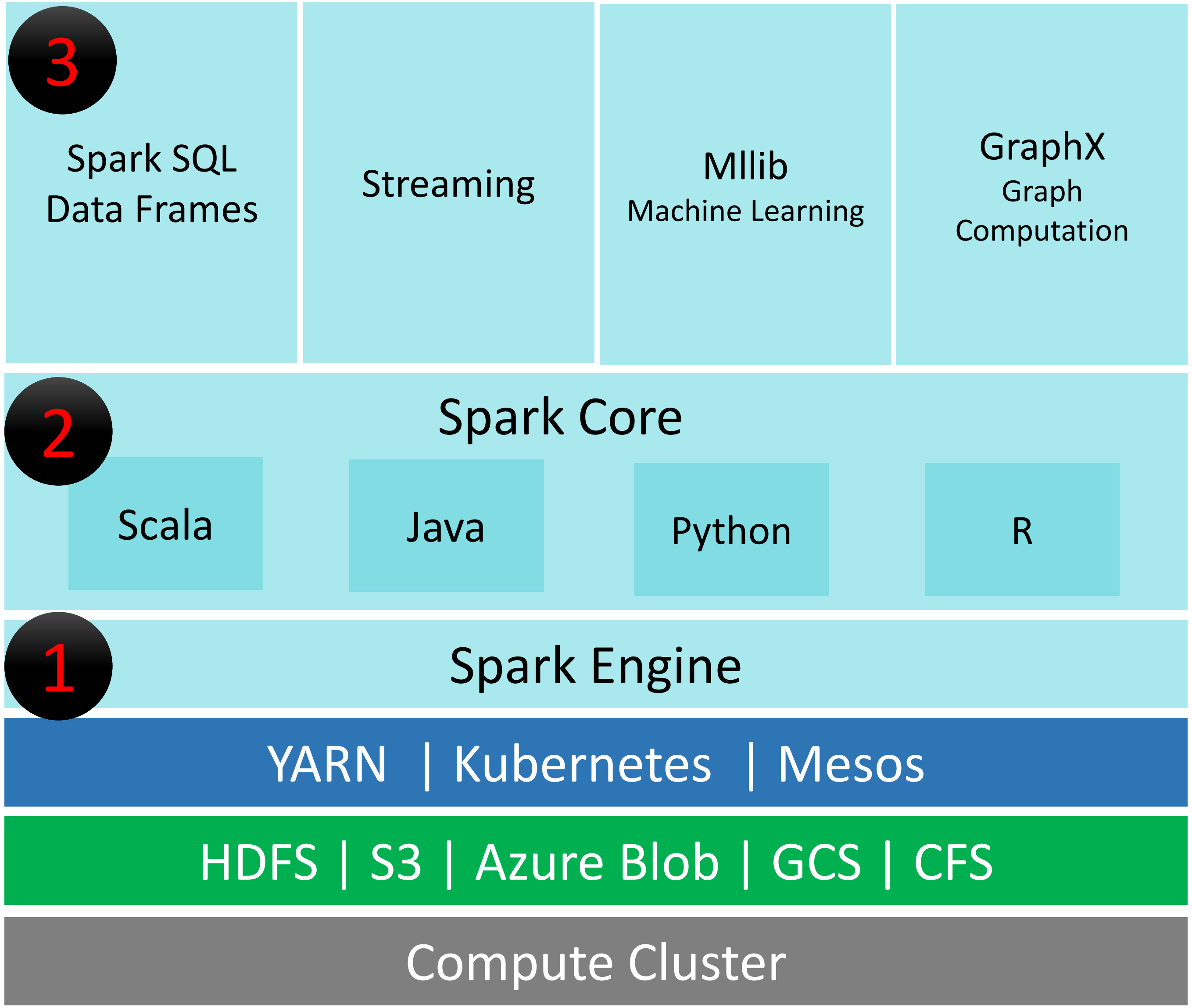
Rounding Up

Conception of Data Lake



LEARNING JOURNAL
www.learningjournal.guru

Spark Ecosystem



3

Spark SQL
Data Frames

Streaming

Mllib
Machine Learning

GraphX
Graph
Computation

2

Spark Core

Scala

Java

Python

R

1

Spark Engine

YARN | Kubernetes | Mesos

HDFS | S3 | Azure Blob | GCS | CFS

Compute Cluster

Spark Installations

1. Local Mode – Command line REPL
2. Development Scala IDE – IntelliJ IDEA
3. Databricks Cloud – Notebooks
4. Cloudera Cluster – Zeppelin Notebooks
5. Other Options – Cloud offerings

Execution Methods

How to execute Spark Programs?

