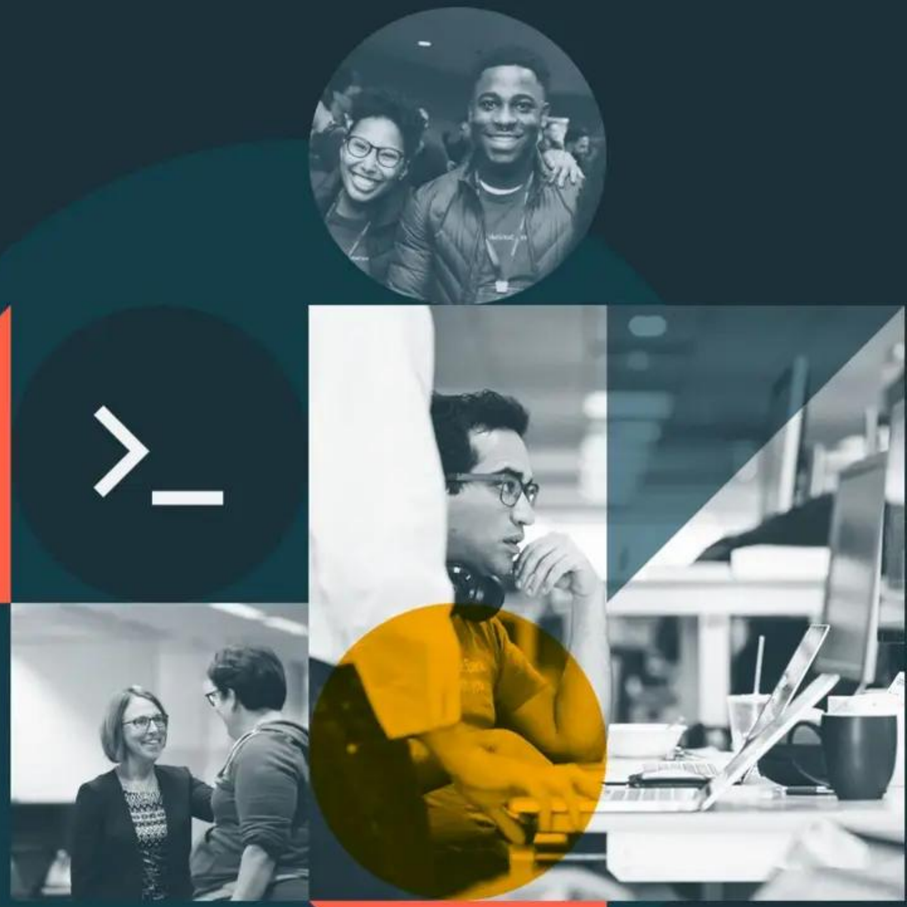
Apache Spark Programming

With Databricks



©2022 Databricks Inc. — All rights reserved

Welcome

©2022 Databricks Inc. — All rights reserved

Course Agenda

Here's where we're headed

DataFrames	Introduction	Databricks Platform	Spark SQL	Reader & Writer	DataFrame & Column
Transformations	Aggregation	Datetimes	Complex Types	Additional Functions	User-Defined Functions
Performance	Spark Architecture	Query Optimization	Partitioning		
Structured Streaming and Delta	Streaming Query	Aggregating Streams	Delta Lake		

©2022 Databricks Inc. — All rights reserved

Lesson Objectives

By the end of this course, you should be able to:

1. Identify core features of Spark and Databricks
2. Describe how DataFrames are created and evaluated in Spark
3. Apply the DataFrame API to process and analyze data
4. Demonstrate how Spark is optimized and executed on a cluster
5. Apply Delta Lake and Structured Streaming to process data

©2022 Databricks Inc. — All rights reserved

Module 1

Introductions

©2022 Databricks Inc. — All rights reserved

Welcome!

Let's get to know you

- Name
- Role and team
- Programing experience
- Motivation for attending
- Personal interest

©2022 Databricks Inc. — All rights reserved

Module 2

Spark Core

©2022 Databricks Inc. — All rights reserved

Spark Core

Databricks Ecosystem
Spark Overview

Spark SQL

Reader & Writer

DataFrame & Column

©2022 Databricks Inc. — All rights reserved

Databricks Ecosystem

©2022 Databricks Inc. — All rights reserved

Lakehouse

One simple platform to unify all of your data, analytics, and AI workloads

Customers

5000+

across the globe

Original creators of:

Apache Spark

Delta Lake

mlflow

©2022 Databricks Inc. — All rights reserved

Data Lake

Lakehouse

One platform to unify all of your data, analytics, and AI workloads

Data Warehouse

©2022 Databricks Inc. — All rights reserved

Data Lake

DELTA LAKE

An open approach to bringing **data management and governance** to data lakes

Better reliability with transactions

48x faster data processing with indexing

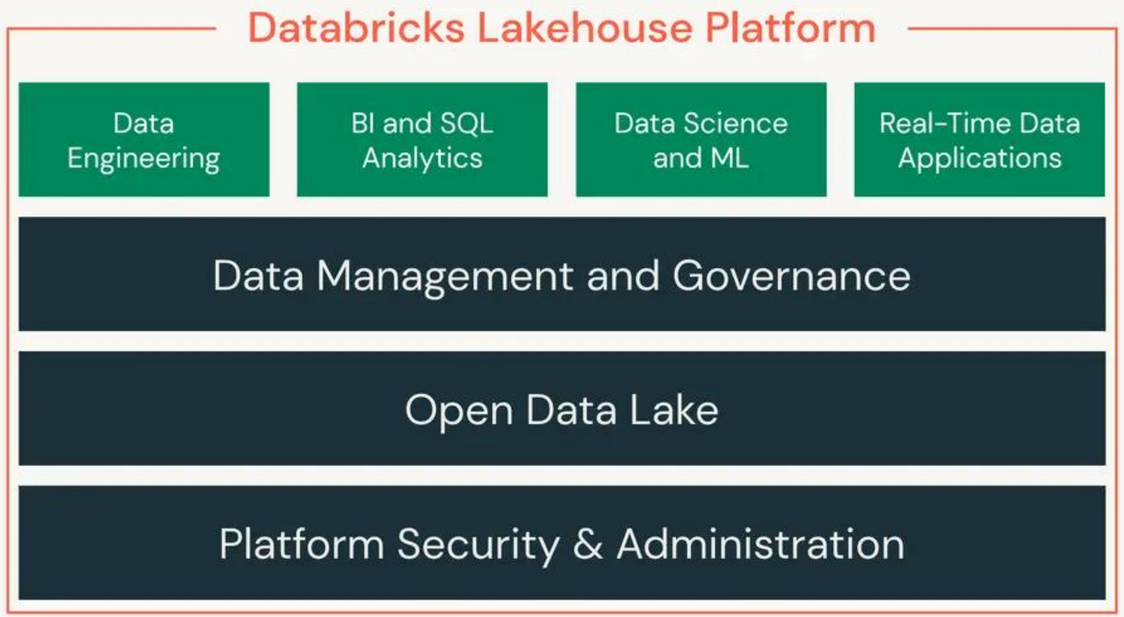
Data governance at scale with fine-grained access control lists

Data Warehouse

©2022 Databricks Inc. — All rights reserved

The Databricks Lakehouse Platform

- Simple
- Open
- Collaborative

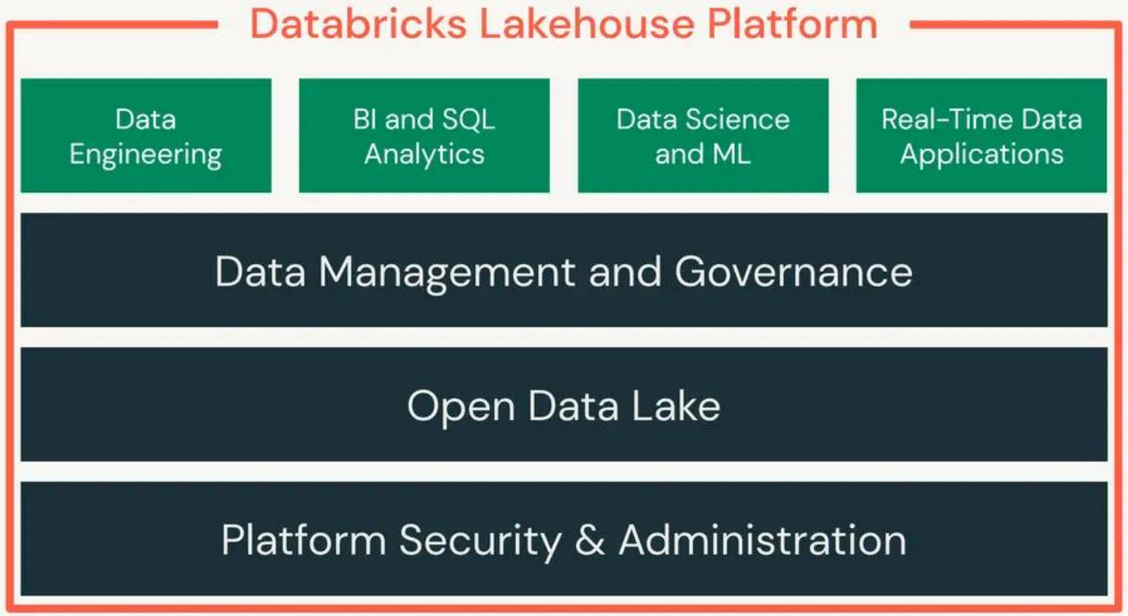


©2022 Databricks Inc. — All rights reserved

The Databricks Lakehouse Platform

Simple

Unify your data, analytics, and AI on one common platform for all data use cases



©2022 Databricks Inc. — All rights reserved

The Databricks Lakehouse Platform

Open

Unify your data ecosystem with open source standards and formats.

Built on the innovation of some of the most successful open source data projects in the world

30 Million+

Monthly downloads

Apache Spark

Delta Lake

mlflow

Koalas

redash

©2022 Databricks Inc. — All rights reserved

The Databricks Lakehouse Platform

Open

Unify your data ecosystem with open source standards and formats.

450+

Partners across the data landscape

Visual ETL & Data Ingestion

Azure Data Factory, Fivetran, Confluent, Informatica, talend, Qlik, StreamSets, precisely, Infoworks, wanDISCO, TRIFACTA, dbt, mongoDB, MATILLION

Business Intelligence

tableau, Power BI, Azure Synapse, Qlik, plotly, looker, Google BigQuery, ThoughtSpot, TIBCO Spotfire, Amazon Redshift

Machine Learning

Amazon SageMaker, Azure Machine Learning, MathWorks, Google AI Platform, RStudio, H2O.ai

databricks Lakehouse Platform

Microsoft Azure, aws, Google Cloud

Data Providers

dun & bradstreet, SAFEGRAPH, experian, yipitDATA, arm INSIGHT

Centralized Governance

AWS Glue, Informatica, Alation, PRIVACERA, IMMUTA, collibra

Top Consulting & SI Partners

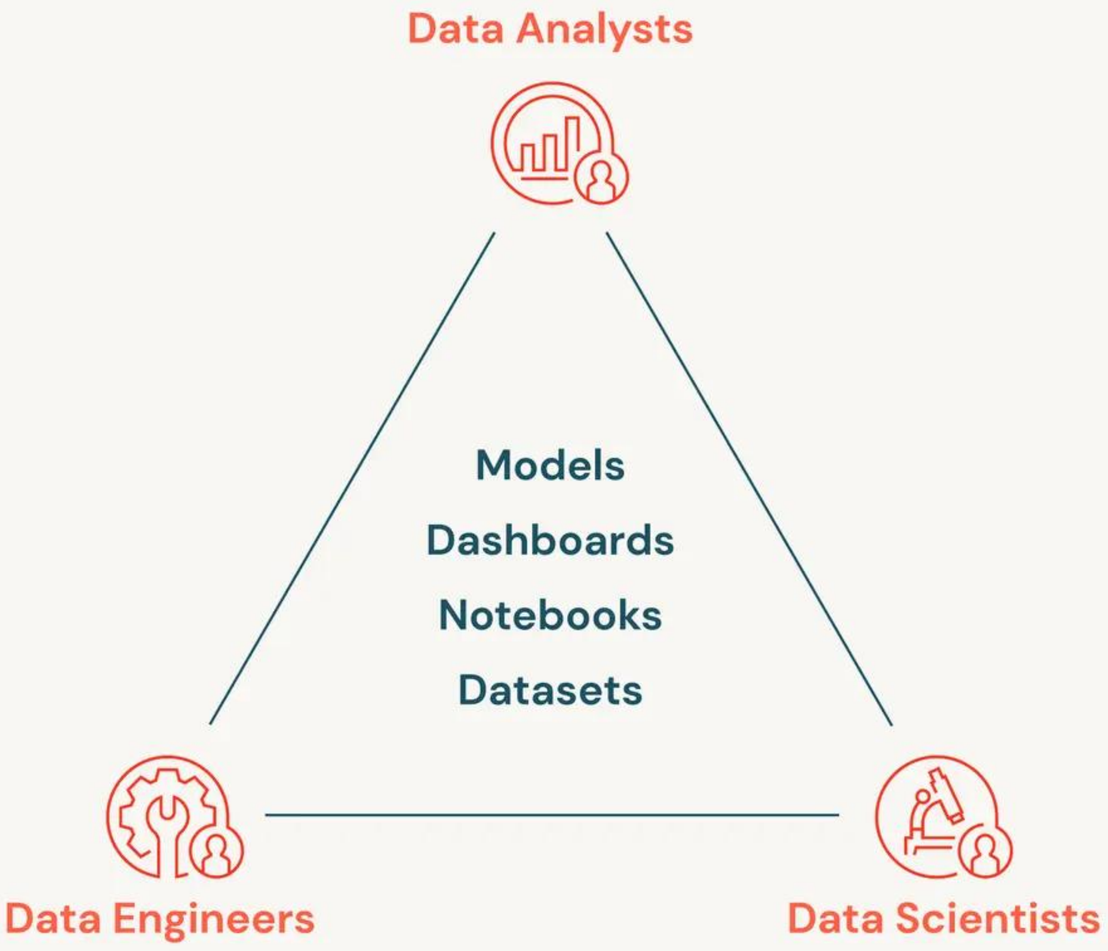
Insight, NEUDESIC, PARIVEDA, accenture, avanade, wipro, Infosys, Tech Mahindra, GO DATA DRIVEN, ZEAL, AGILISIUM, Hitachi Solutions, ONICA, Capgemini, Cognizant, Deloitte., ELASTACLOUD, OPENVALUE, b.telligent, kumulus, BlueShift, deep_dive, Mindtree, slalom, TATA CONSULTANCY SERVICES, JUST ANALYTICS, Intellify, Versor

©2022 Databricks Inc. — All rights reserved

The Databricks Lakehouse Platform

Collaborative

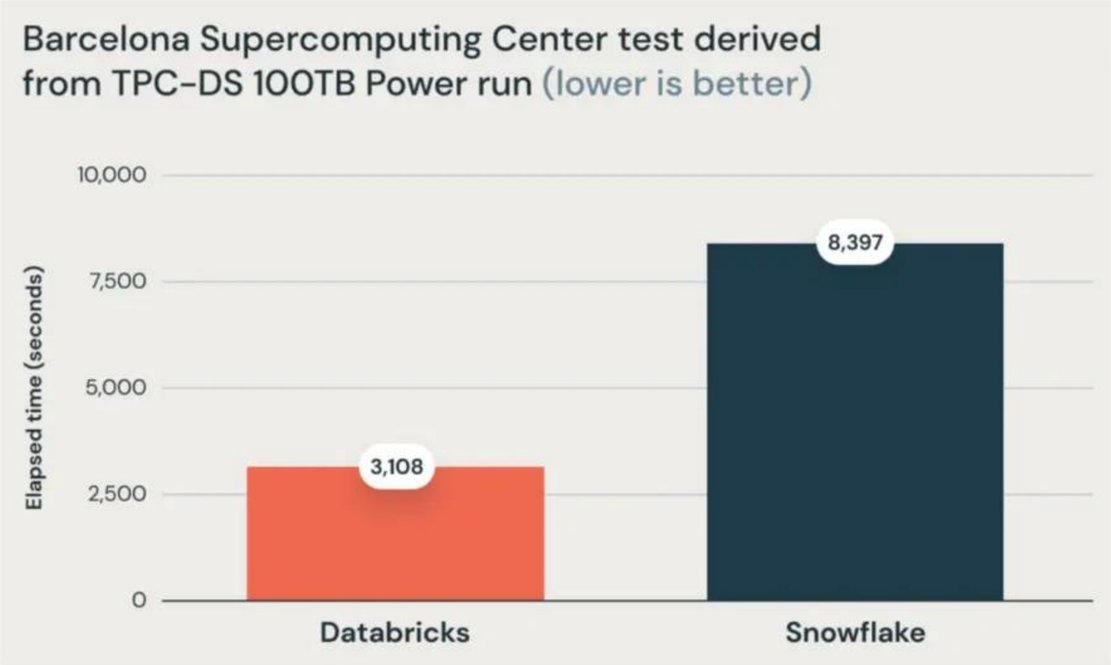
Unify your data teams to collaborate across the entire data and AI workflow



©2022 Databricks Inc. — All rights reserved

TPC-DS

Databricks SQL set **official data warehousing performance record** – outperformed the previous record by 2.2x.



