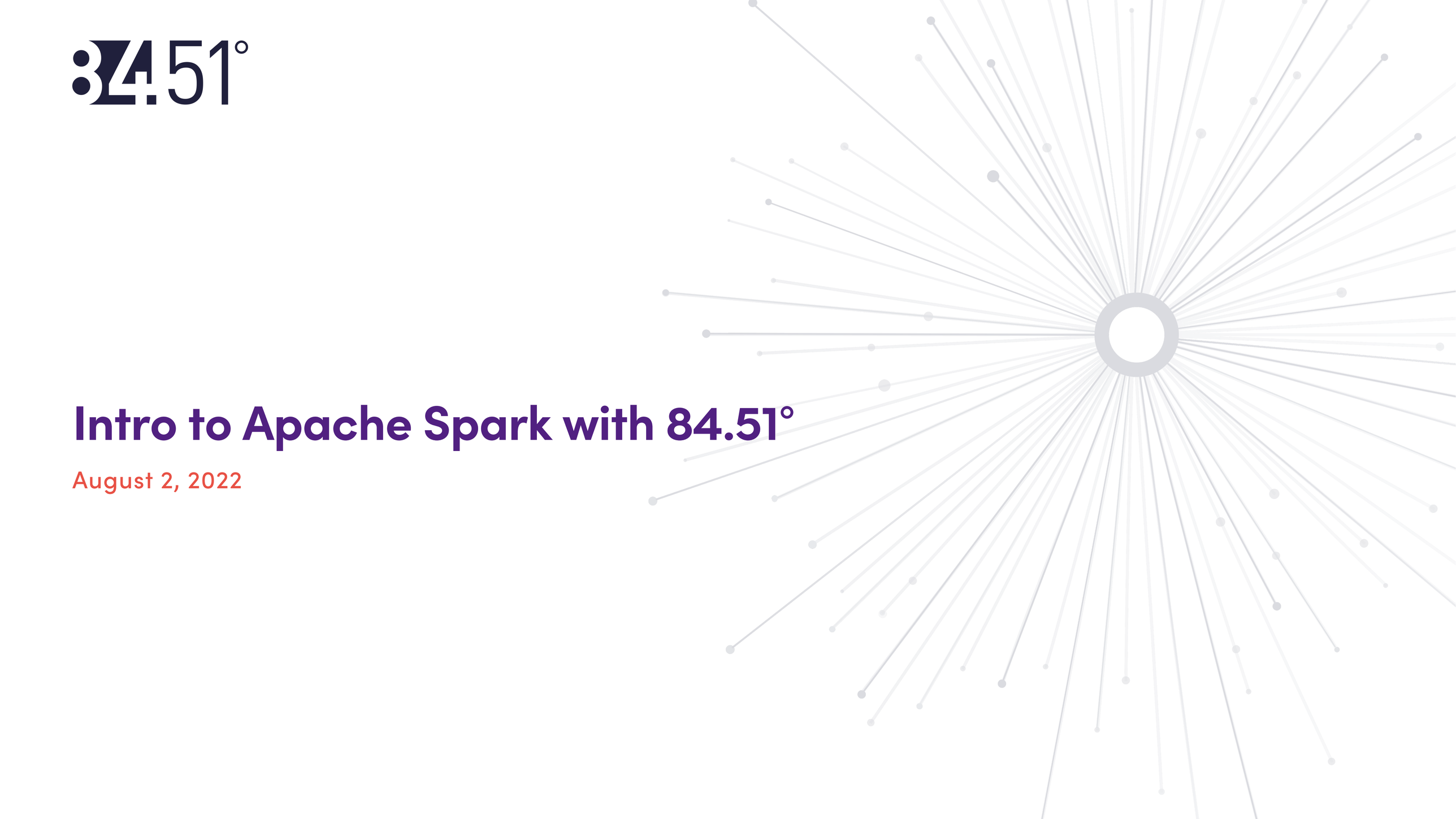


Intro to Apache Spark with 84.51°

August 2, 2022

Workshop Agenda

- Intro to Spark and the Databricks UI (30 mins)
- 10 min break
- Guided Exercises Part I (25 mins)
- 5 min break
- Guided Exercises Part II (25 mins)
- 10 min break
- Individual Exercises (1 hour)