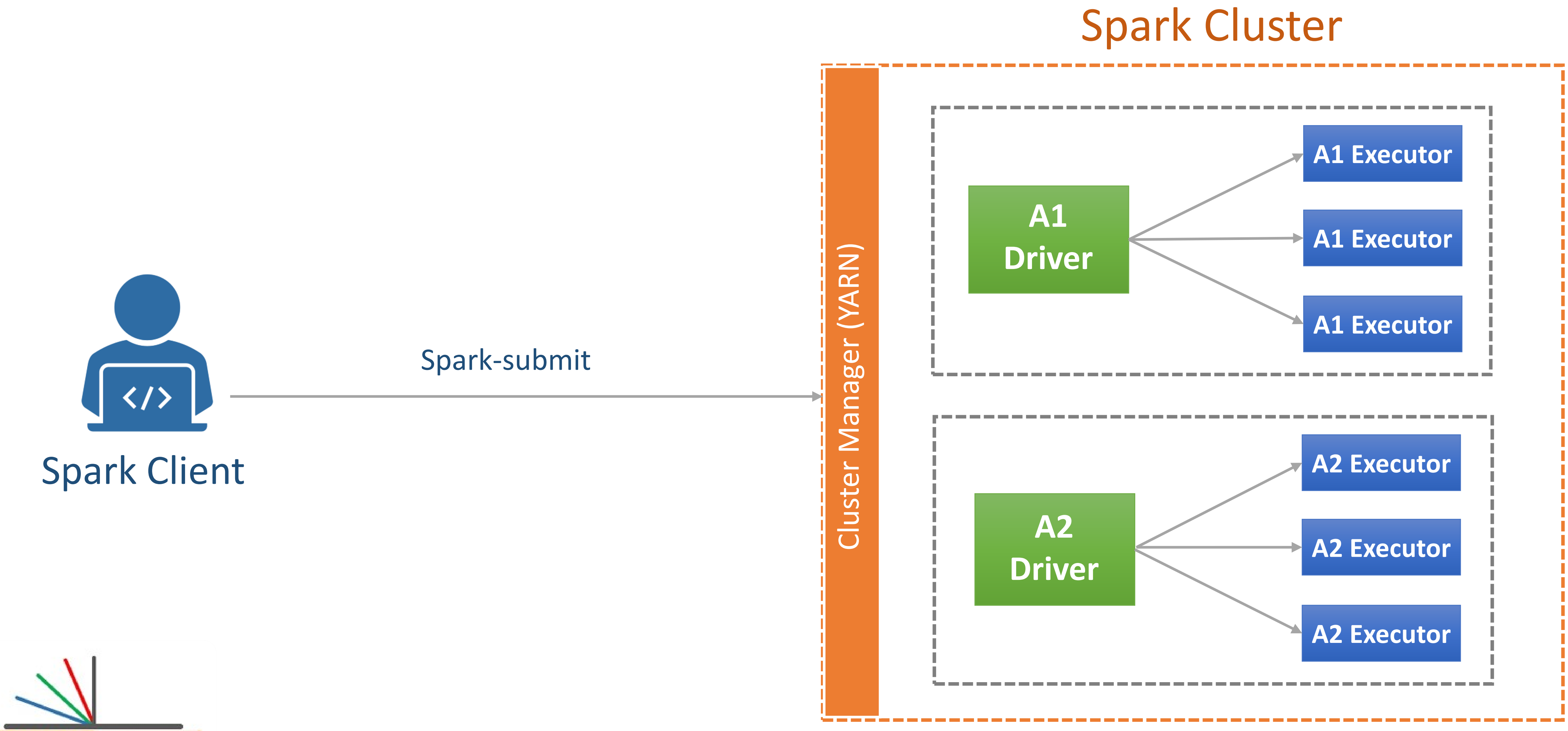
1. Interactive Clients

   spark-shell, Notebook

2. Submit Job

   spark-submit, Databricks Notebook, Rest API

Processing Model



Spark Cluster Managers & Deployment Modes

Cluster Manager

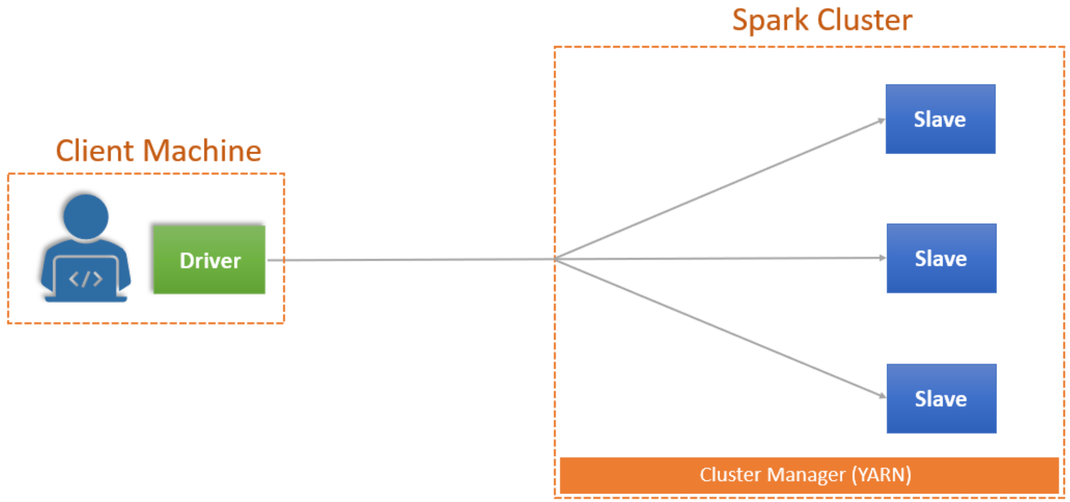
1. local[n]
2. YARN
3. Kubernetes
4. Mesos
5. Standalone