©2022 Databricks Inc. — All rights reserved

Spark Overview

©2022 Databricks Inc. — All rights reserved

- De-facto standard unified analytics engine for big data processing

- Largest open-source project in data processing

- Technology created by the founders of Databricks

©2022 Databricks Inc. — All rights reserved

Spark Benefits

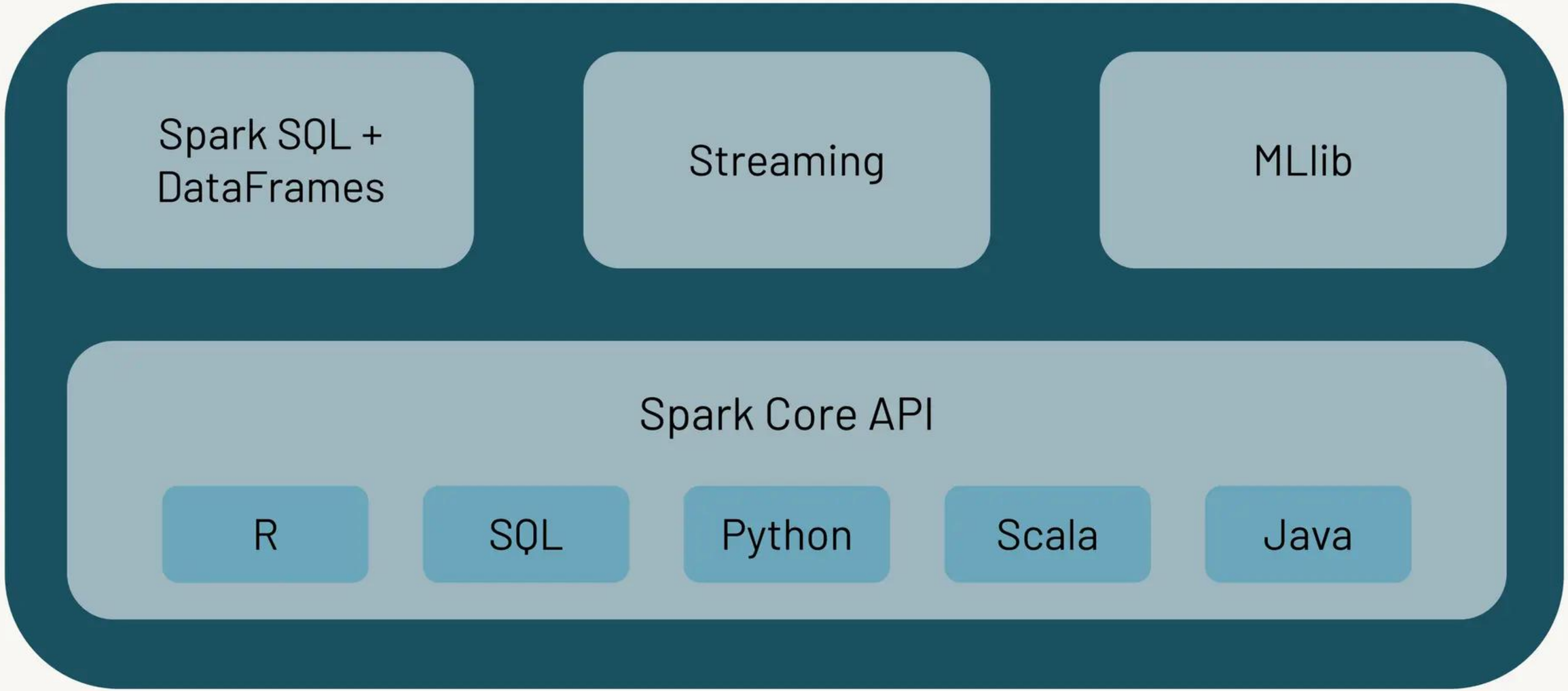
Fast

Easy to Use

Unified

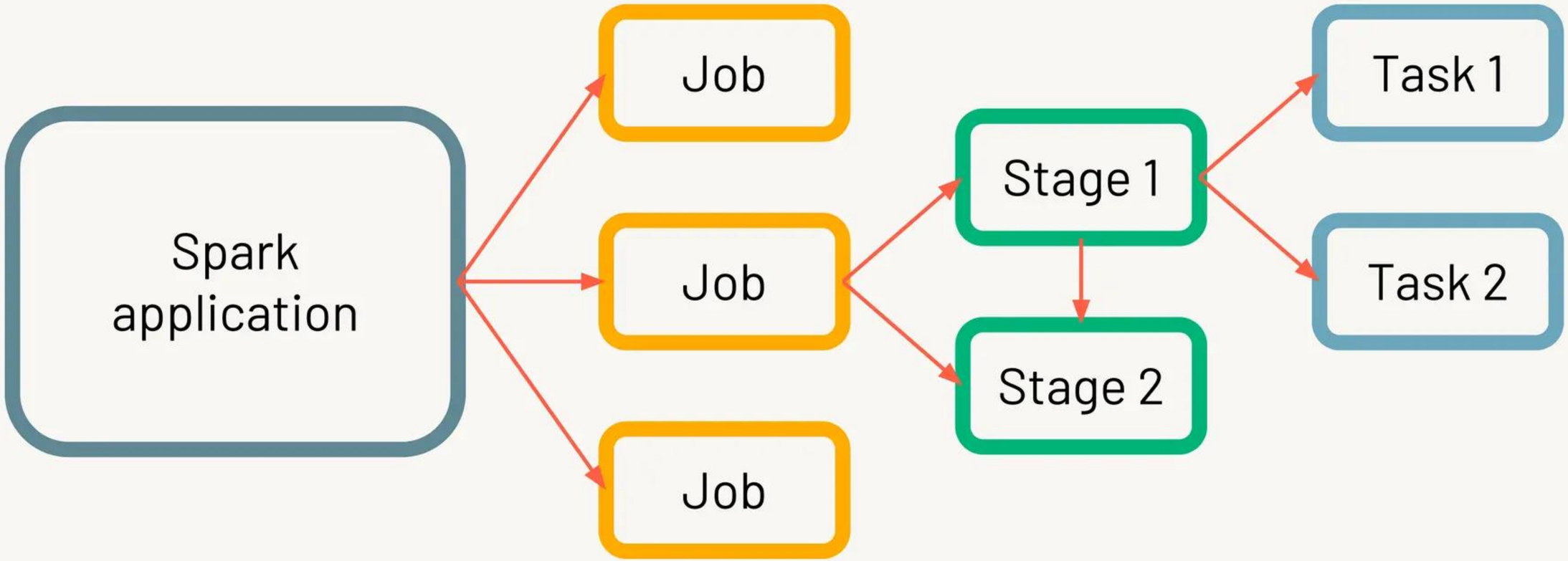
©2022 Databricks Inc. — All rights reserved

Spark API



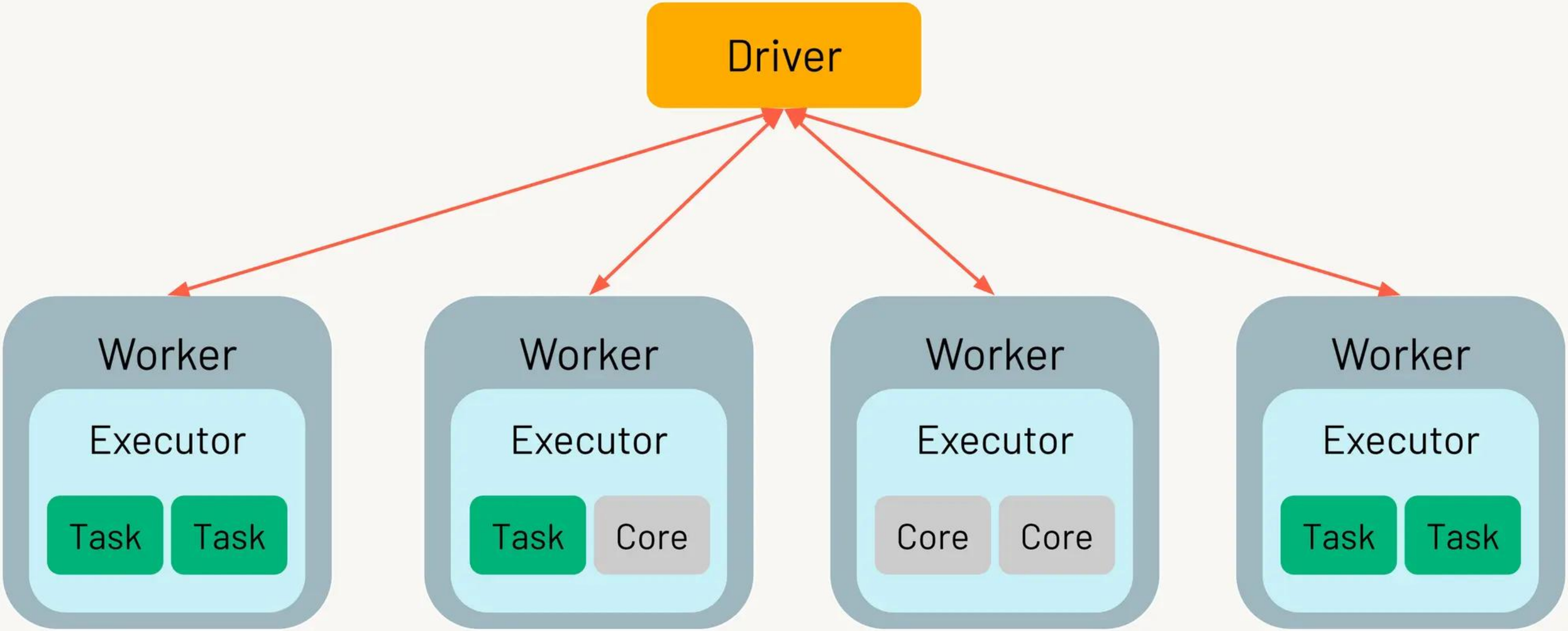
©2022 Databricks Inc. — All rights reserved

Spark Execution



©2022 Databricks Inc. — All rights reserved

Spark Cluster



©2022 Databricks Inc. — All rights reserved

Bonus: Magic commands in Notebook cells

Magic commands allow you to override default languages as well as a few auxiliary commands that run utilities/commands. For example:

1. %python, %r, %scala, %sql Switch languages in a command cell
2. %sh Run shell code (runs only on Spark Driver, and not the Workers)
3. %fs Shortcut for **dbutils** filesystem commands
4. %md Markdown for styling the display
5. %run Execute a remote Notebook from a Notebook
6. %pip Install new Python libraries

©2022 Databricks Inc. — All rights reserved

CSV

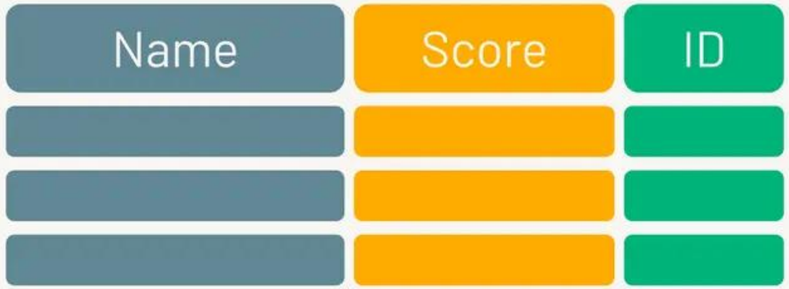
Comma Separated Values

```
item_id, name, price, qty
M_PREM_Q, Premium Queen Mattress, 1795, 35
M_STAN_F, Standard Full Mattress, 945, 24
M_PREM_F, Premium Full Mattress, 1695, 45
M_PREM_T, Premium Twin Mattress, 1095, 18
```

©2022 Databricks Inc. — All rights reserved

Parquet

A columnar storage format



Row-Oriented data on disk

Column-Oriented data on disk

©2022 Databricks Inc. — All rights reserved

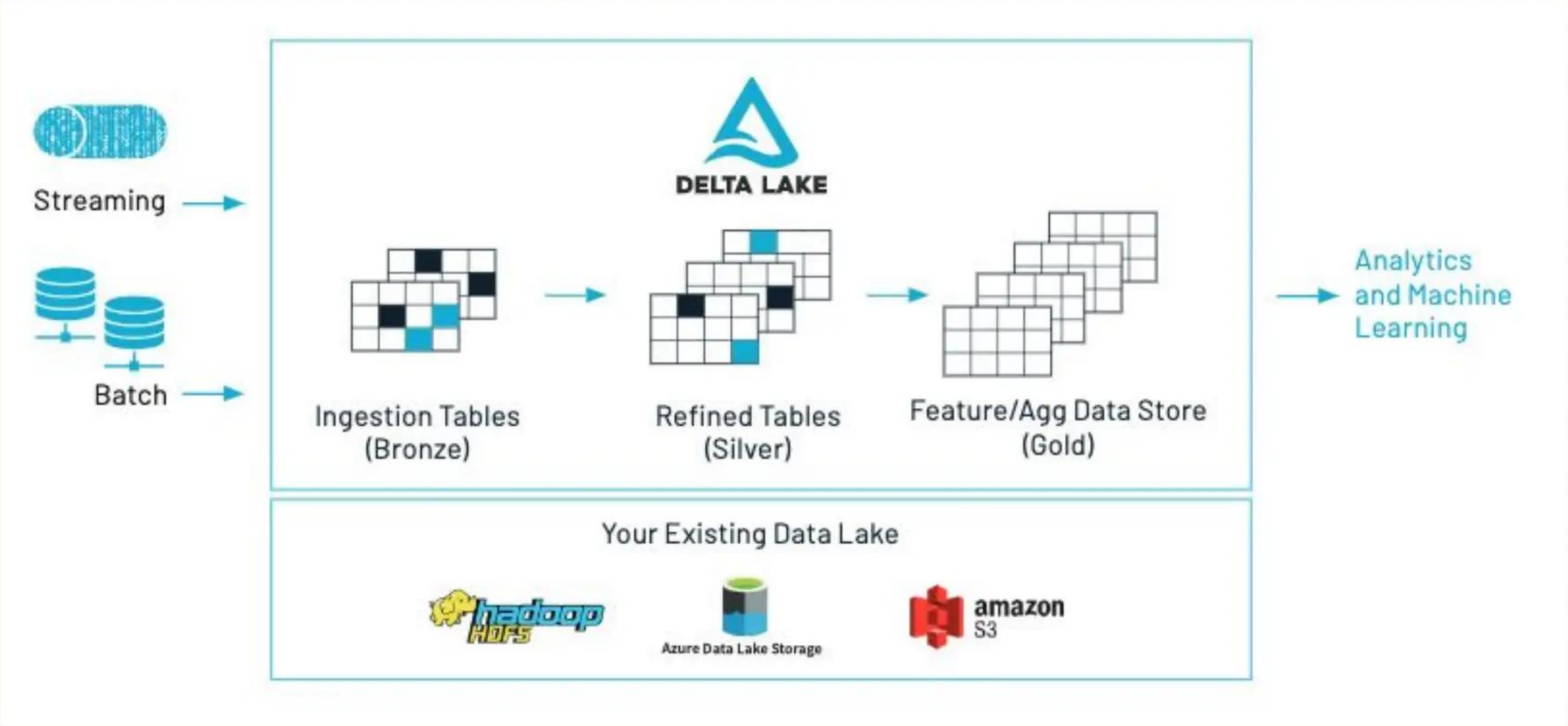
Delta Lake

Technology designed to be used with Apache Spark to build robust data lakes

DELTA LAKE

©2022 Databricks Inc. — All rights reserved

Open-source Storage Layer



©2022 Databricks Inc. — All rights reserved

Delta Lake's Key Features

- ACID transactions
- Time travel (data versioning)
- Schema enforcement and evolution
- Audit history
- Parquet format
- Compatible with Apache Spark API