Please put comments/questions in the chat! ☺

About the Presenters

Data Scientists and Engineers at 84.51°

Caitlin Casar
Data Engineer

Theo Randolph
Data Scientist

Tara Jawahar
Data Engineer

Olivia Hatch
Data Engineer

Luke Lavin
Data Engineer

Anthony Igel
Data Engineer

Who We Are

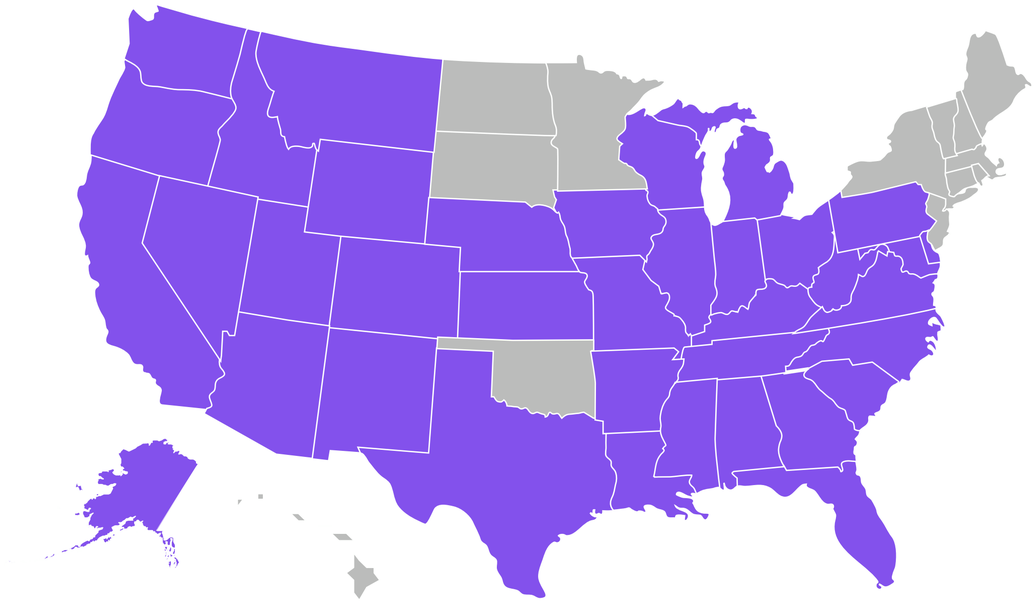
84.51° brings together customer data, predictive analytics and marketing strategy to drive **sales growth** and **customer loyalty** for Kroger and more than 300 consumer-packaged-goods companies in the U.S.

The Kroger Company

2,800 Stores in 35 States

60MM+

Households



Kroger is a $115 Billion company with 2800 stores in 35 states, operating under 2 dozen banners serving 60MM households across America.

It starts with the customer

Shopper data helps Kroger and 84.51° put insights into action

60MM+ Households
visit the store

Kroger Plus card
used at checkout

Capture and analyze
data, derive insights

INFO SESSION + COFFEE CHATS

Week of October 3rd

Look out for Details + Links in Handshake

Applications will be open for internships + full time opportunities for data science and engineering!

Apply by October 9th

Join our MeetUps:

https://www.meetup.com/84-51-technology-meet-up/


APACHE
Spark
TM

WHAT EVEN IS SPARK?

WHAT IS SPARK?

Distributed platform that can handle datasets/jobs too big for Python or R

- Offers an API (functions, basically) that lets you run use Spark entirely from within Python/R

Cluster ~ many connected computers (nodes) that work together to do a single job (you might say, “a **cluster** of **nodes**”)

n.b. worker = node = server = machine

- We will likely use these synonymously for the rest of the day

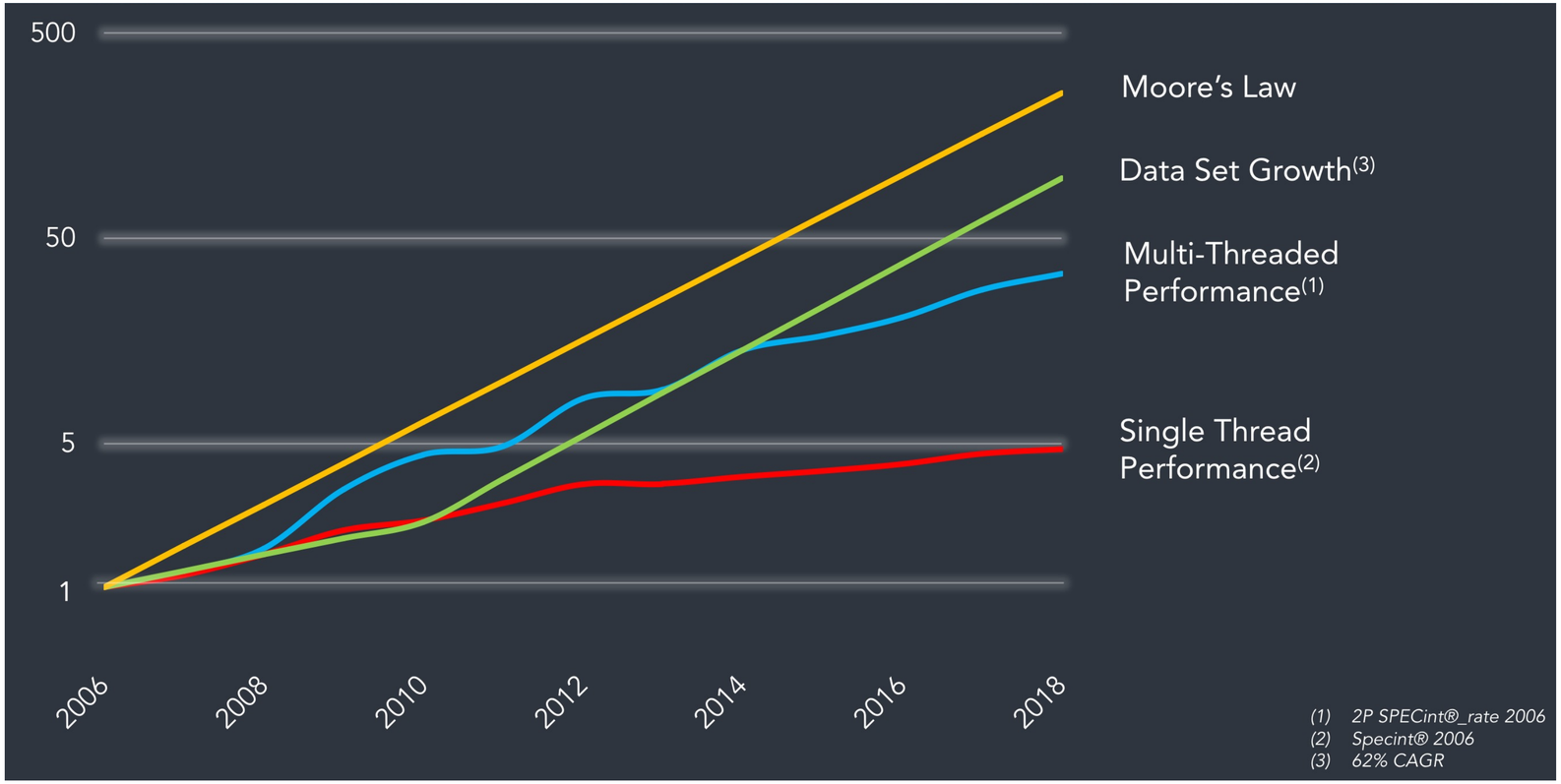
WHY A DISTRIBUTED SYSTEM?