Deployment Modes

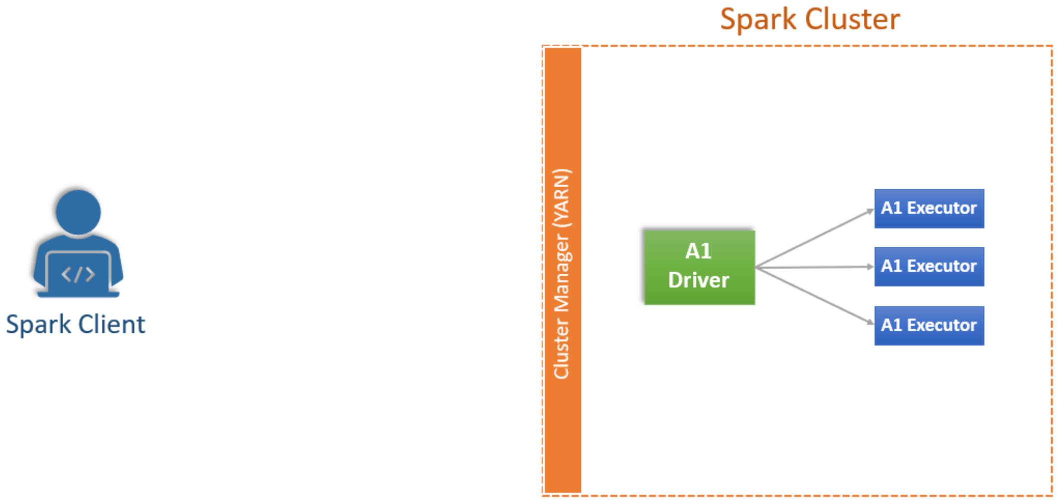
1. Client Mode
2. Cluster Mode

Deployment Modes

Client Mode



Cluster Mode



LEARNING JOURNAL
www.learningjournal.guru

Spark Execution Model

Cluster Managers

1. local[n]
2. YARN

Execution Modes

1. Client
2. Cluster

Execution Tools

1. spark-shell
2. spark-submit

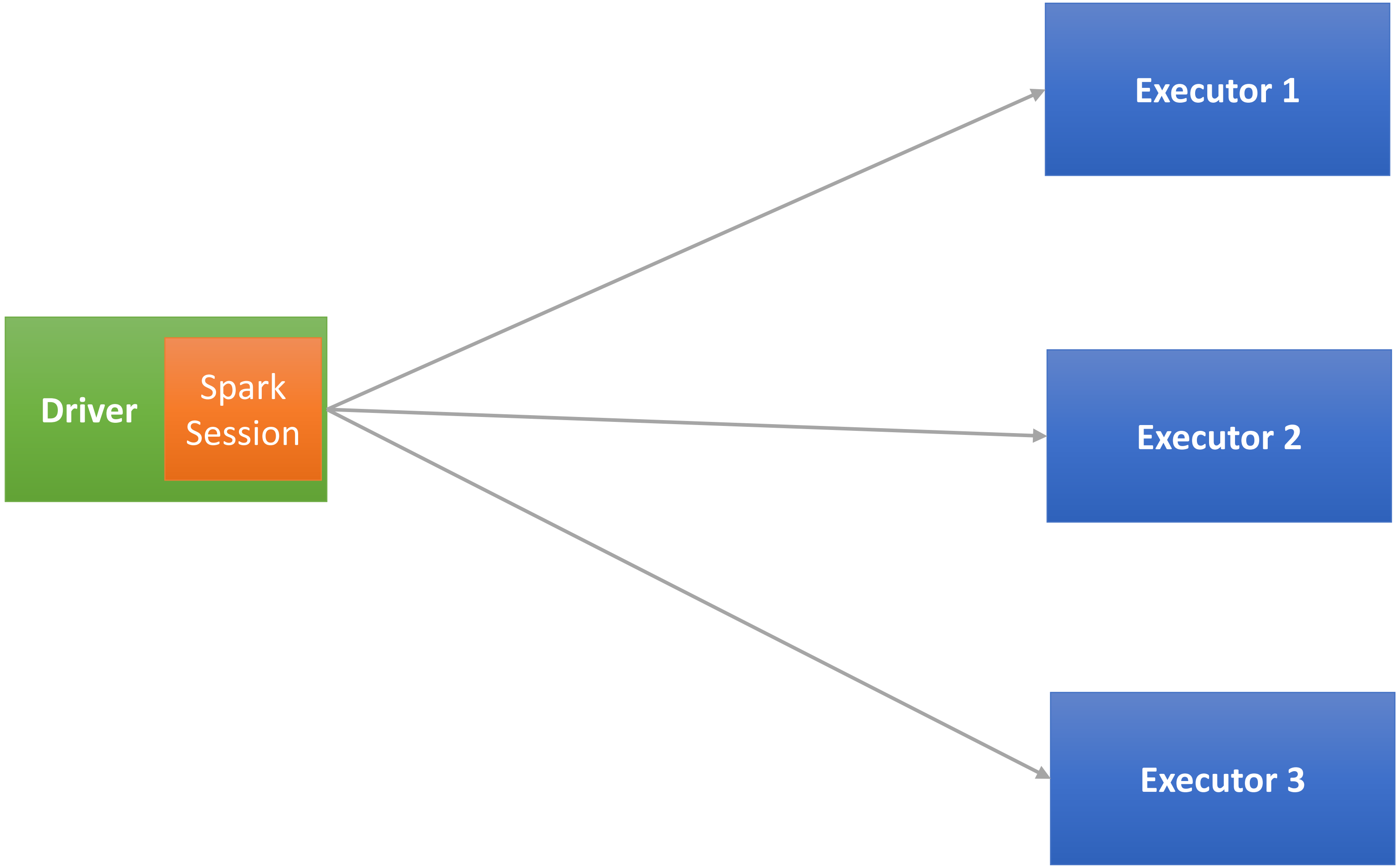
Cluster	Mode	Tool
Local	Client Mode	spark-shell
~~Local~~	~~Client Mode~~	~~spark-submit~~
~~Local~~	~~Cluster Mode~~	~~spark-shell~~
~~Local~~	~~Cluster Mode~~	~~spark-submit~~