©2022 Databricks Inc. — All rights reserved

Module 3

Functions

©2022 Databricks Inc. — All rights reserved

Functions

Aggregation

Datetimes

Complex Types

Additional Functions

User-Defined Functions

©2022 Databricks Inc. — All rights reserved

Module 4

Performance

©2022 Databricks Inc. — All rights reserved

Performance

Spark Architecture

Query Optimization

Partitioning

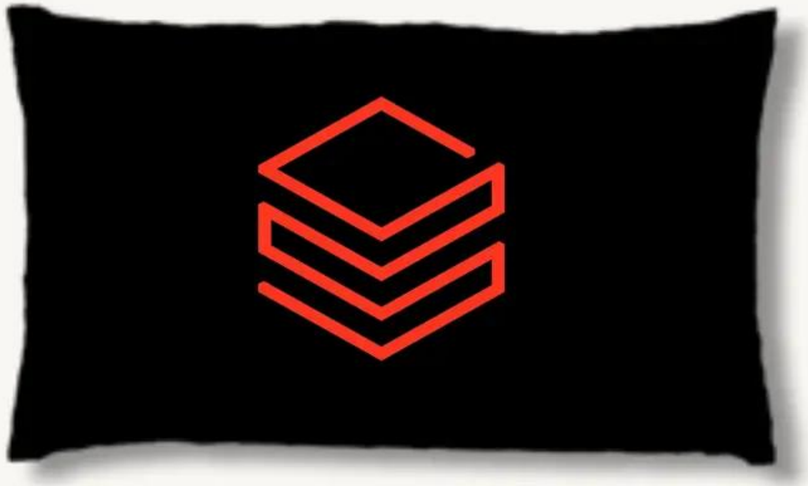
©2022 Databricks Inc. — All rights reserved

Spark Architecture

Spark Cluster

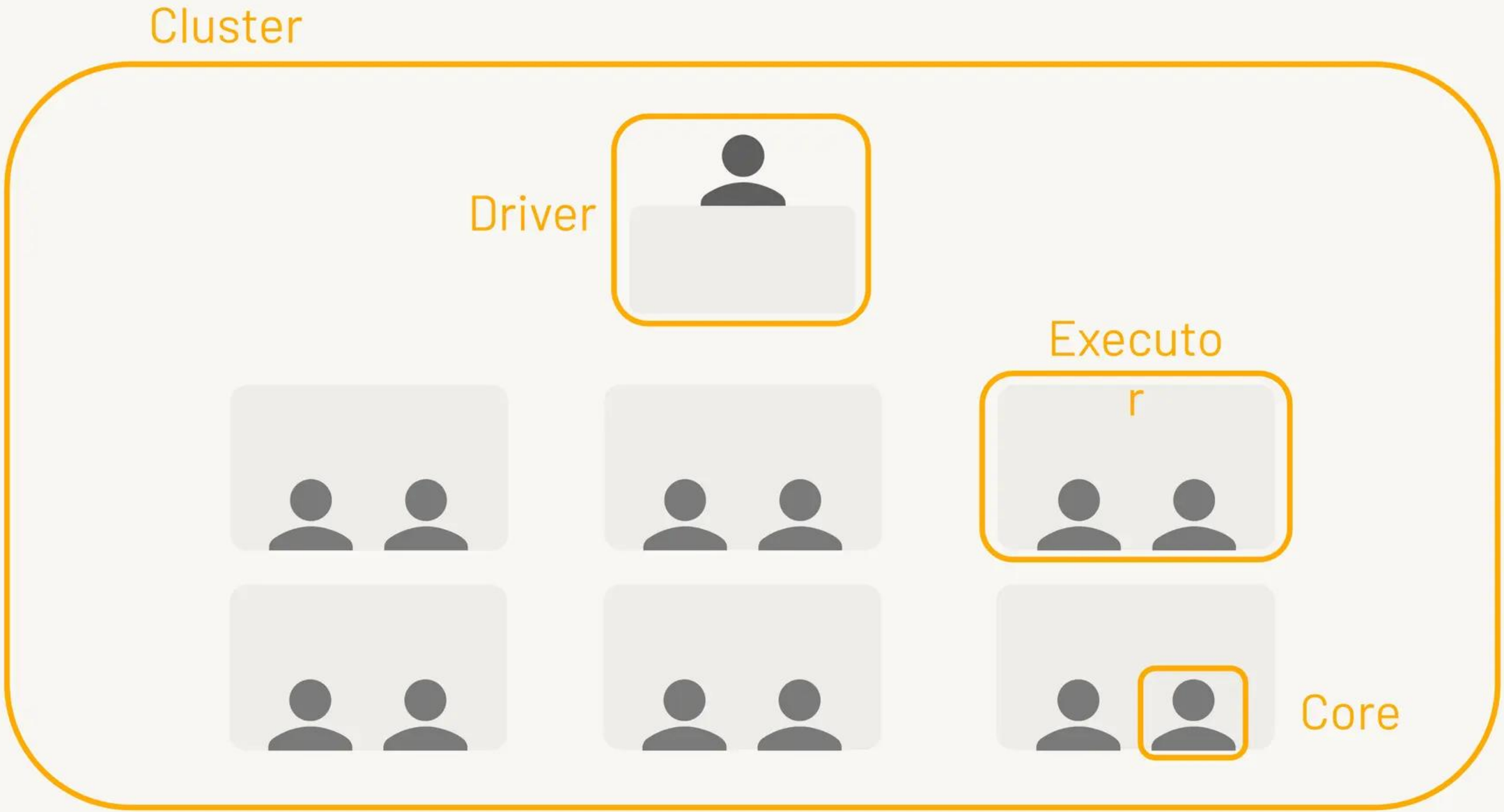
Spark Execution

©2022 Databricks Inc. — All rights reserved

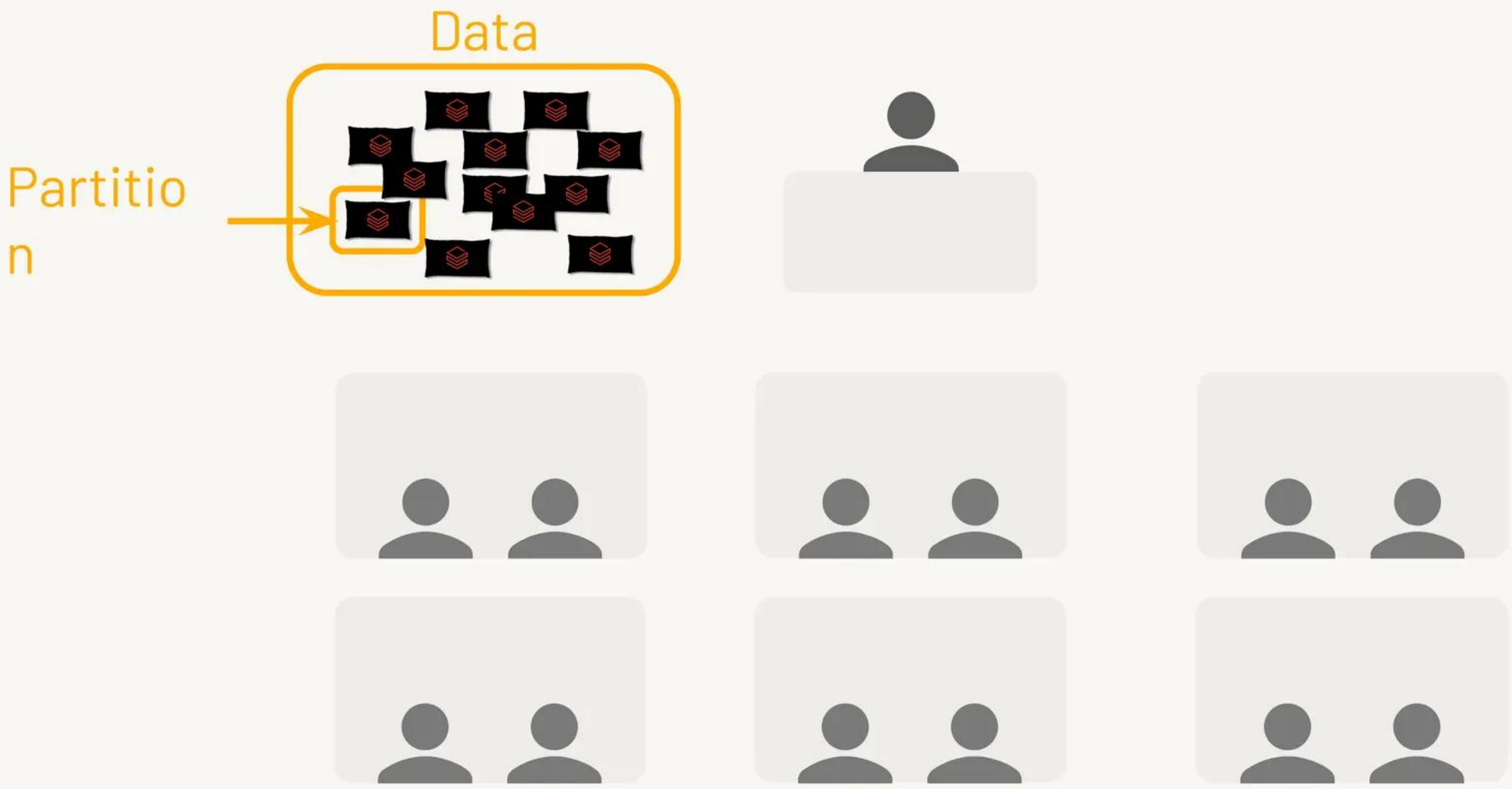


Scenario 1: Filter out brown pieces from candy bags

©2022 Databricks Inc. — All rights reserved

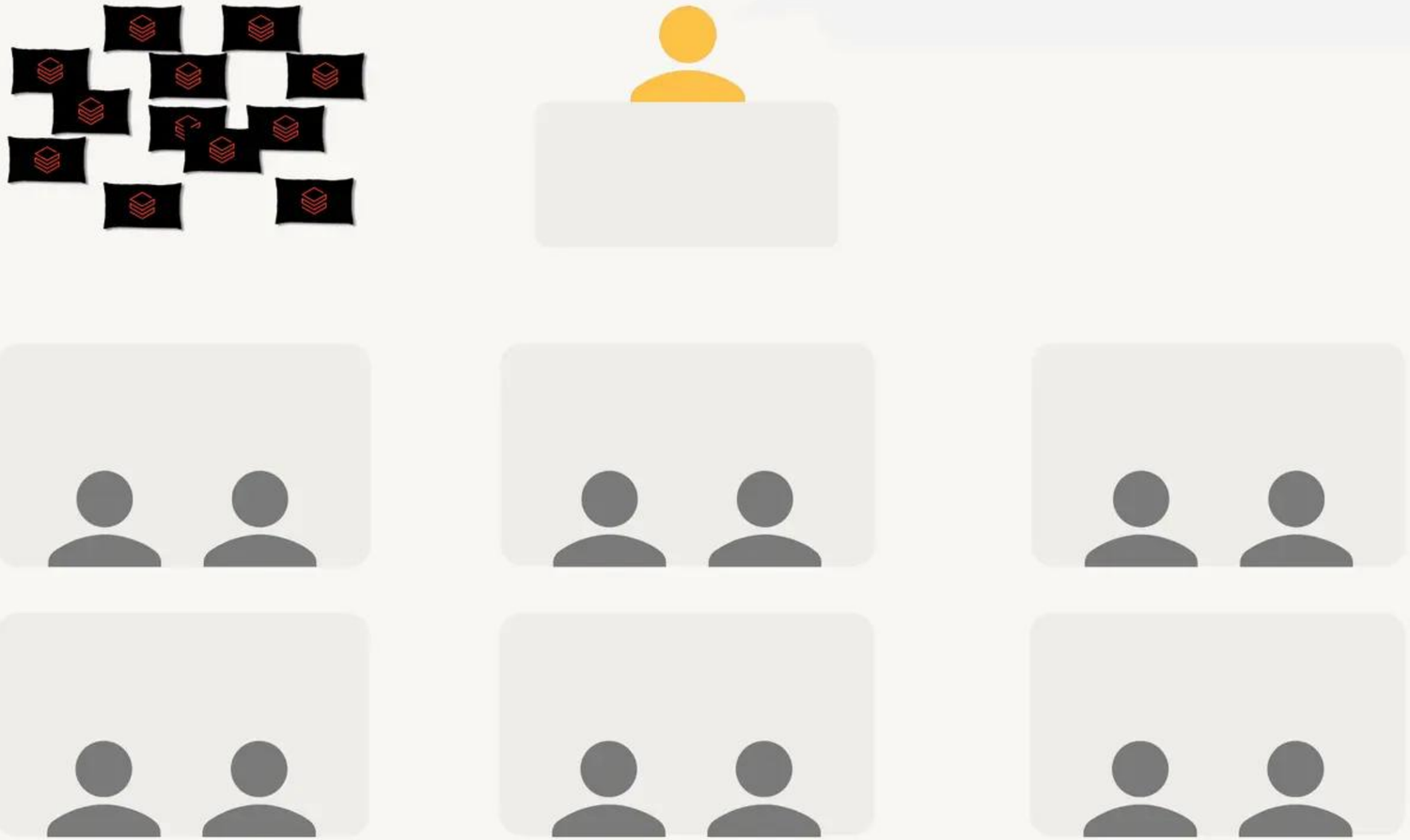


©2022 Databricks Inc. — All rights reserved



©2022 Databricks Inc. — All rights reserved

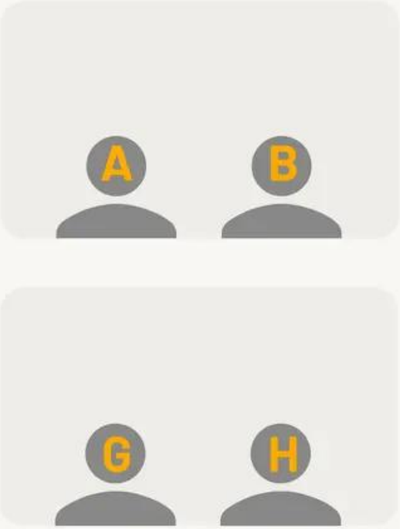
We need filter out brown pieces from these candy bags

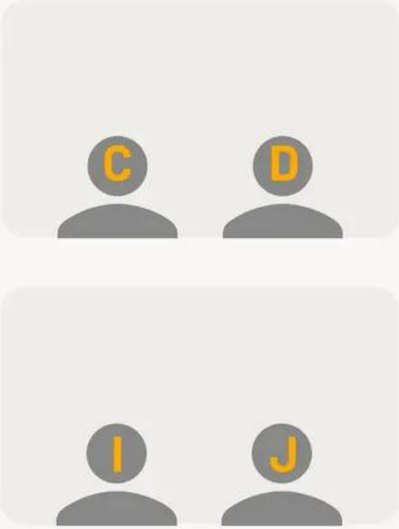


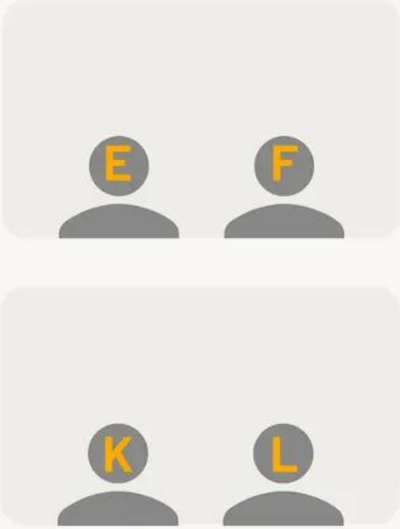
©2022 Databricks Inc. — All rights reserved

©2022 Databricks Inc. — All rights reserved

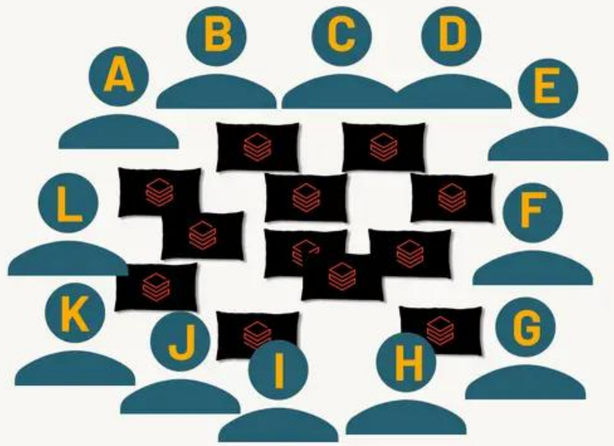
Student A get bag # 1
Student B get bag # 2
Student C get bag # 3
...