Frankly, distributed computing is harder for programmers

- Don’t be surprised to find that Spark is full of new errors!

So there must be another compelling reason to use it, right?

WHY A DISTRIBUTED SYSTEM?



Datasets are growing more quickly than our CPUs are getting faster

- CPU speed hitting limits in sizing, heat dissipation

Multicore architectures can come closer to keeping up

Certain common data tasks (filters, selects) are trivially parallelizable

Memory (aka RAM) sizes have also been slowing down in growth

- So using memory across multiple machines allows us to fit our data better

SPARK
INSIDE THE BOX

WHAT IS SPARK?

A black box?

When learning, you can think about Spark as a black box.

Spark can operate as a system you ask for answers to your queries.

WHAT IS SPARK?

~~A black box?~~

BUT it's important to know a bit about how Spark works in order to use it well

This is more true with Spark than it was with Oracle (or traditional RDBMS)

- Oracle would intelligently guess how much memory and CPU you needed – Spark needs to ask you for resources

So we're going to open the black box and look inside

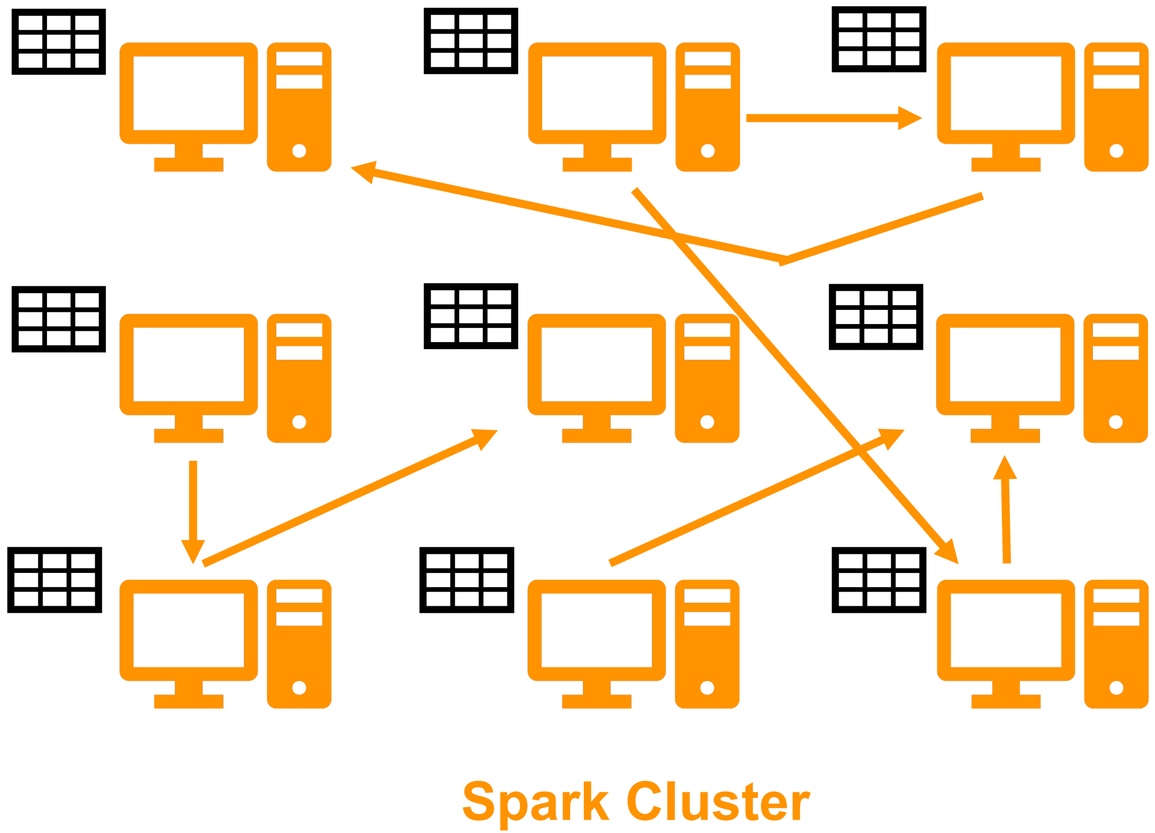
SPARK INSIDE THE BOX

Loading Data

- Our data lives in persistent storage – in this example it's Google Cloud Platform (GCP), but AWS and Azure are also common
- When the spark cluster requests data, it is split up into chunks row-wise and distributed across the cluster