Cluster	Mode	Tool
YARN	Client Mode	spark-shell
~~YARN~~	~~Client Mode~~	~~spark-submit~~
~~YARN~~	~~Cluster Mode~~	~~spark-shell~~
YARN	Cluster Mode	spark-submit

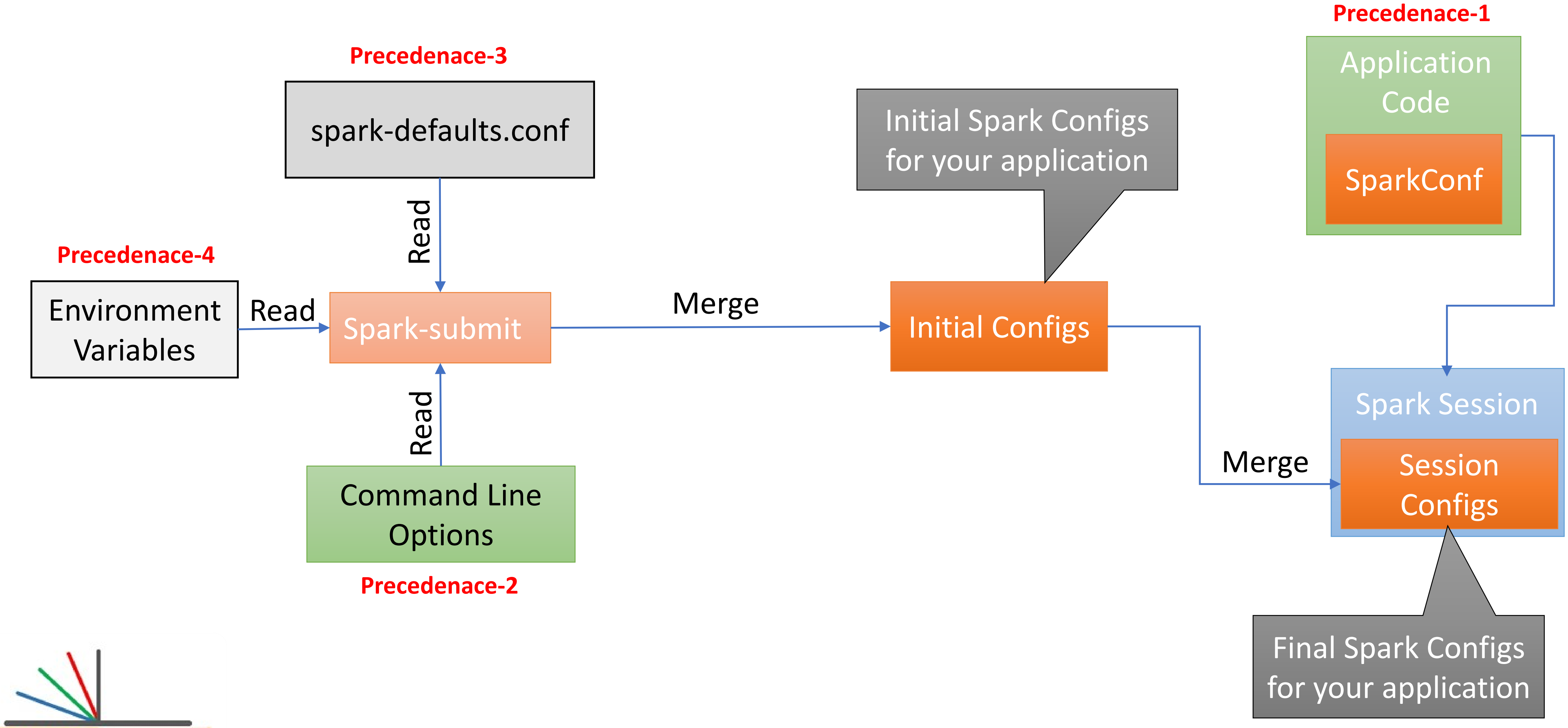
Developer Experience

1. Creating and Configuring Spark Project using your IDE
2. Configuring Log4J for your Spark Application
3. Creating and Configuring Spark Session
4. Managing your Spark Session Configurations using spark.conf
5. Creating a modular Structure for your Spark Application
6. Unit Testing Spark Application
7. Debugging Spark Drivers and executors
8. Building and packaging your Spark Application
9. Deploying your Spark Application on a Cluster
10. Collecting Application Logs from Spark Cluster