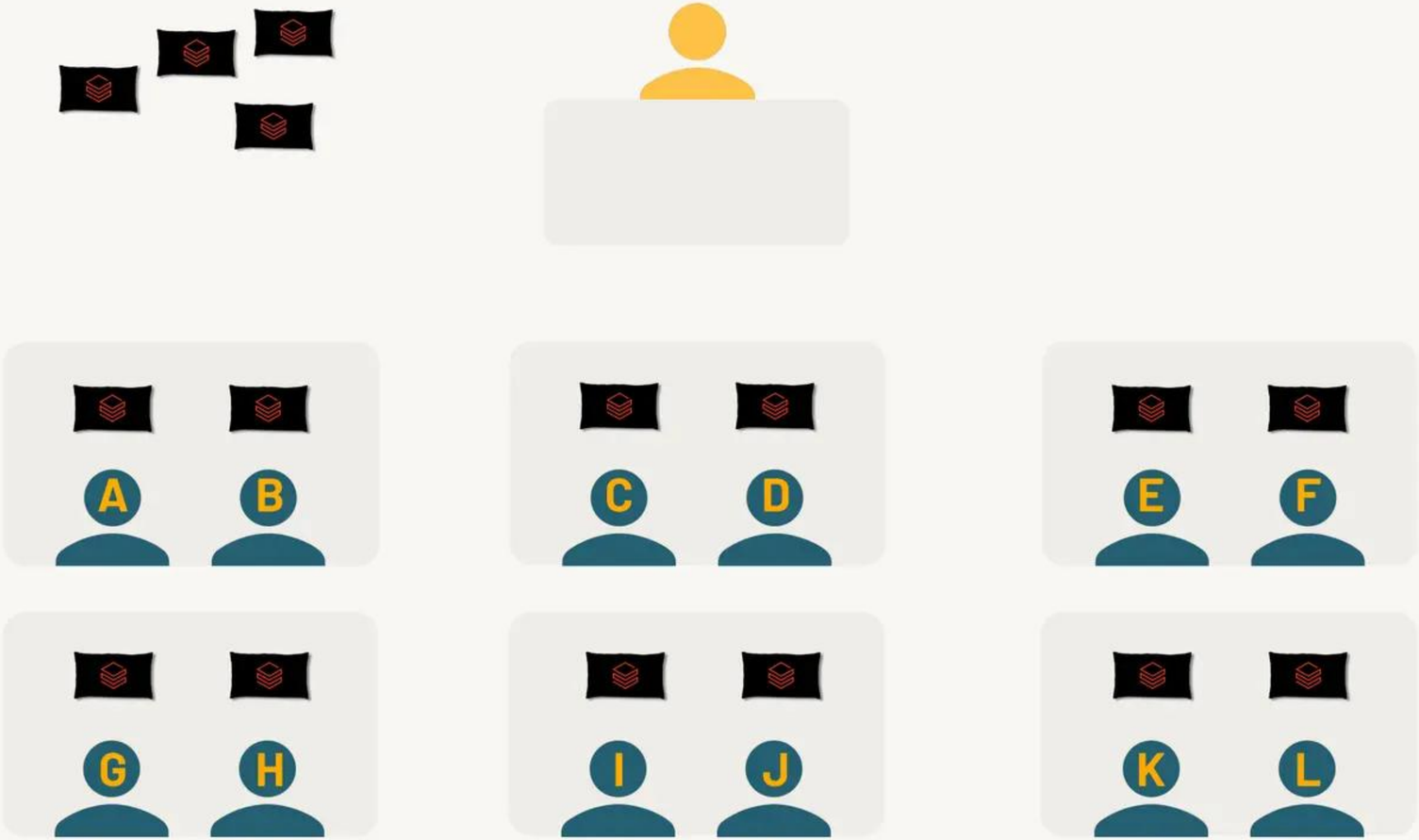




©2022 Databricks Inc. — All rights reserved

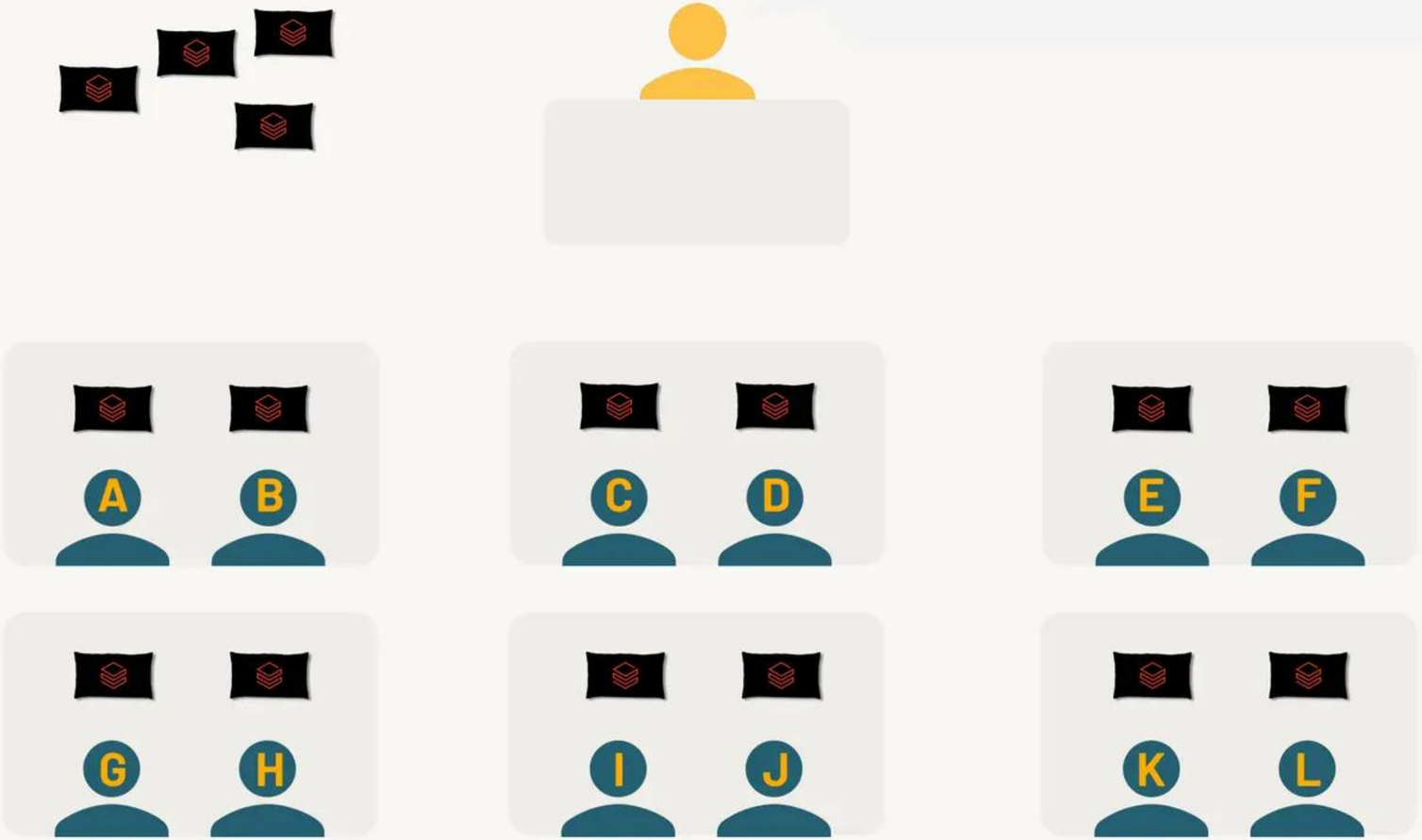


©2022 Databricks Inc. — All rights reserved



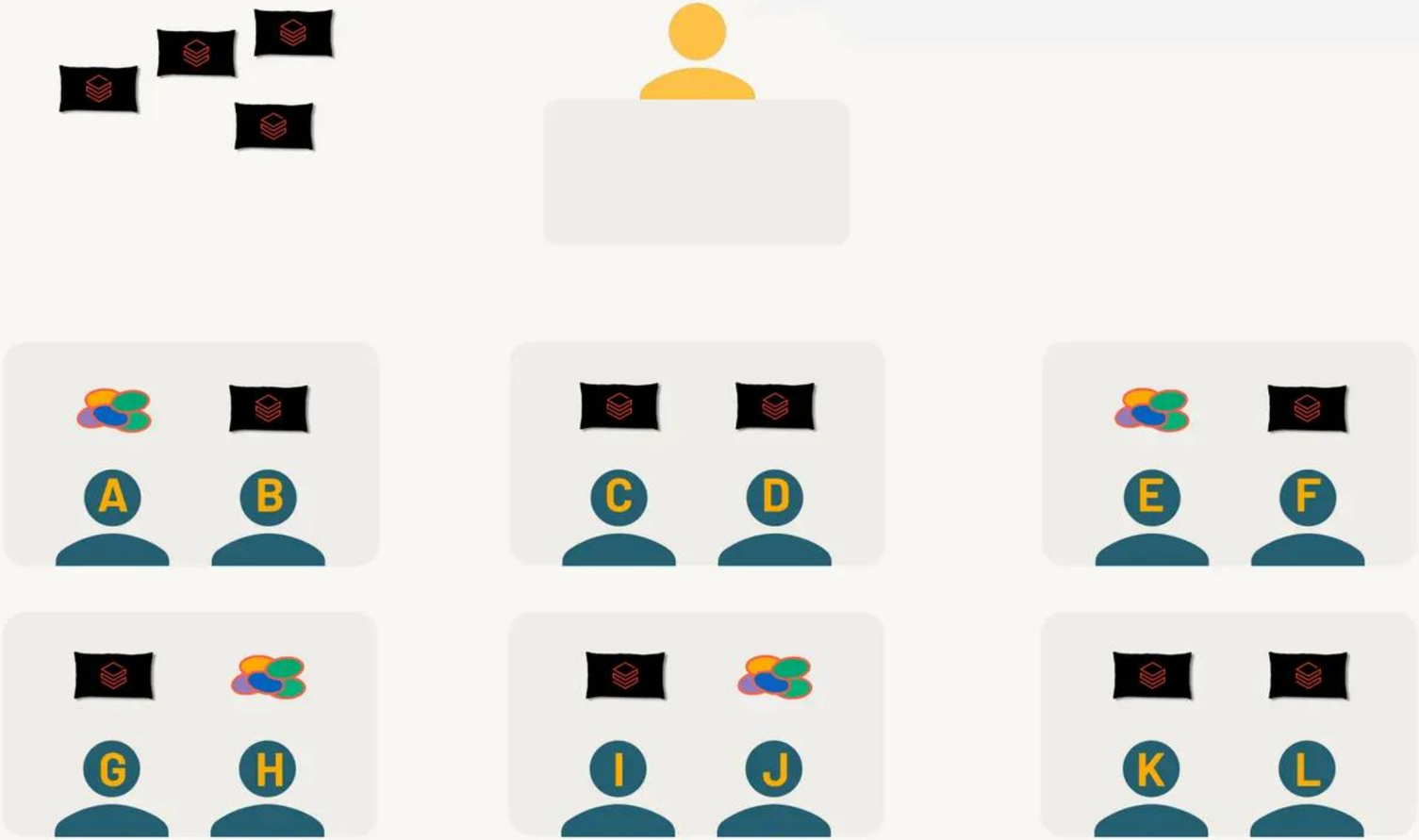
©2022 Databricks Inc. — All rights reserved

Eliminate the brown candy pieces and pile the rest in the corner

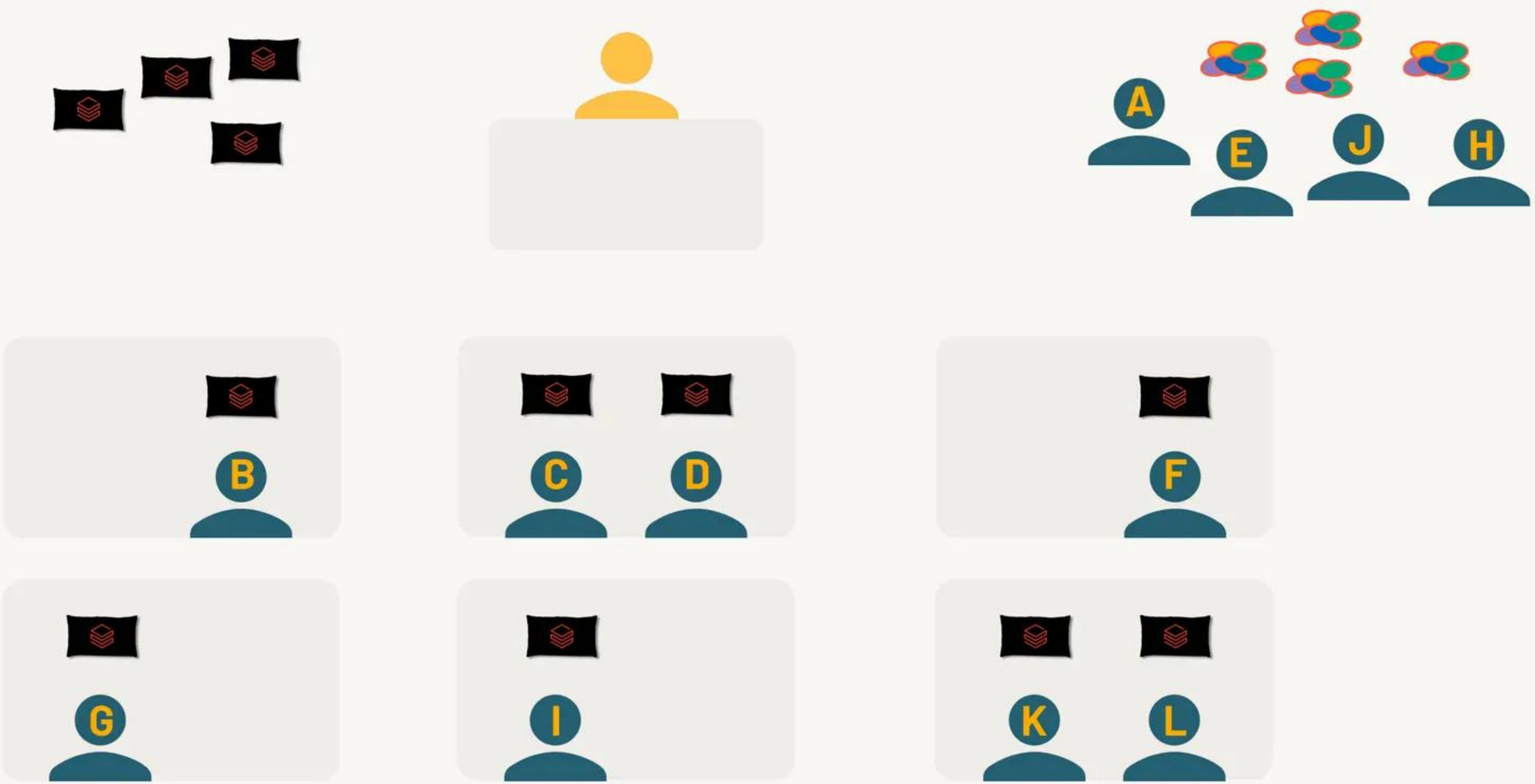


©2022 Databricks Inc. — All rights reserved

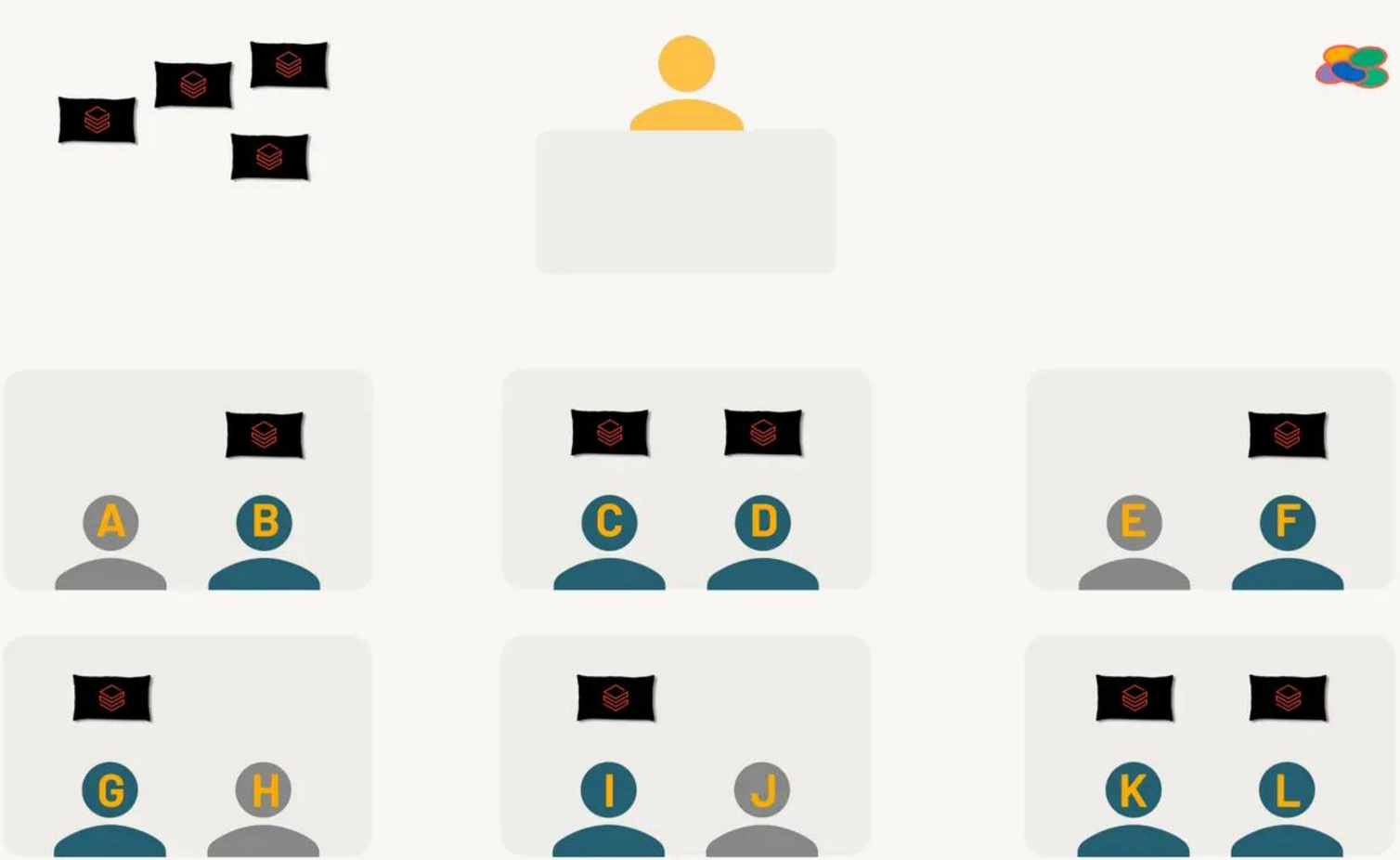
Eliminate the brown candy pieces and pile the rest in the corner