SPARK INSIDE THE BOX

Computation

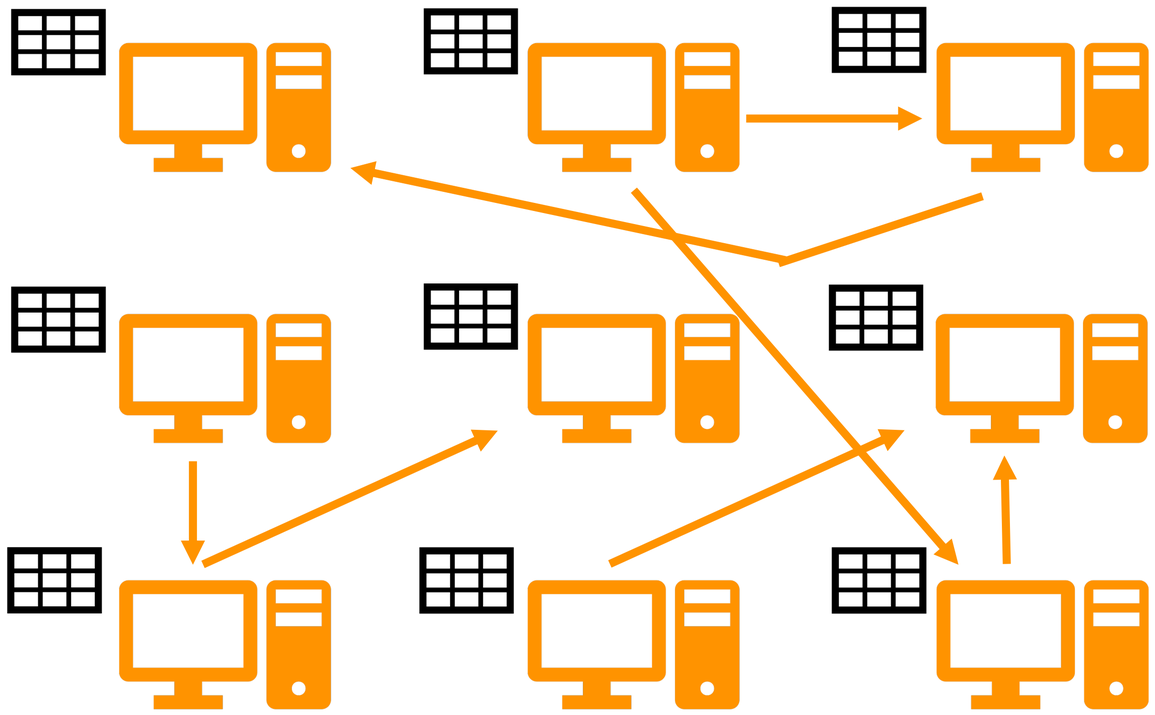


Spark Cluster

- During computation, each node will execute as much as possible independently (“narrow” operations)
  - Map, filter
- However, some operations require coordination across nodes (“wide” operations)
  - Joins, group bys, aggregate, repartition
- Communicating and moving data between nodes is the bottle-neck in Spark