Spark Session



Spark Session Configs



Precedenace-3

spark-defaults.conf

Read

Precedenace-4

Environment Variables

Read

Spark-submit

Read

Command Line Options

Precedenace-2

Merge

Initial Spark Configs for your application

Initial Configs

Precedenace-1

Application Code

SparkConf

Merge

Spark Session

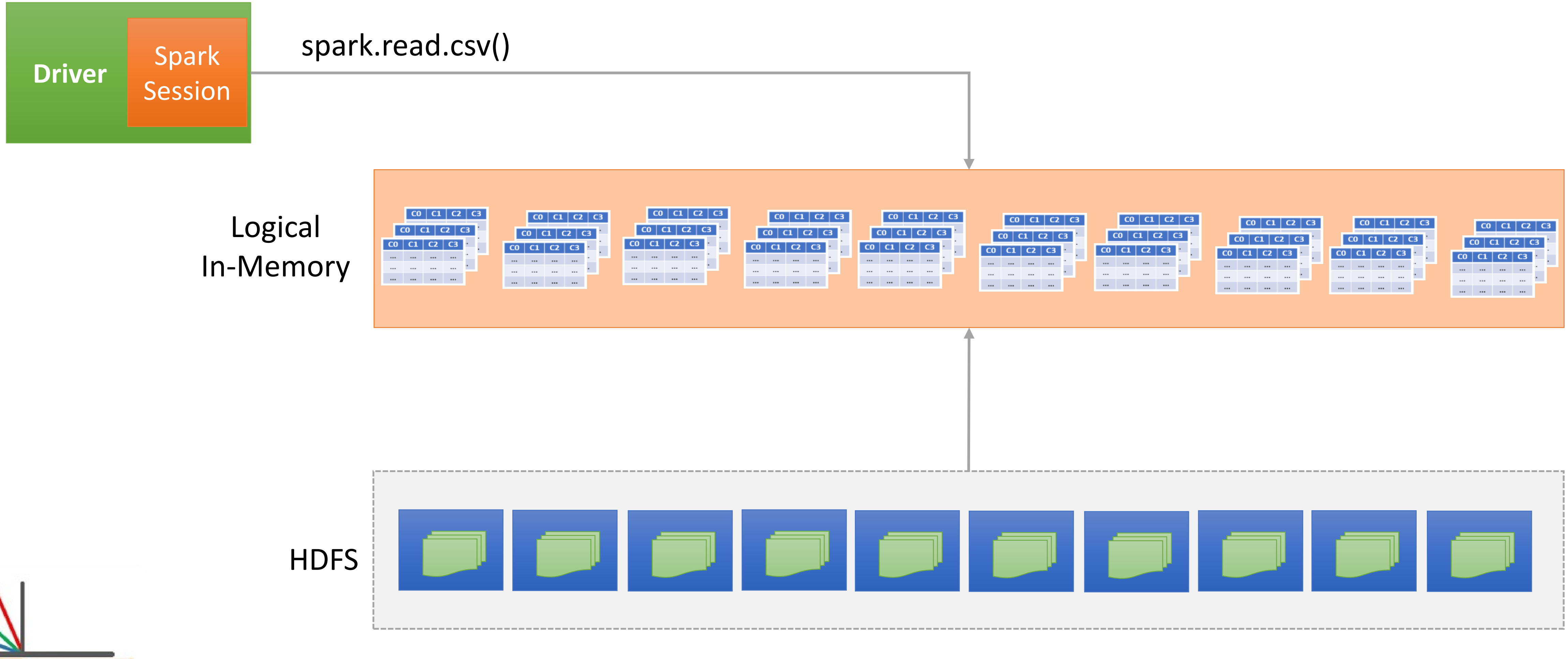
Session Configs

Final Spark Configs for your application

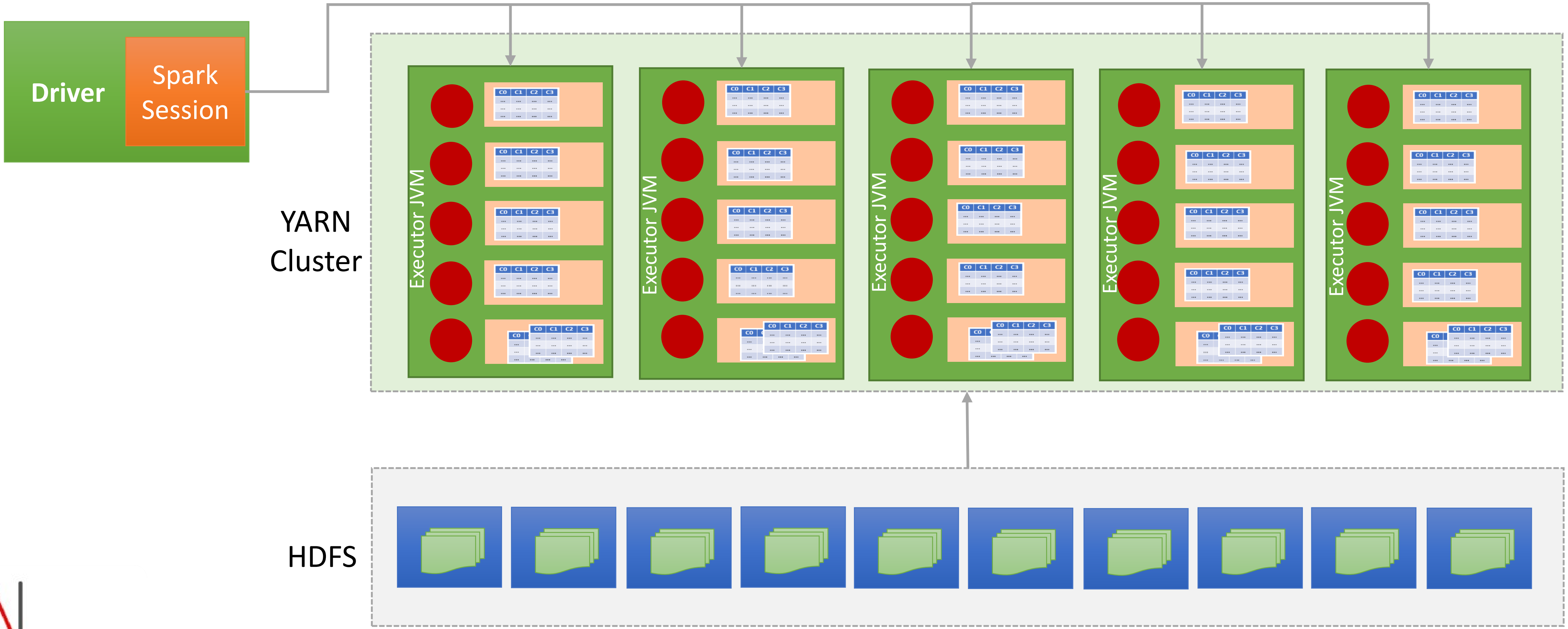
LEARNING JOURNAL

www.learningjournal.guru

Spark Data Frame Partitions

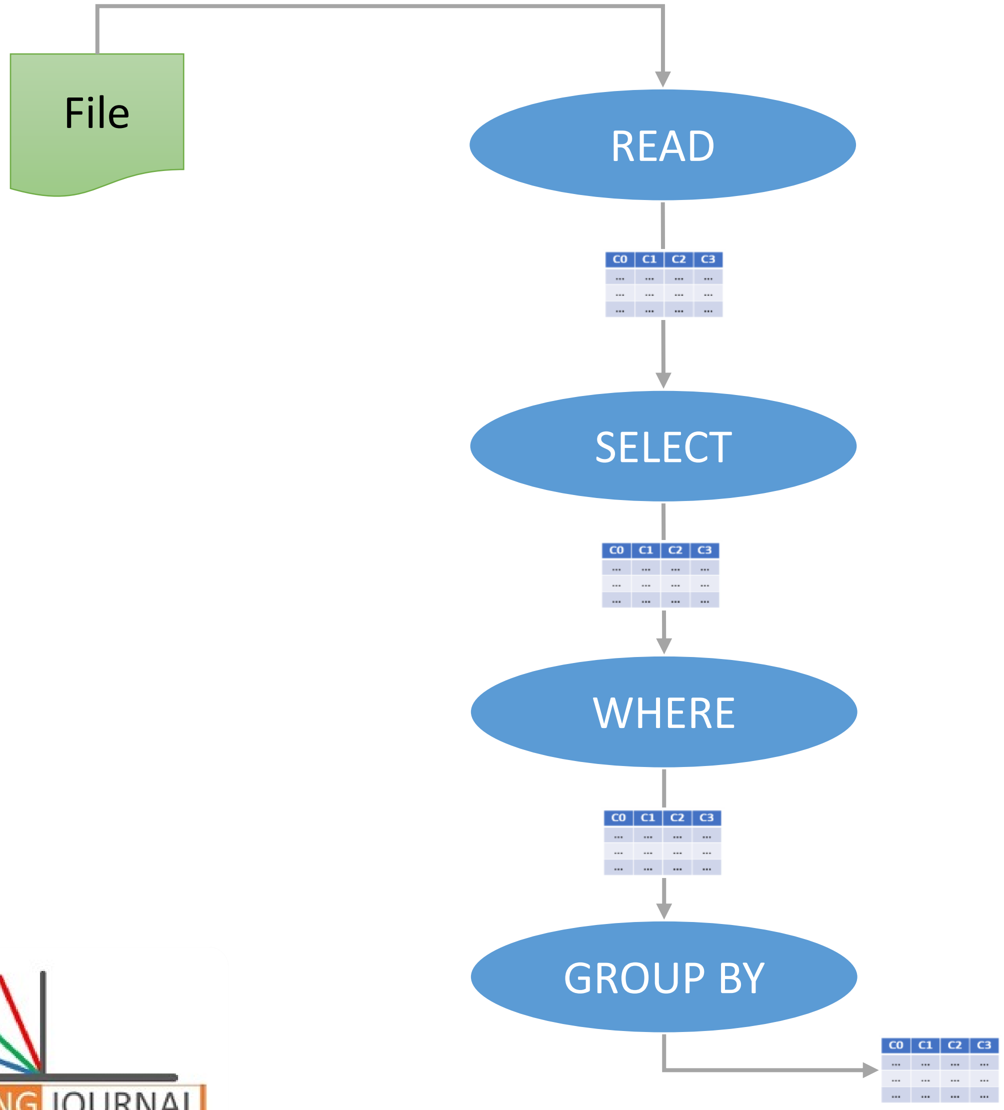