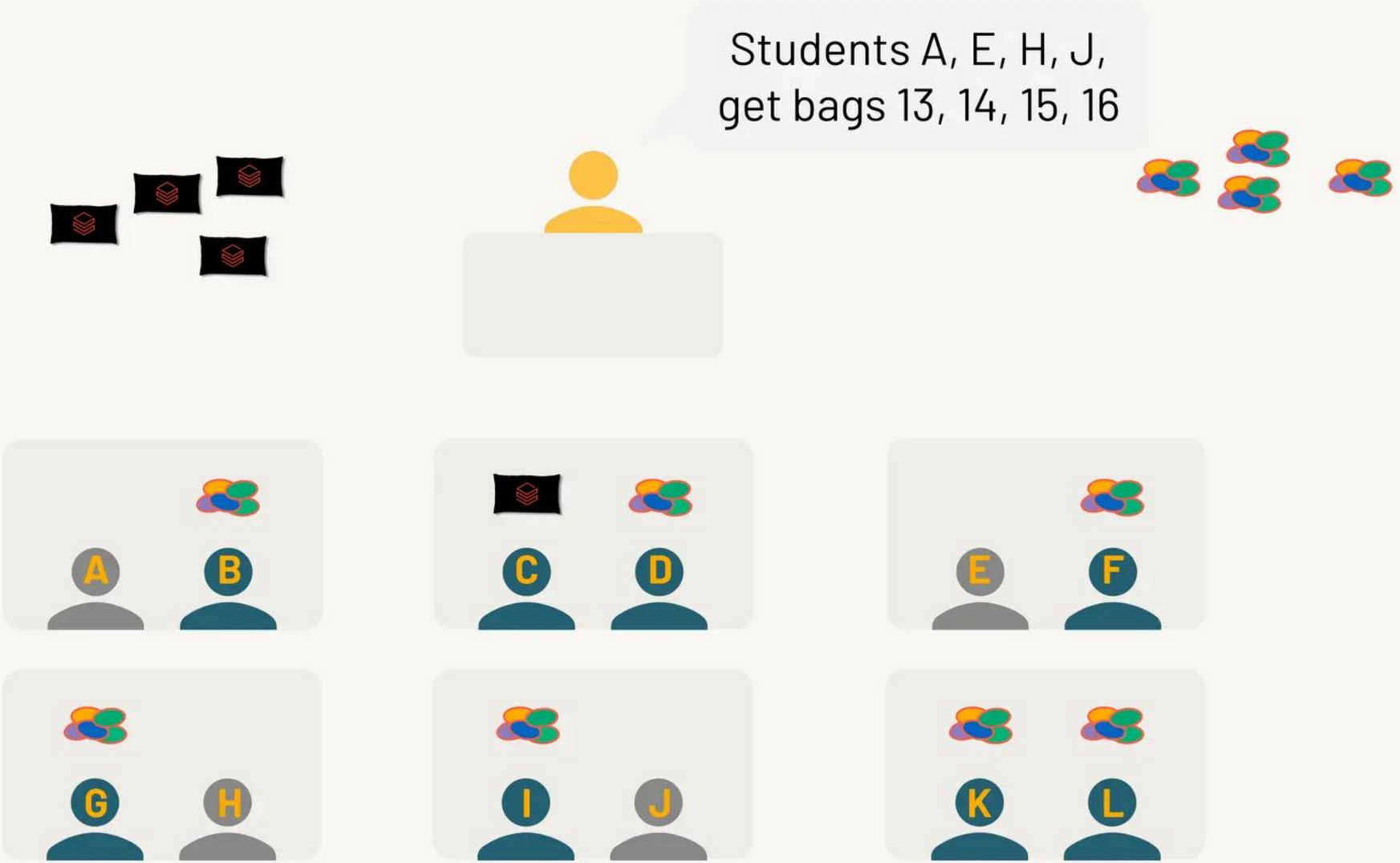
©2022 Databricks Inc. — All rights reserved



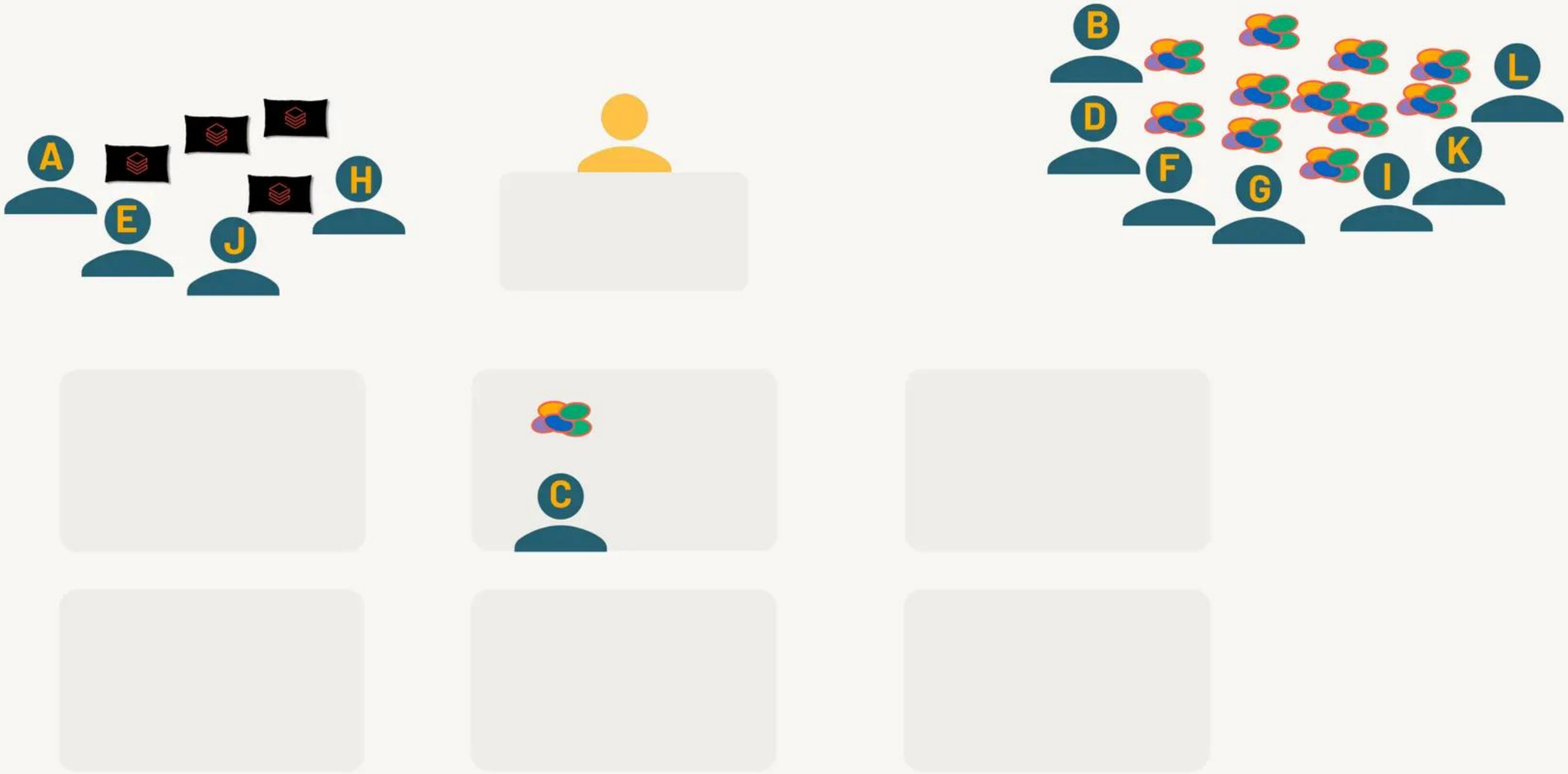
©2022 Databricks Inc. — All rights reserved



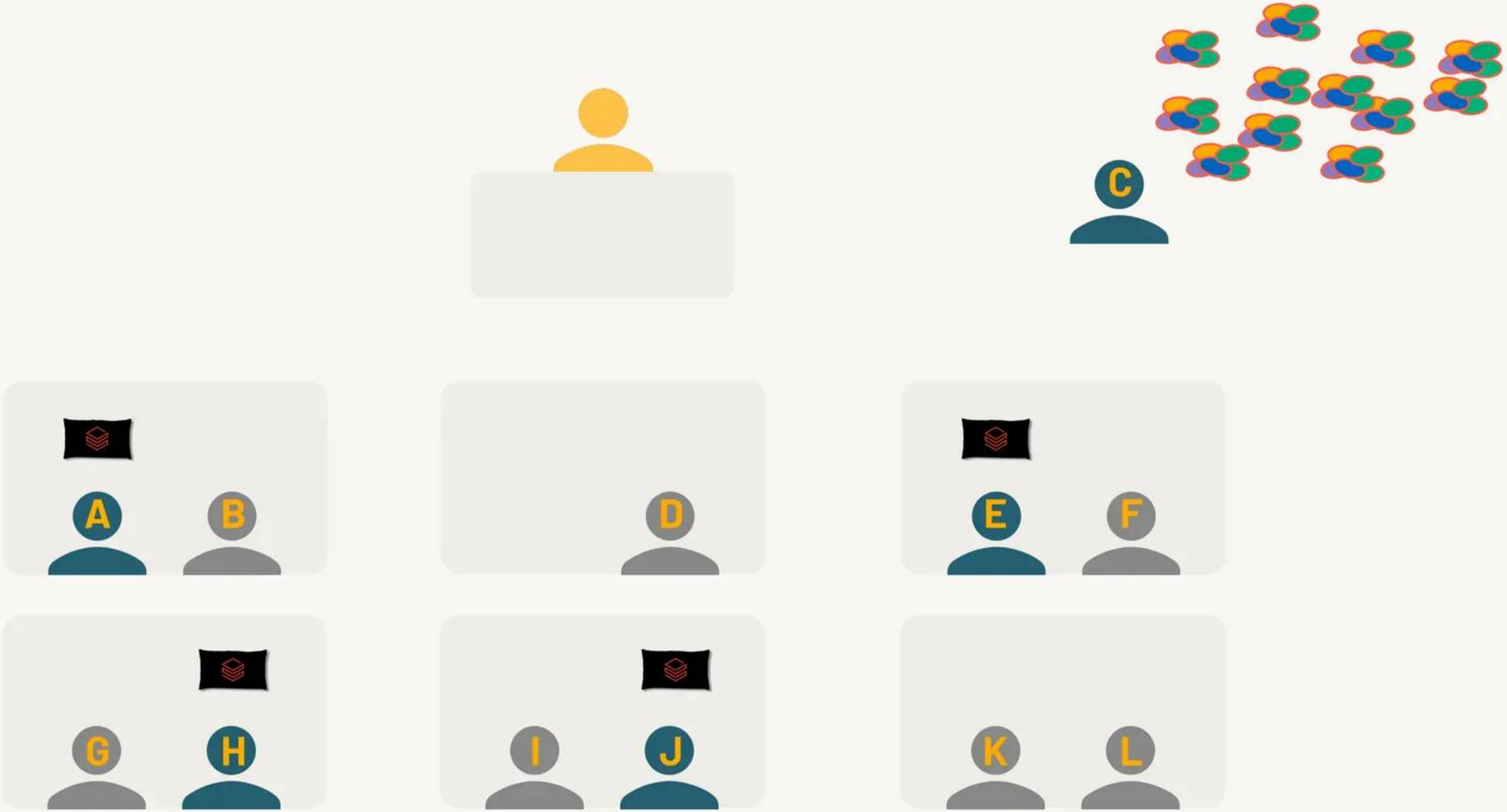
©2022 Databricks Inc. — All rights reserved



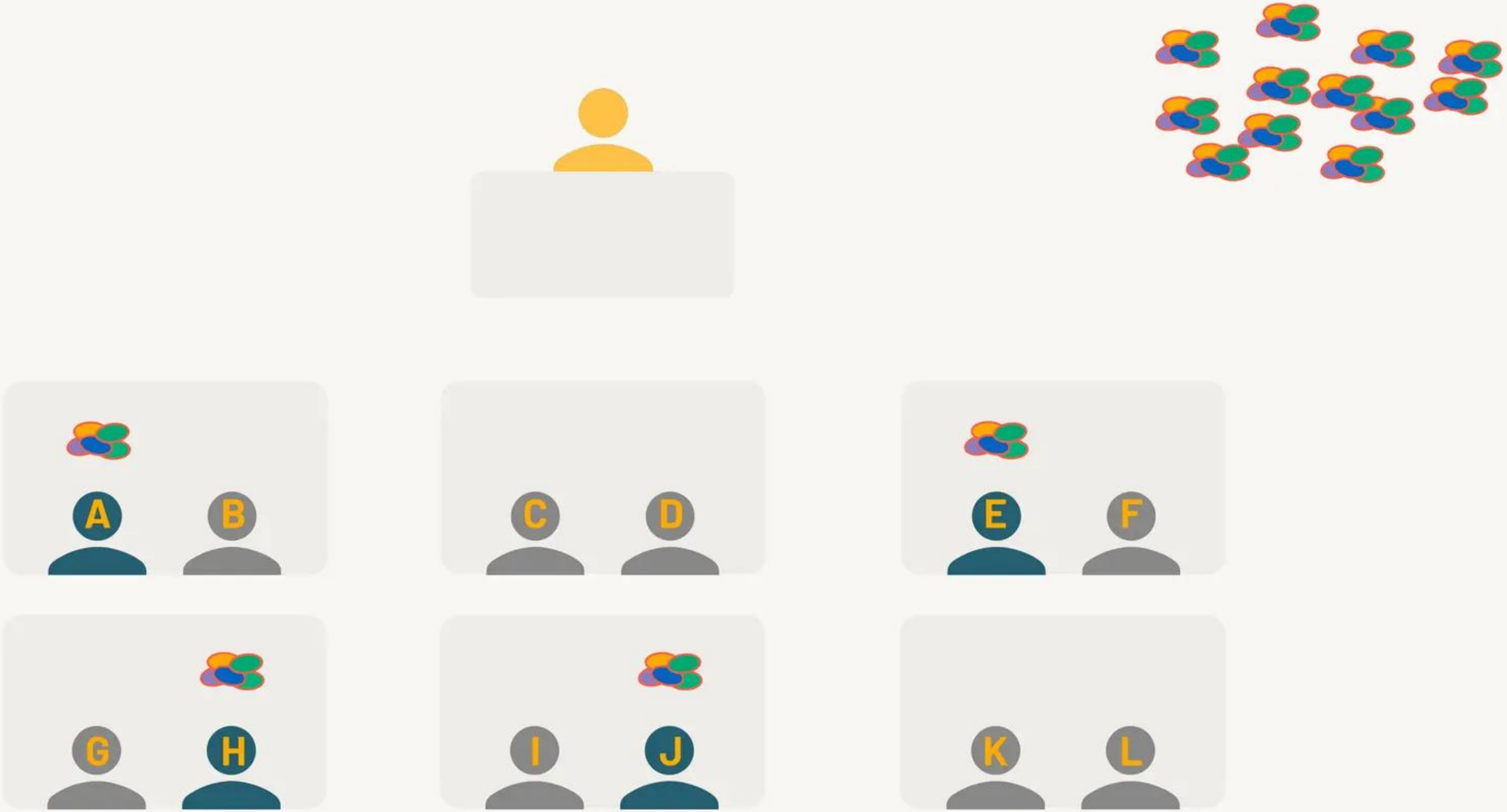
©2022 Databricks Inc. — All rights reserved