SPARK INSIDE THE BOX

Computation

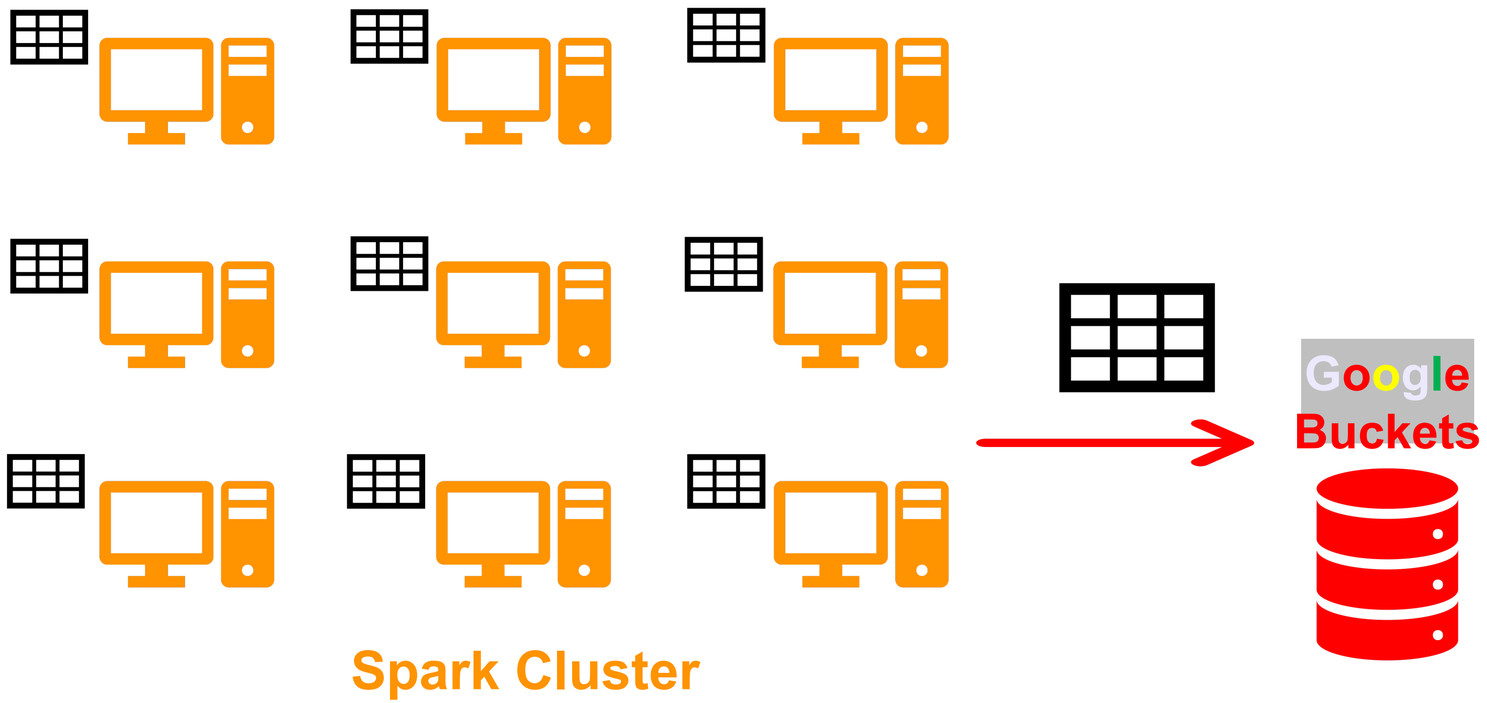


Spark Cluster

- Independent work followed by “shuffling” data repeats as necessary

SPARK INSIDE THE BOX

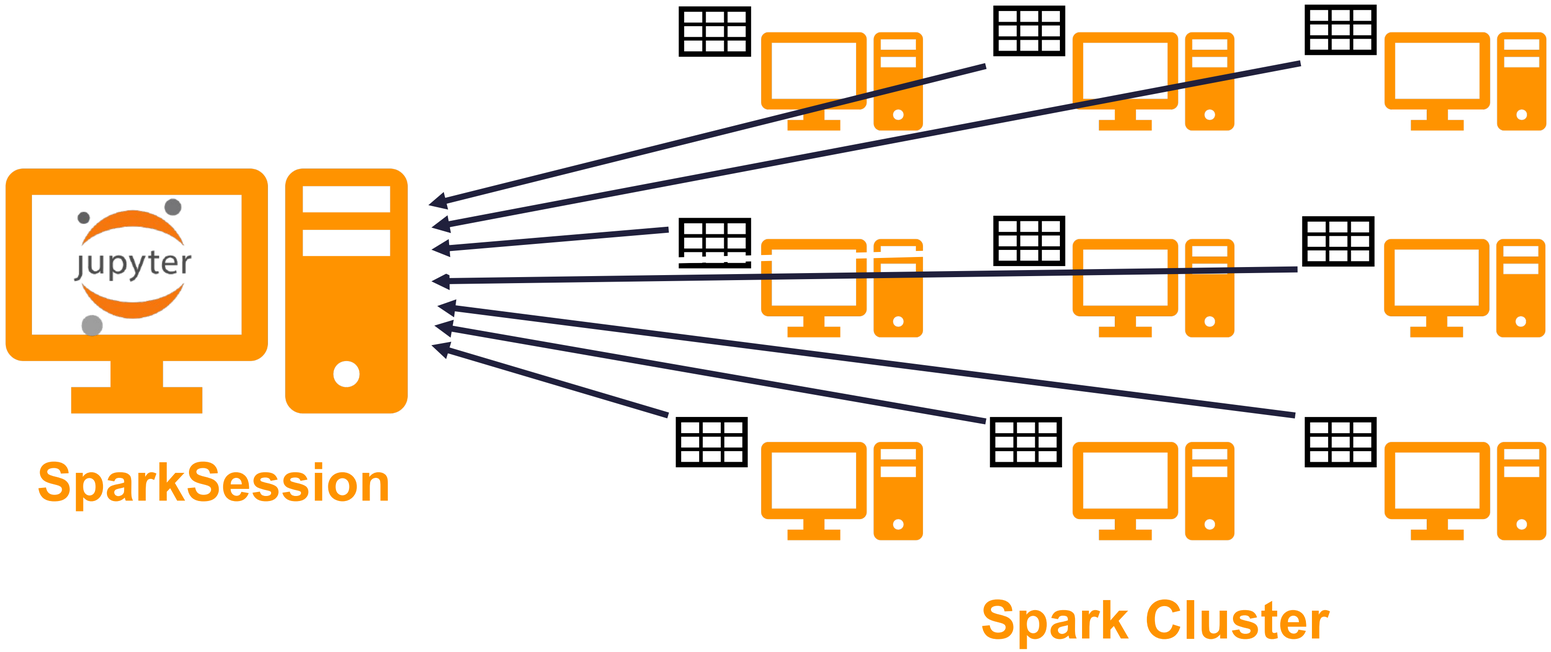
Saving Data



- Spark handles combining the distributed data when you write it out

COLLECTING DATA

COLLECTING



- In Spark, each worker owns a piece of the data