Spark Data Frame Partitions



LEARNING JOURNAL
www.learningjournal.guru

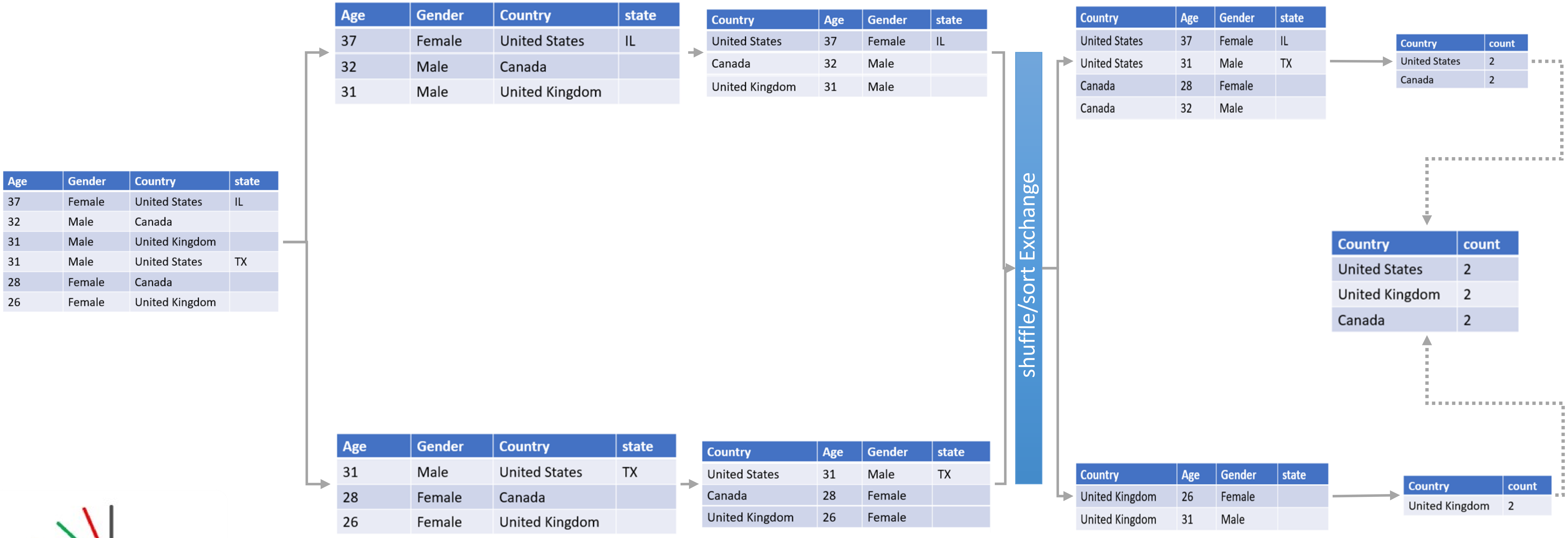
Spark Transformations DAG



1. Transformations
    1. Narrow Dependency
    2. Wide Dependency
2. Actions

Wide Dependency Transformation

A transformation that requires data from other partitions to produce valid results.



Age	Gender	Country	state
37	Female	United States	IL
32	Male	Canada	
31	Male	United Kingdom	
31	Male	United States	TX
28	Female	Canada	
26	Female	United Kingdom	

Age	Gender	Country	state
37	Female	United States	IL