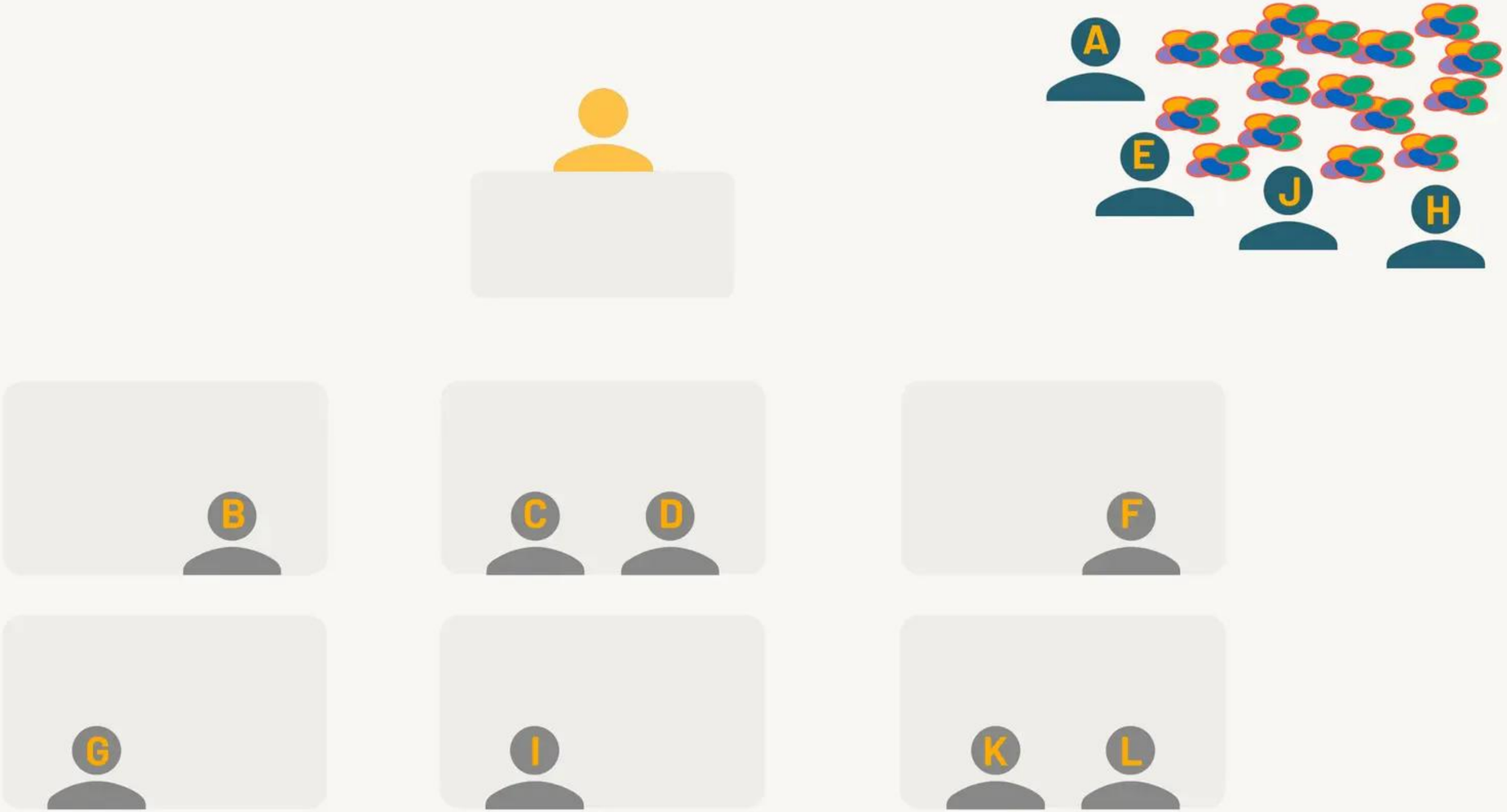
©2022 Databricks Inc. — All rights reserved



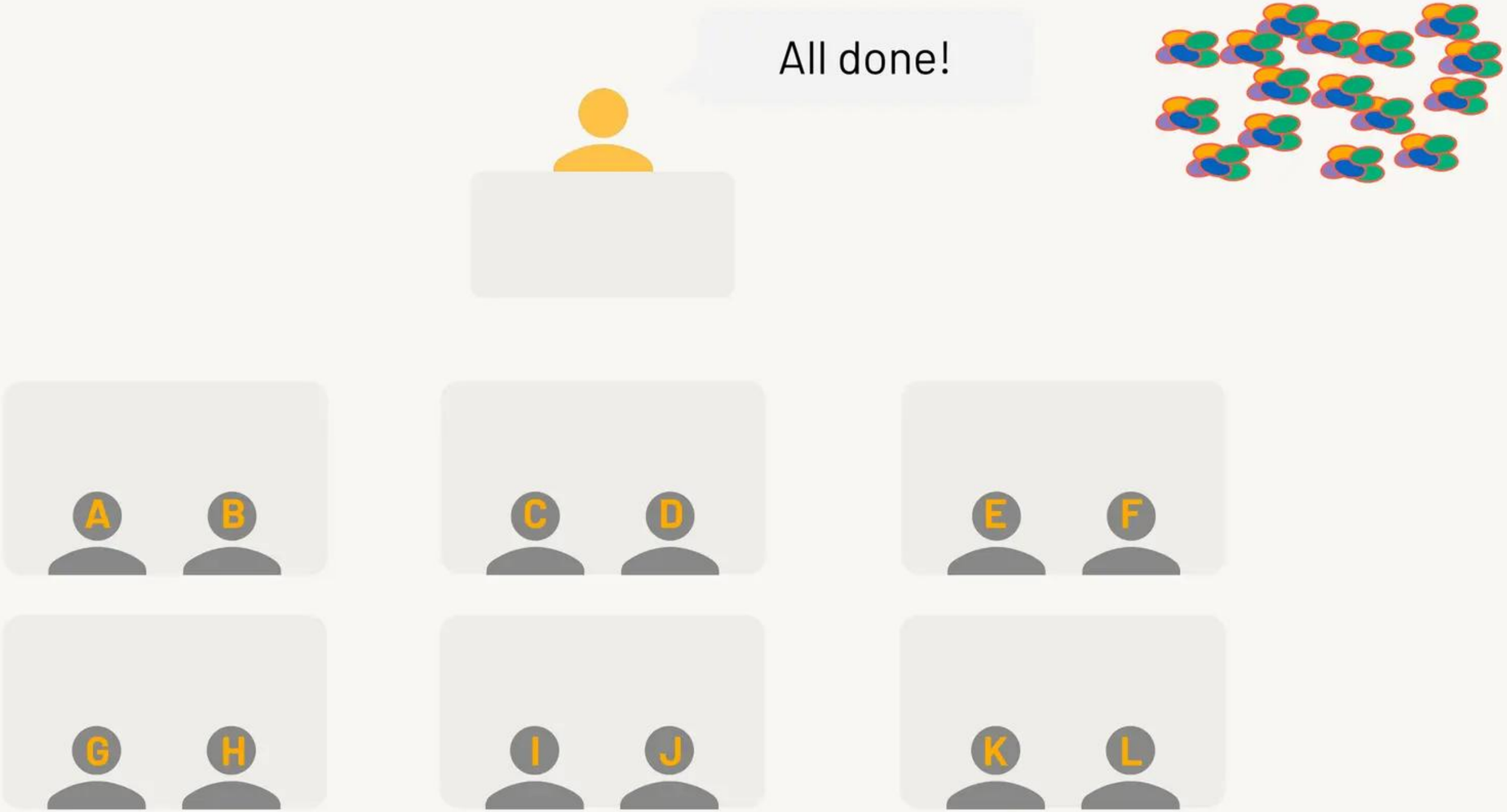
©2022 Databricks Inc. — All rights reserved



©2022 Databricks Inc. — All rights reserved



©2022 Databricks Inc. — All rights reserved



©2022 Databricks Inc. — All rights reserved

Scenario 2: Count total pieces in candy bags

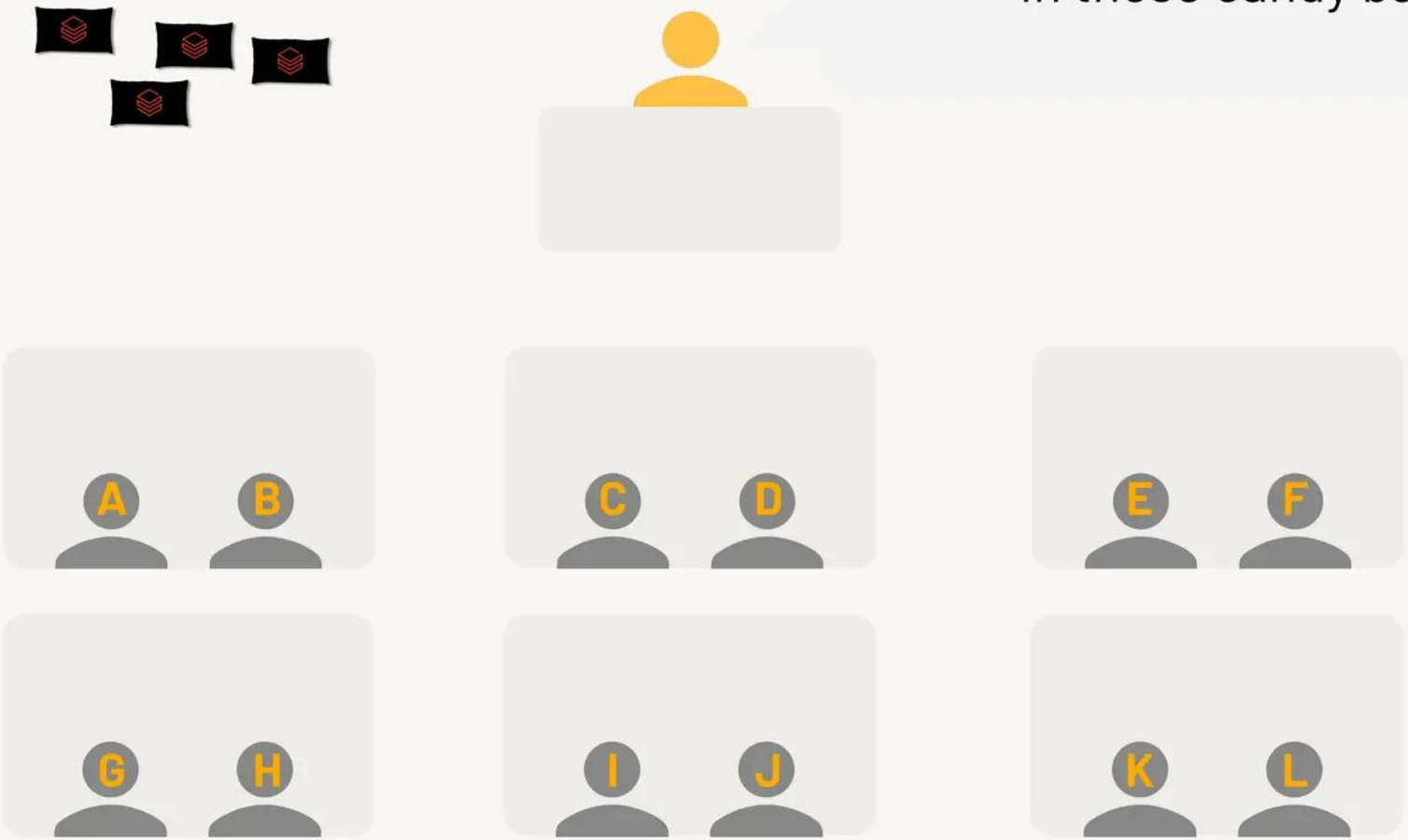
©2022 Databricks Inc. — All rights reserved

Stage 1: Local Count

©2022 Databricks Inc. — All rights reserved

Stage 1: Local Count

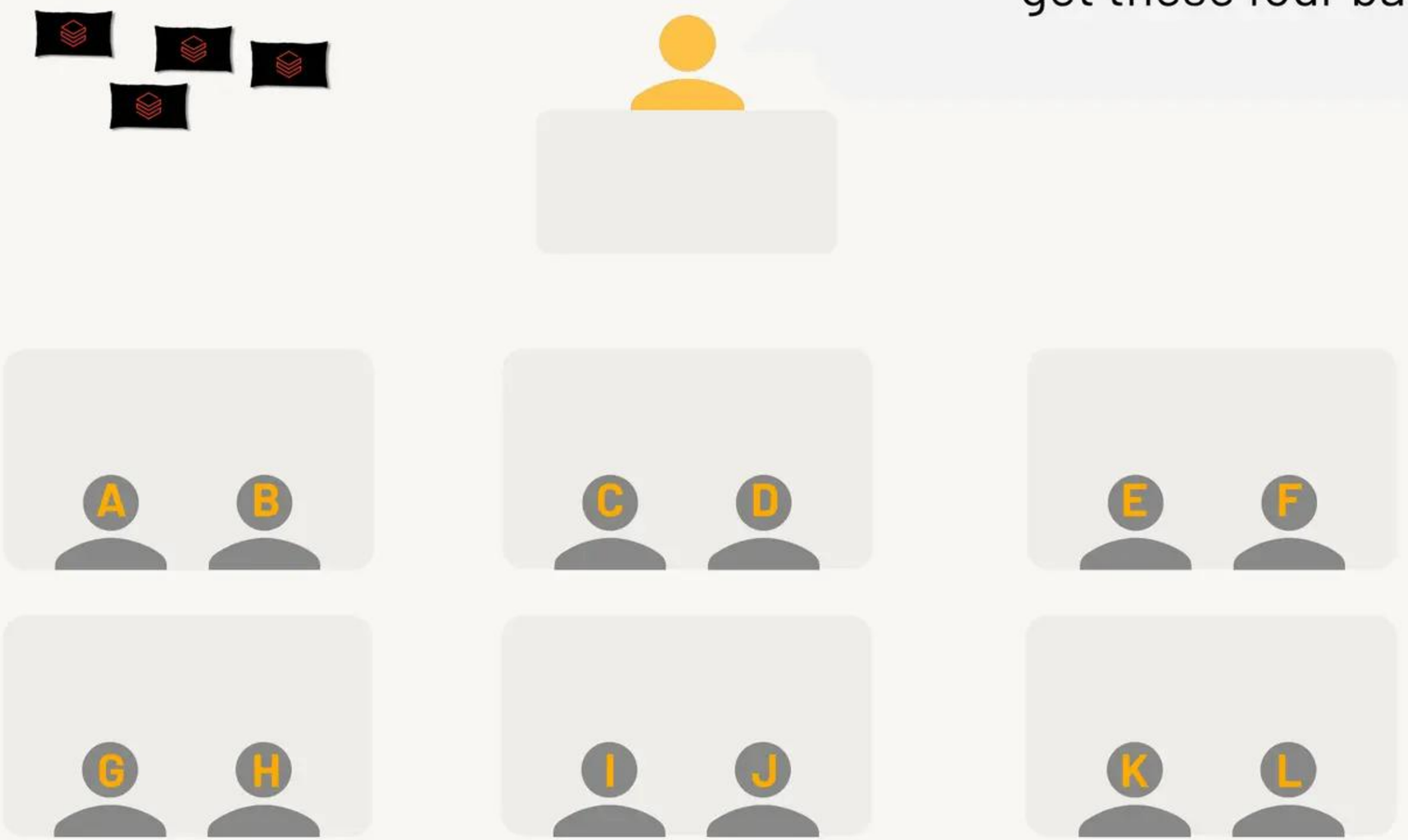
We need to count the total pieces in these candy bags