- “Collecting” is bringing back data to the server where Spark is running (e.g. Databricks, Jupyter, Rstudio, etc)
- This means you’re bringing lots of data back to **one computer**! Use it with caution!

COLLECTING

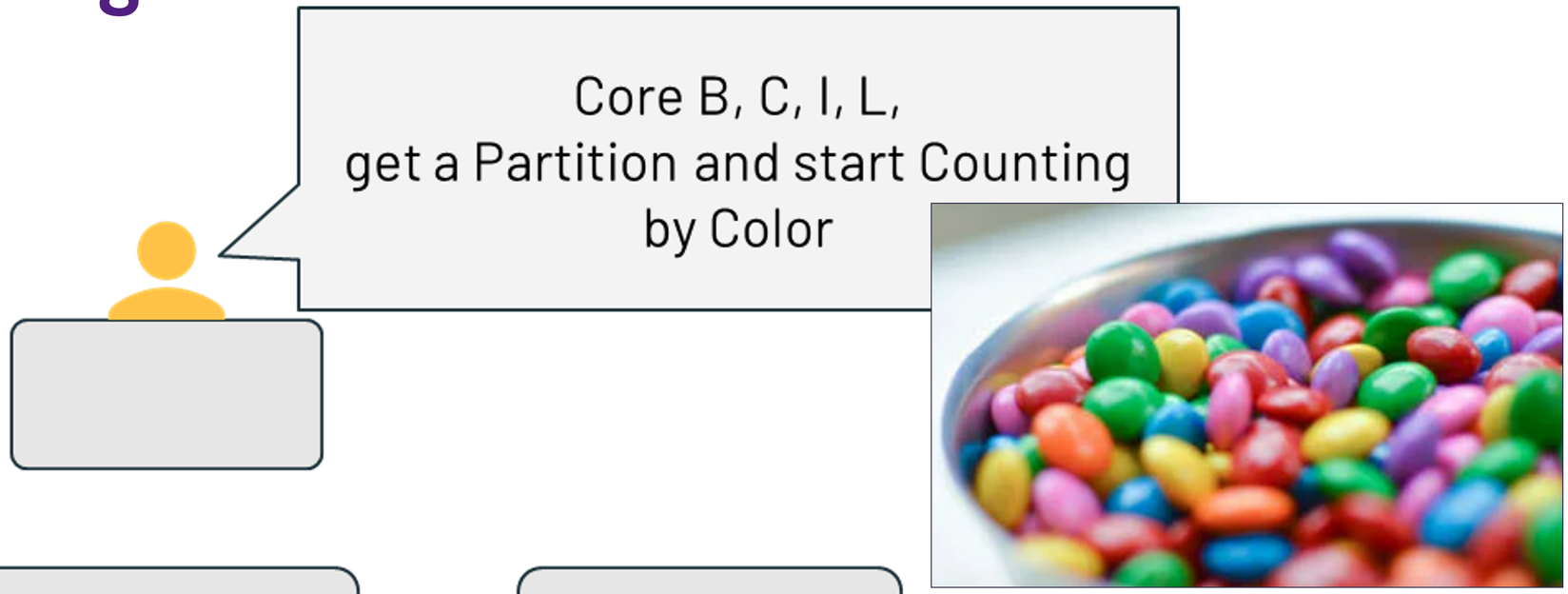
Some tasks require collection: Plotting, exporting locally, native Python and R functions