32	Male	Canada	
31	Male	United Kingdom	

Age	Gender	Country	state
31	Male	United States	TX
28	Female	Canada	
26	Female	United Kingdom	

Country	Age	Gender	state
United States	37	Female	IL
Canada	32	Male	
United Kingdom	31	Male	

Country	Age	Gender	state
United States	31	Male	TX
Canada	28	Female	
United Kingdom	26	Female	

shuffle/sort Exchange

Country	Age	Gender	state
United States	37	Female	IL
United States	31	Male	TX
Canada	28	Female	
Canada	32	Male	

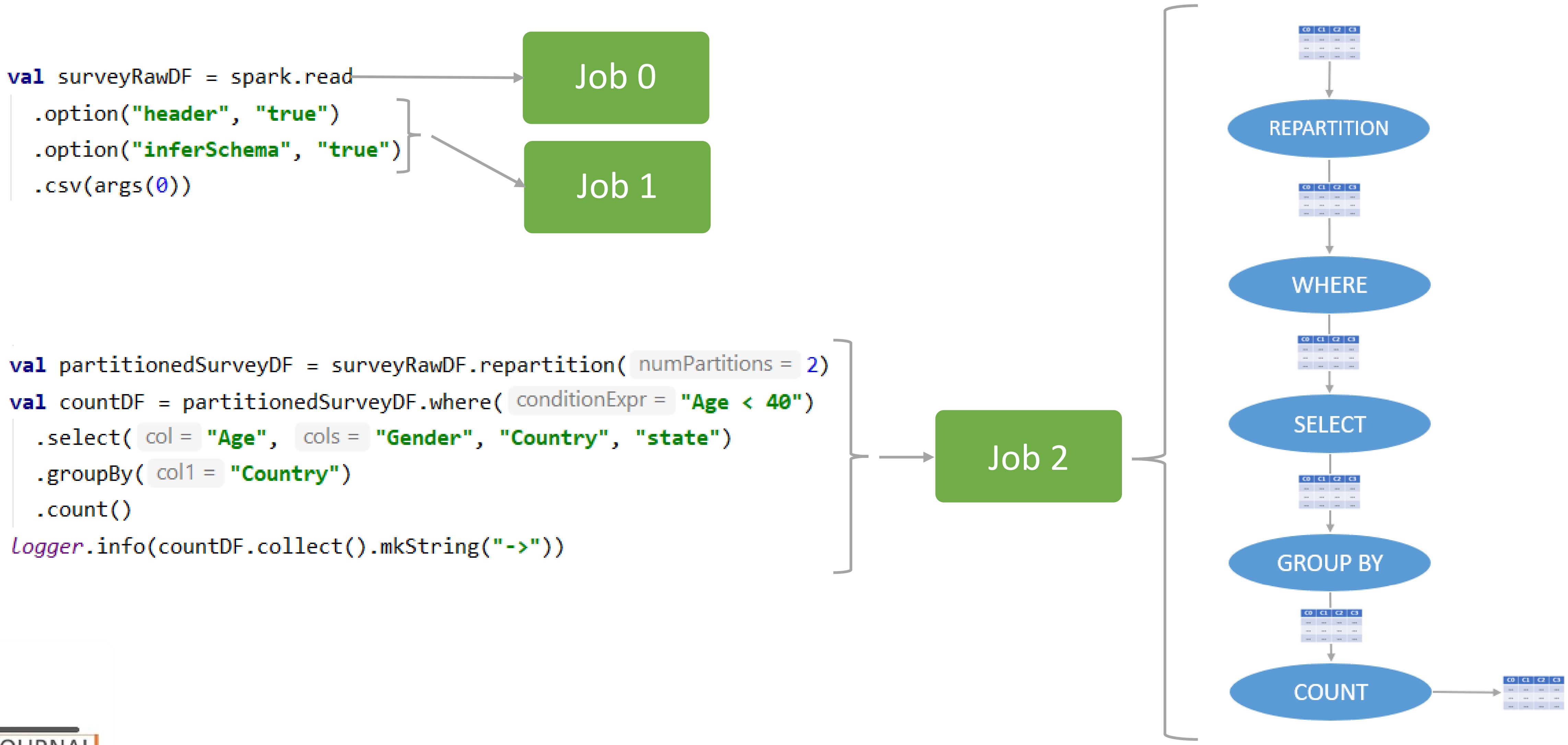
Country	Age	Gender	state
United Kingdom	26	Female	
United Kingdom	31	Male	