©2022 Databricks Inc. — All rights reserved

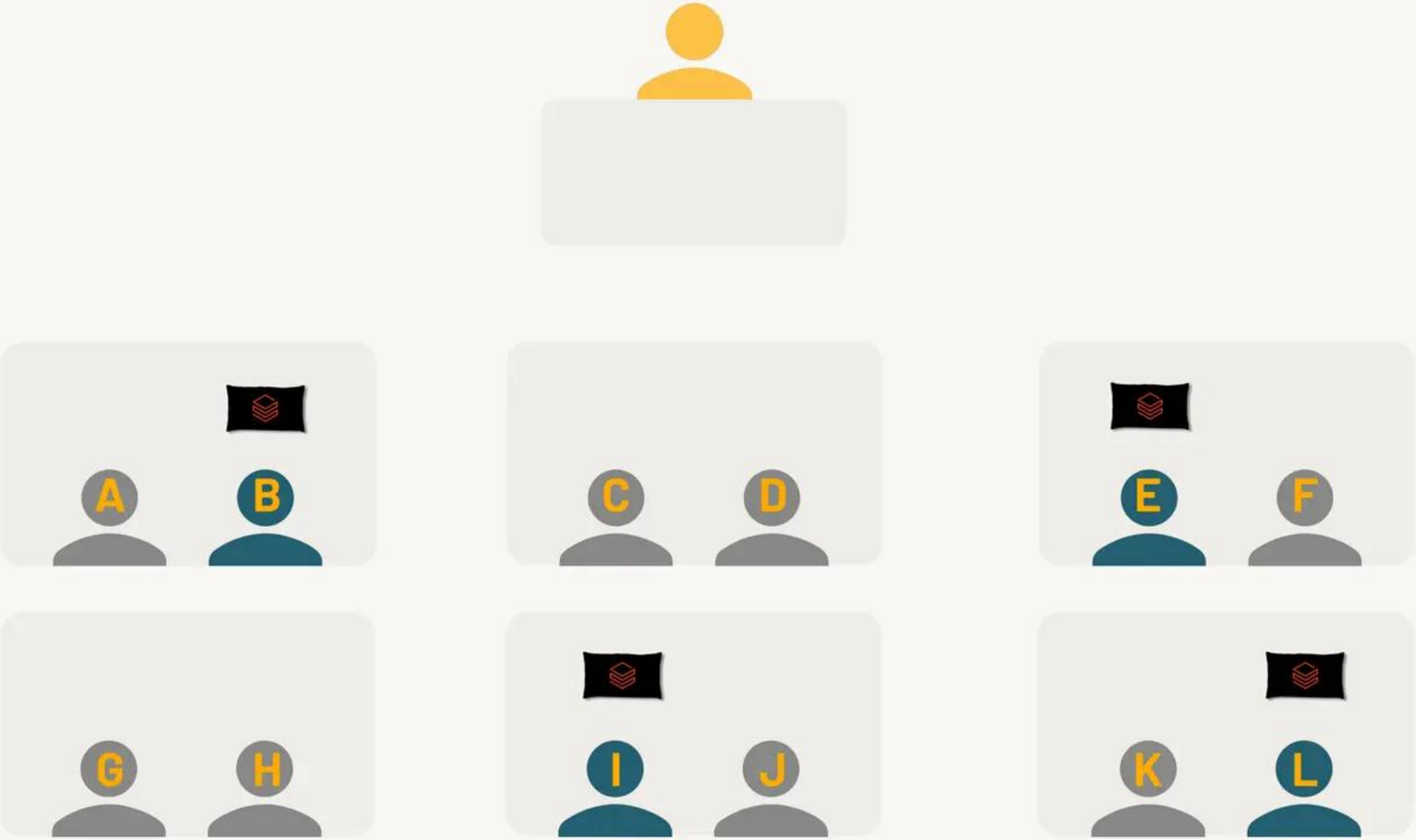
Stage 1: Local Count

Students B, E, I, L,
get these four bags



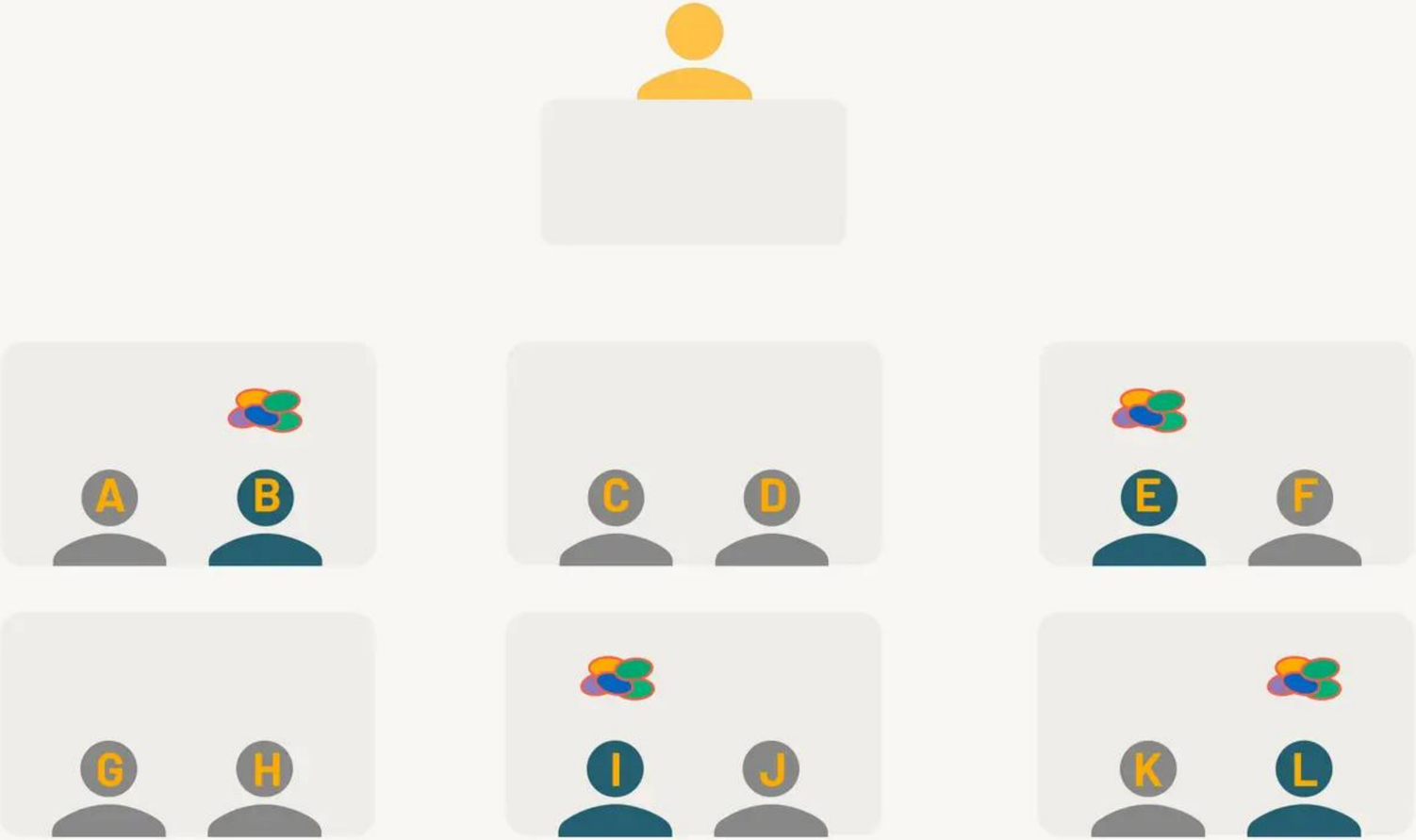
©2022 Databricks Inc. — All rights reserved

Stage 1: Local Count



©2022 Databricks Inc. — All rights reserved

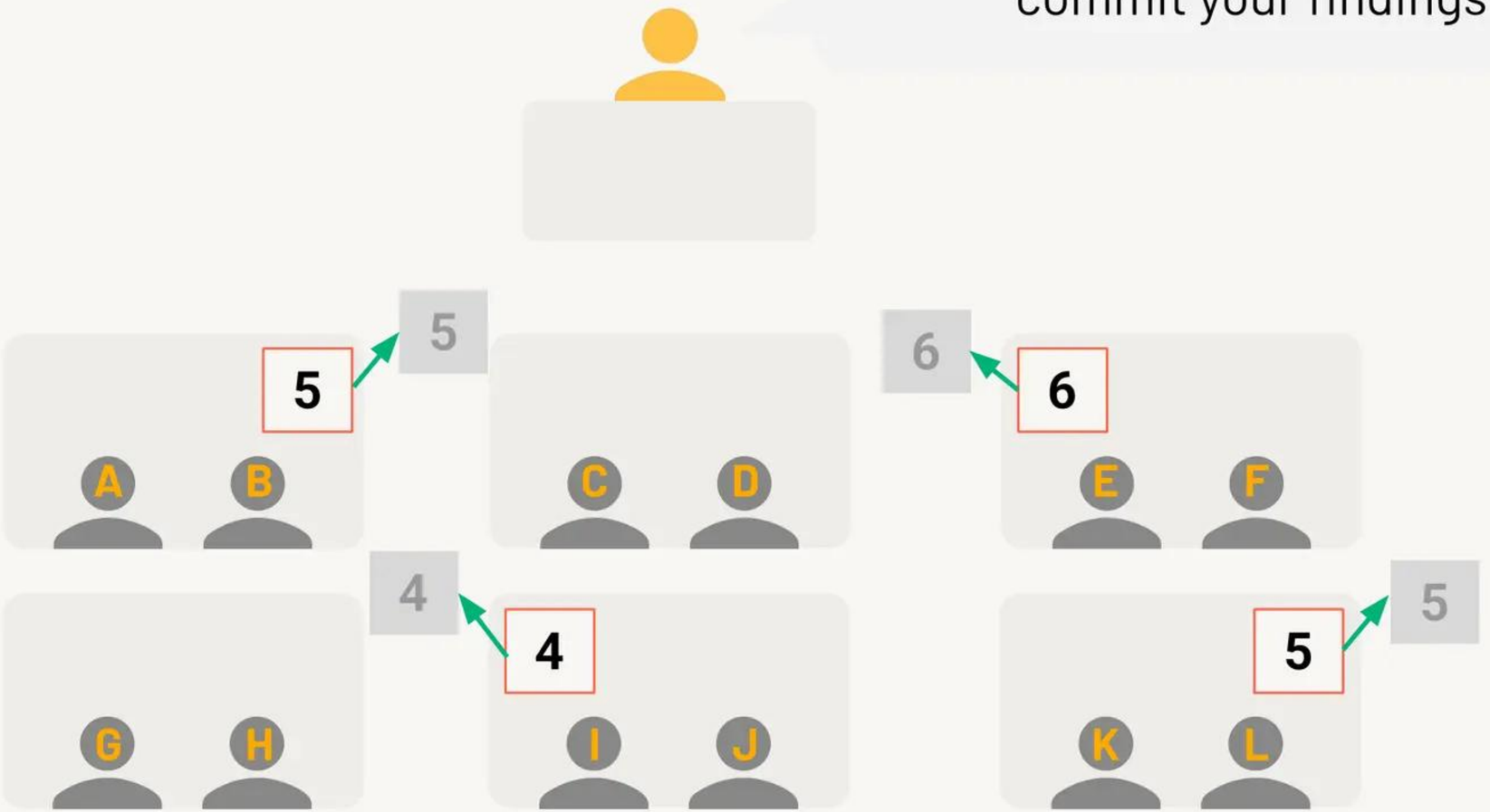
Stage 1: Local Count



©2022 Databricks Inc. — All rights reserved

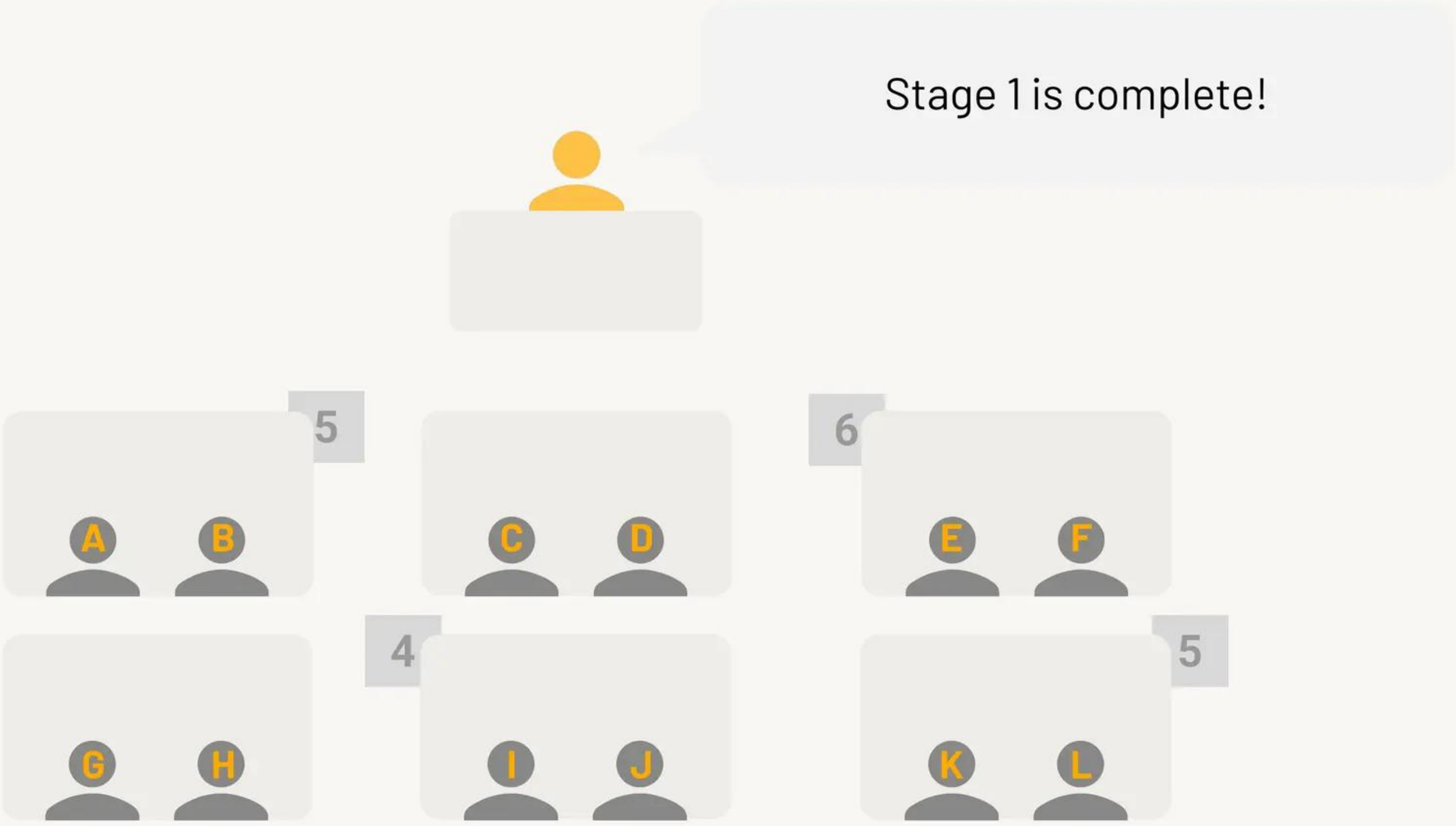
Stage 1: Local Count

Students B, E, I, L,
commit your findings



©2022 Databricks Inc. — All rights reserved

Stage 1: Local Count



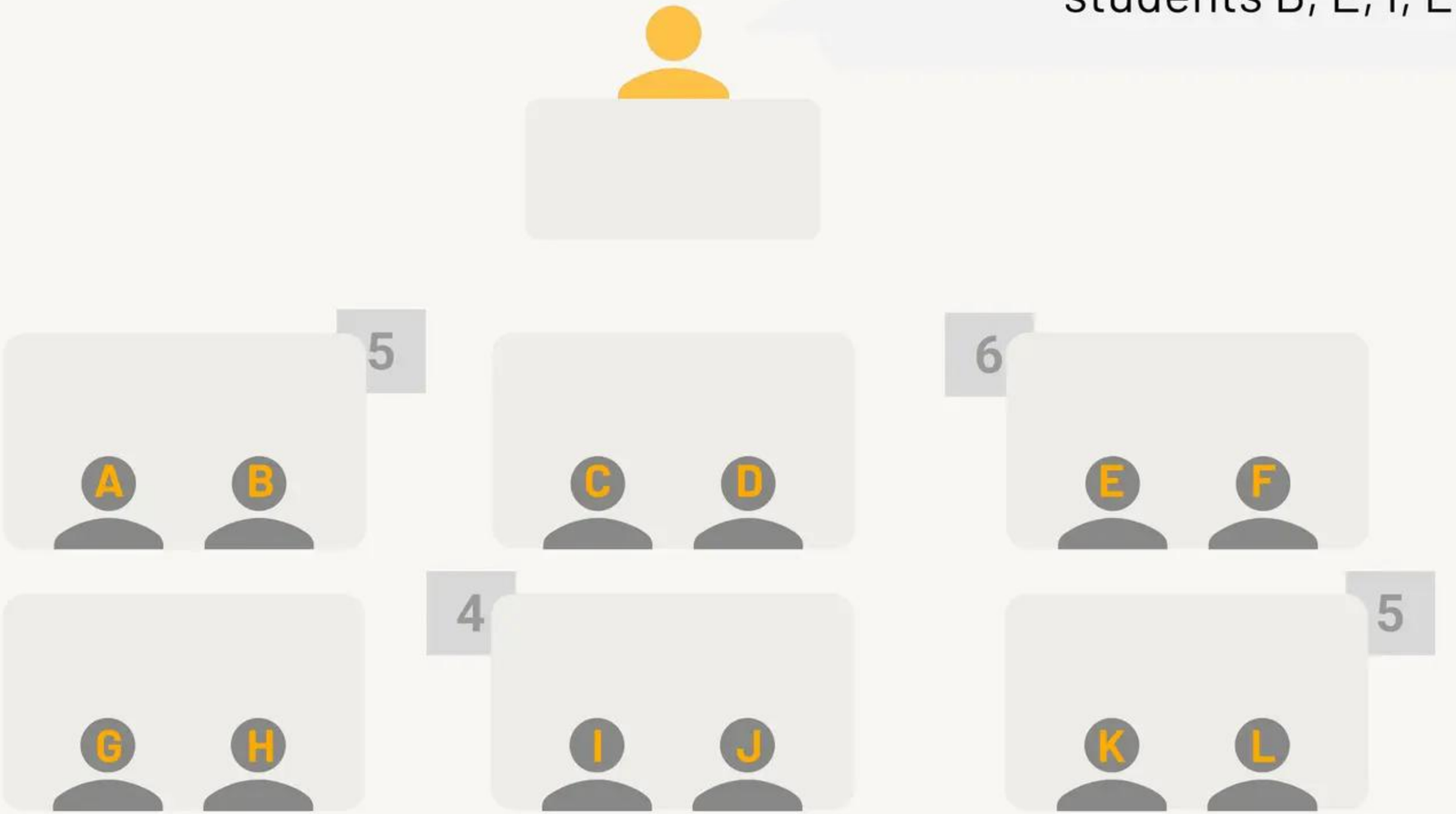
©2022 Databricks Inc. — All rights reserved