Examples:

- Using ggplot
- Using matplotlib
- Using functionality Spark doesn’t support (model interpretation tools)

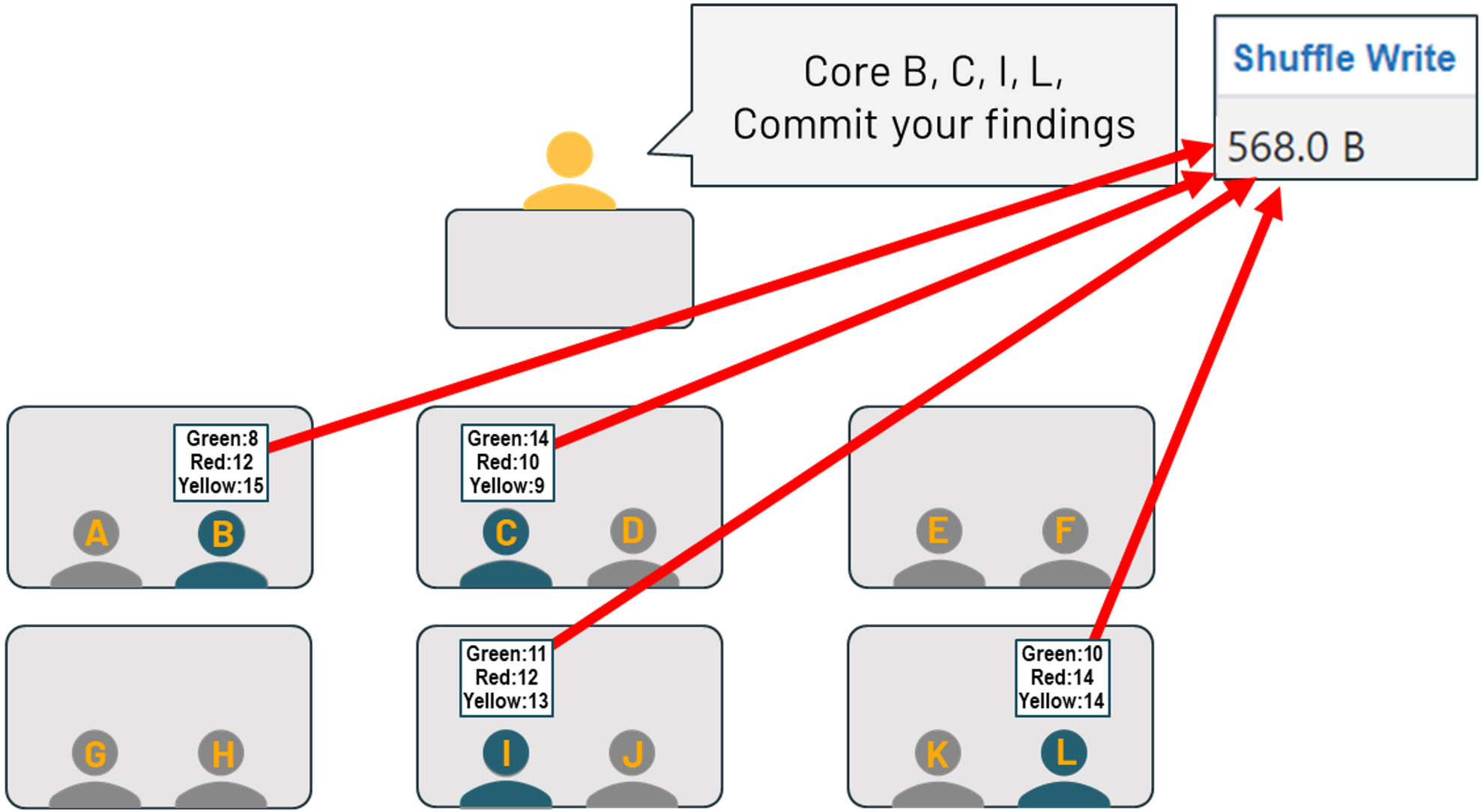
EXAMPLE: COUNTING CANDY BY COLOR

Spark Execution: Stage 1. Local count



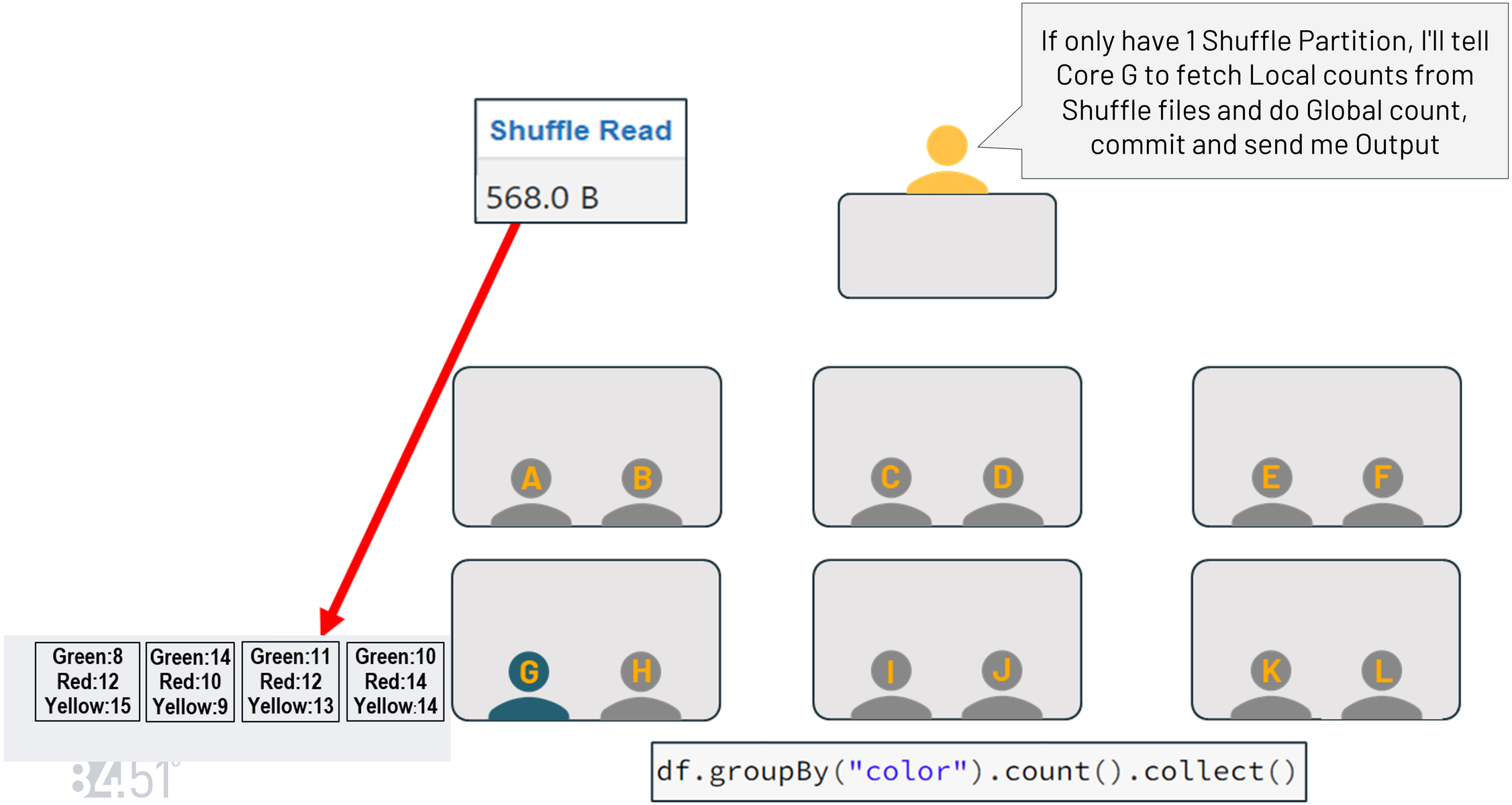

```
df.groupBy("color").count().collect()
```