Country	count
United States	2
Canada	2

Country	count
United Kingdom	2

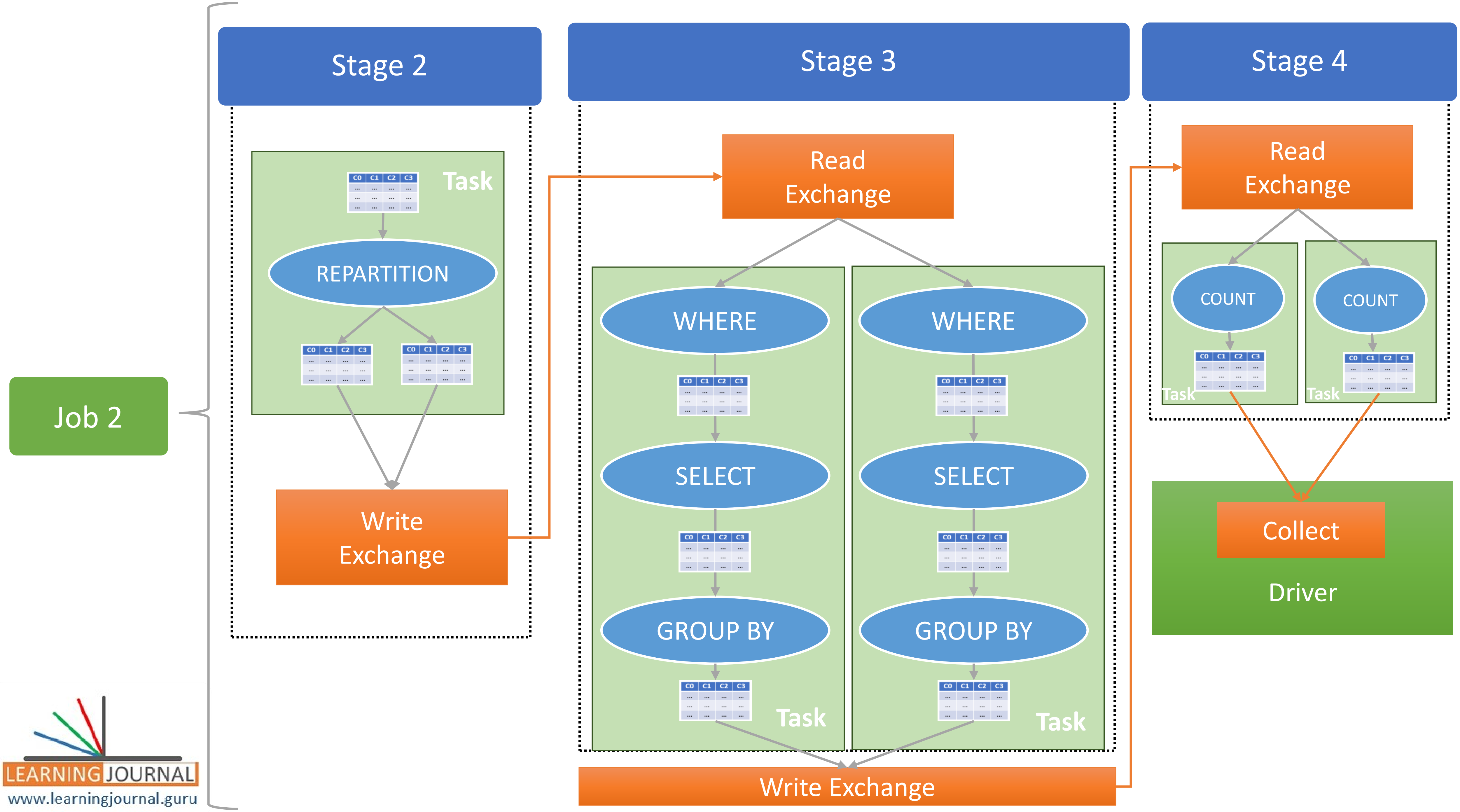
Country	count
United States	2
United Kingdom	2
Canada	2

Spark Execution Plan



```
val surveyRawDF = spark.read
  .option("header", "true")
  .option("inferSchema", "true")
  .csv(args(0))
```

```
val partitionedSurveyDF = surveyRawDF.repartition( numPartitions = 2)
val countDF = partitionedSurveyDF.where( conditionExpr = "Age < 40")
  .select( col = "Age",  cols = "Gender", "Country", "state")
  .groupBy( col1 = "Country")
  .count()
logger.info(countDF.collect().mkString("->"))
```


Job 2
Stage 2
Task
C0 C1 C2 C3
REPARTITION
Write Exchange
Stage 3
Read Exchange
WHERE
SELECT
GROUP BY
Task
WHERE
SELECT
GROUP BY
Task
Write Exchange
Stage 4
Read Exchange
COUNT
Task
COUNT
Task
Collect
Driver

LEARNING JOURNAL

www.learningjournal.guru