Spark Execution: Stage 1. Local count complete

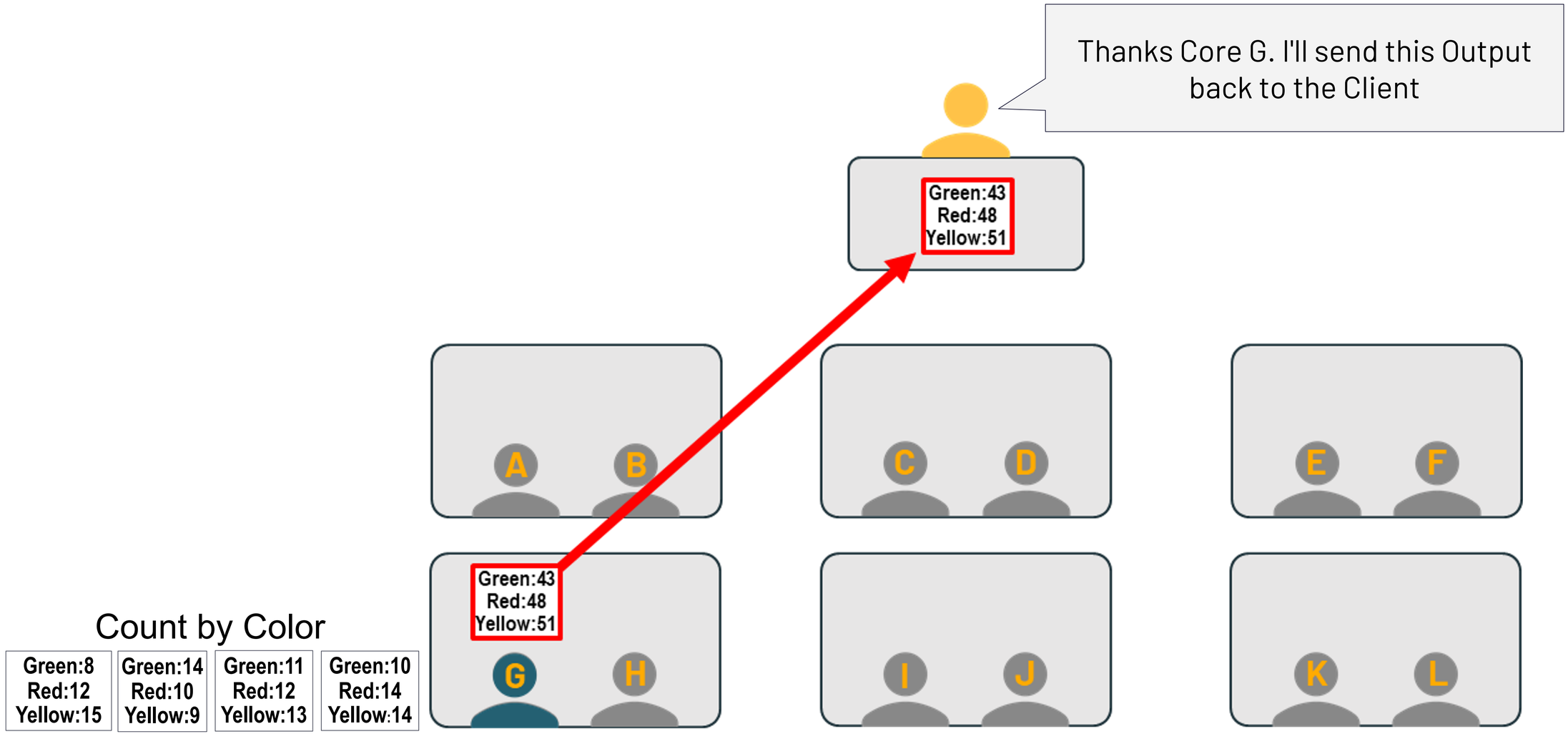



```
df.groupBy("color").count().collect()
```

Spark Execution: Stage 2. Global count



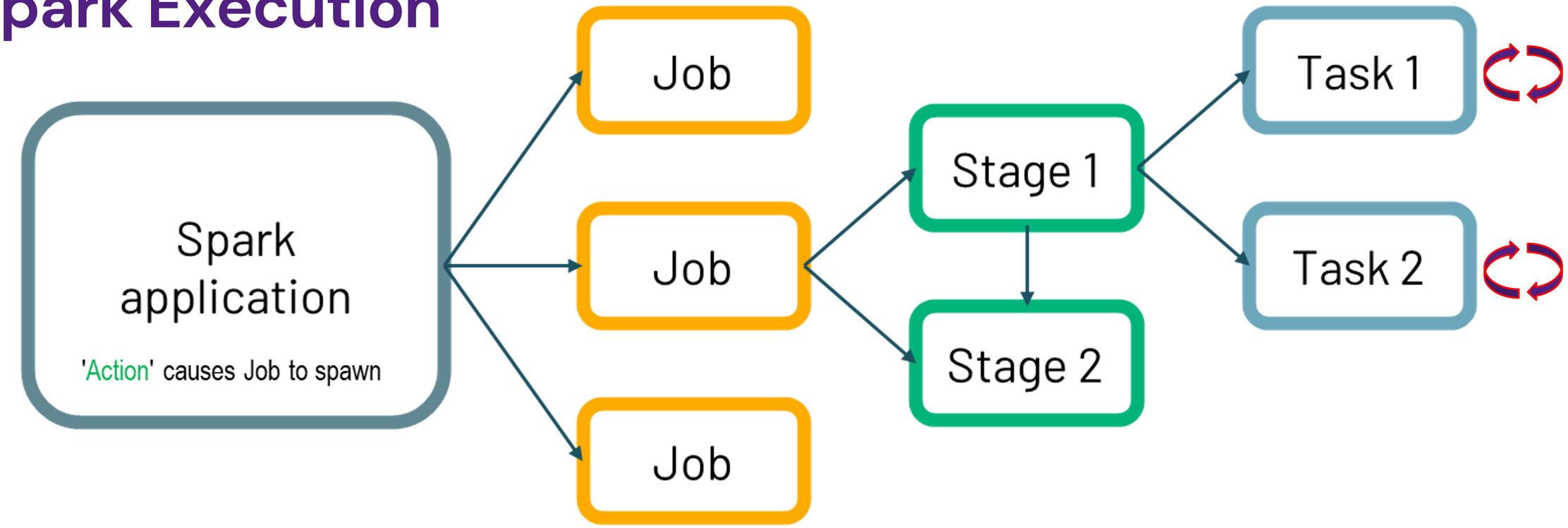
Spark Execution: Stage 2. Global count, send to Driver



QUESTIONS?

APPENDIX

Spark Execution




```
trafficDF1 = (eventsDF.groupBy(col("traffic_source"), window(col("createdAt"), "1 hour"))
                      .agg(approx_count_distinct("user_id").alias("active_users"))
                      .select("traffic_source", "active_users", hour(col("window.start")).alias("hour"))
                      .orderBy("hour", "active_users"))
display(trafficDF1)
```

An Action spawns Job(s)

Spark UI

Job Id (Job Group)	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
194 (3814184425400510911_5162481	3/3	10/10

Spark Execution

Task = Running a Memory Partition on Core within Executor

Driver creates bite-size Memory Partitions that Executor's Cores will then run in Parallel as Tasks (units of work)

rows

partitionId	count
0	487836
1	487808
2	509062
3	509024
4	509009