Stage 2: Global Count

©2022 Databricks Inc. — All rights reserved

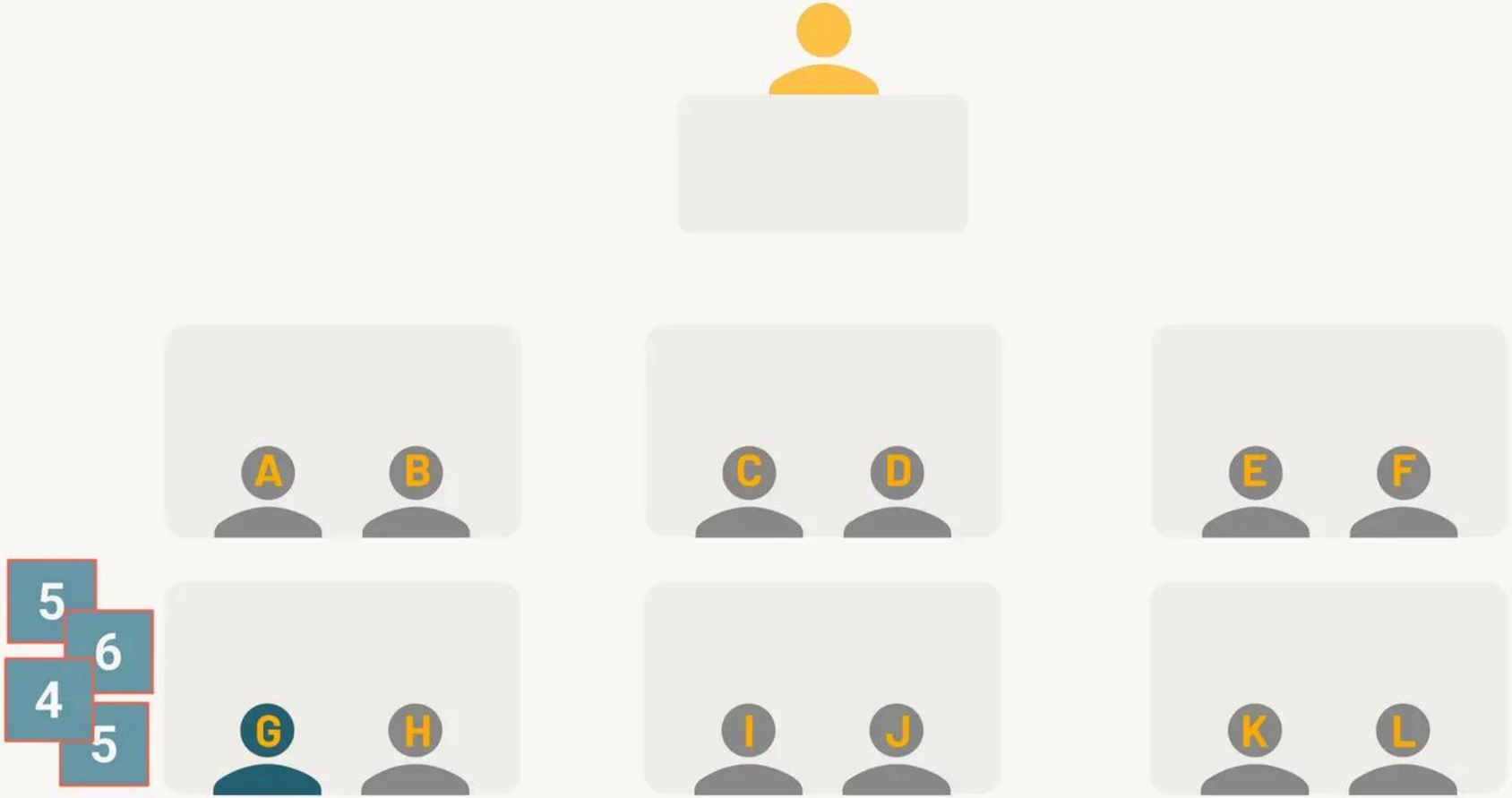
Stage 2: Global Count

Student G, fetch counts from students B, E, I, L



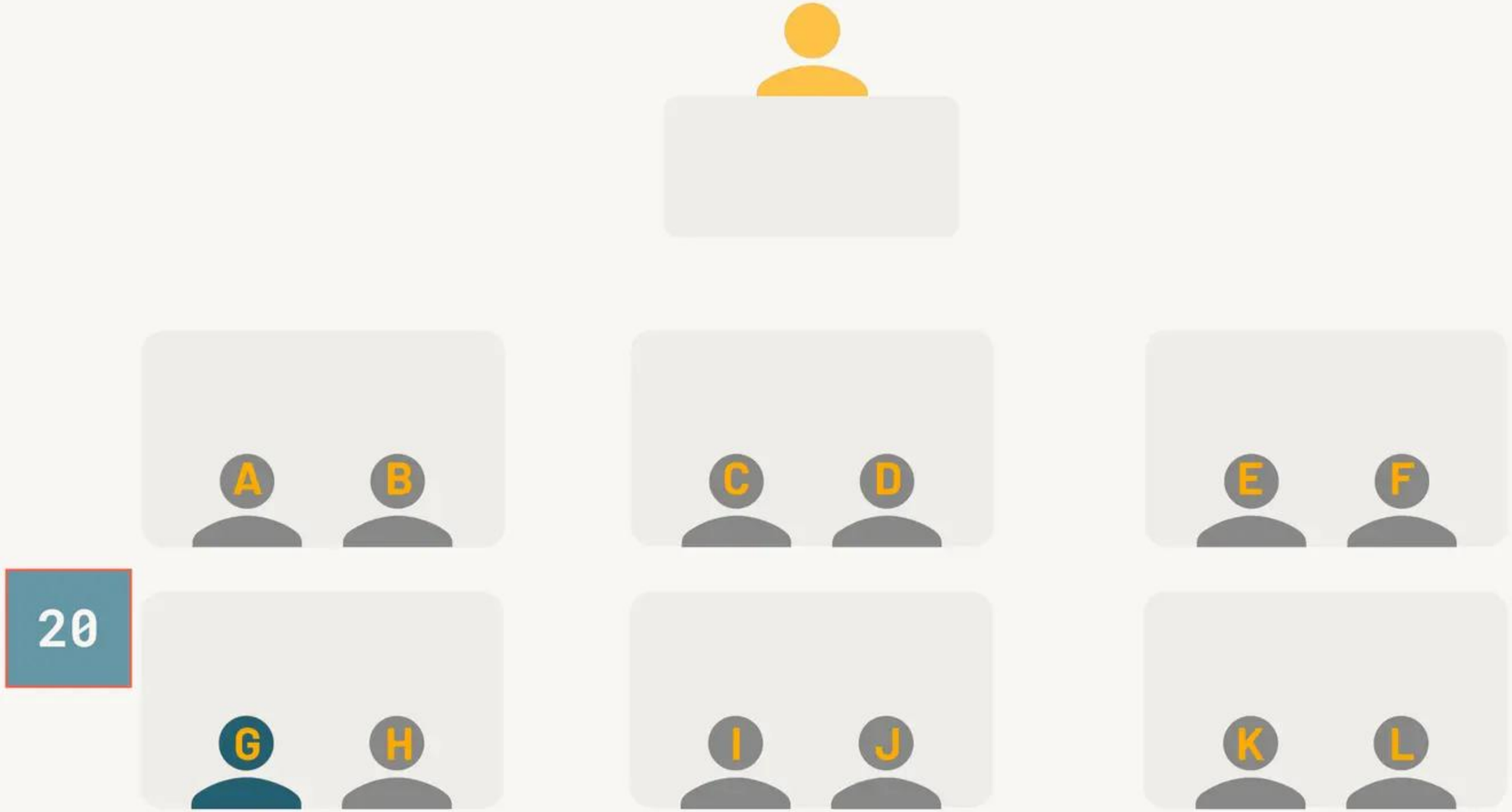
©2022 Databricks Inc. — All rights reserved

Stage 2: Global Count



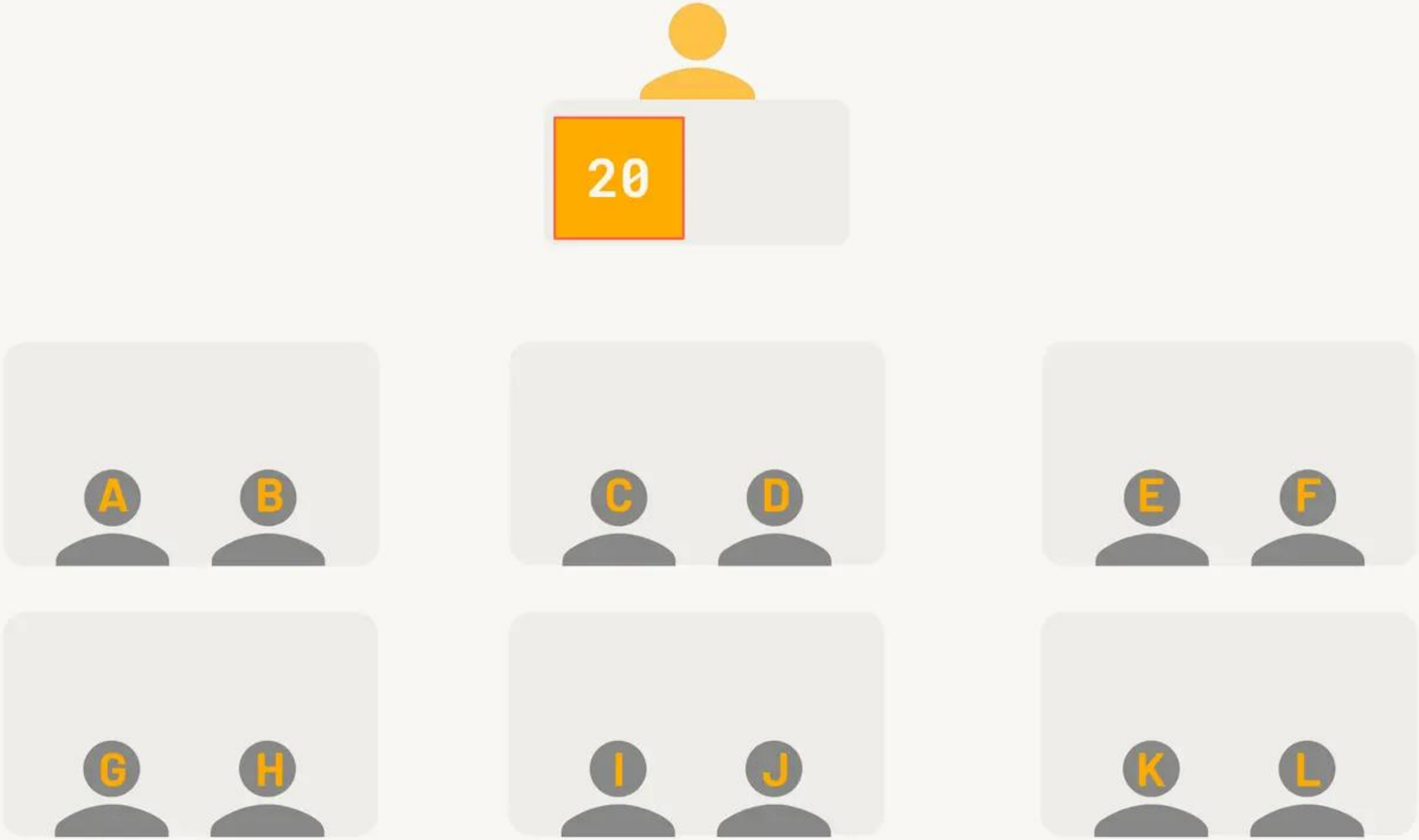
©2022 Databricks Inc. — All rights reserved

Stage 2: Global Count



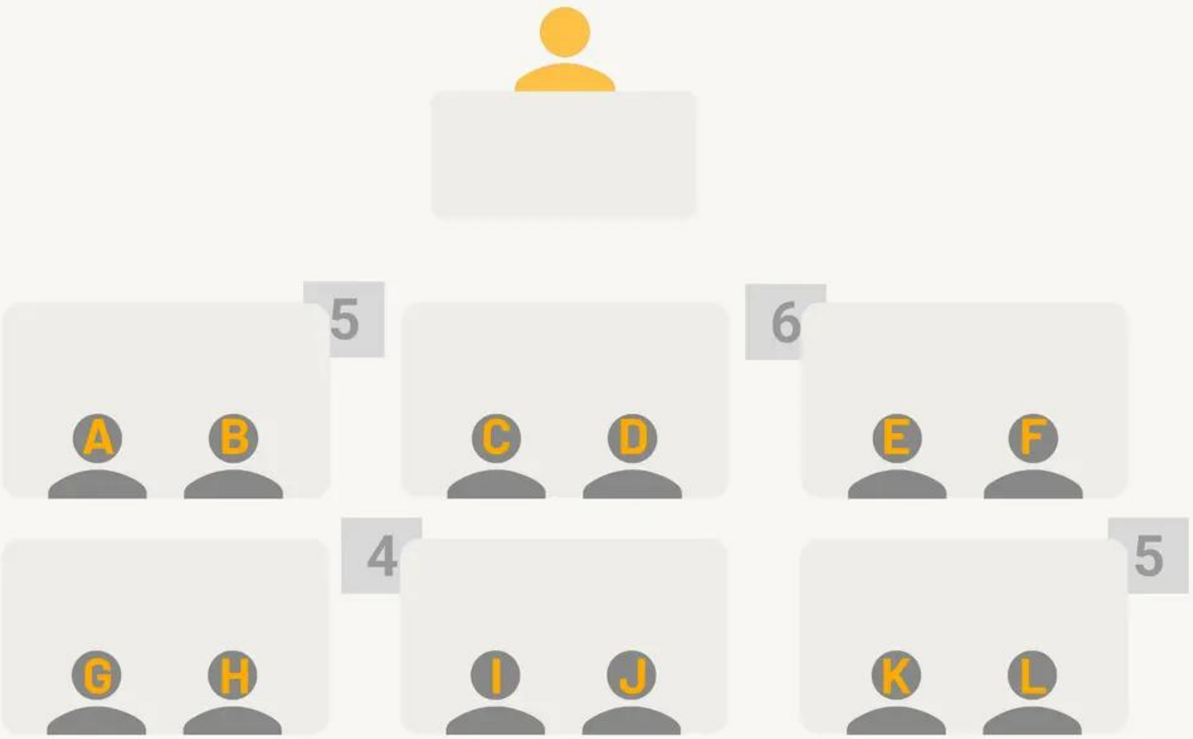
©2022 Databricks Inc. — All rights reserved

Stage 2: Global Count

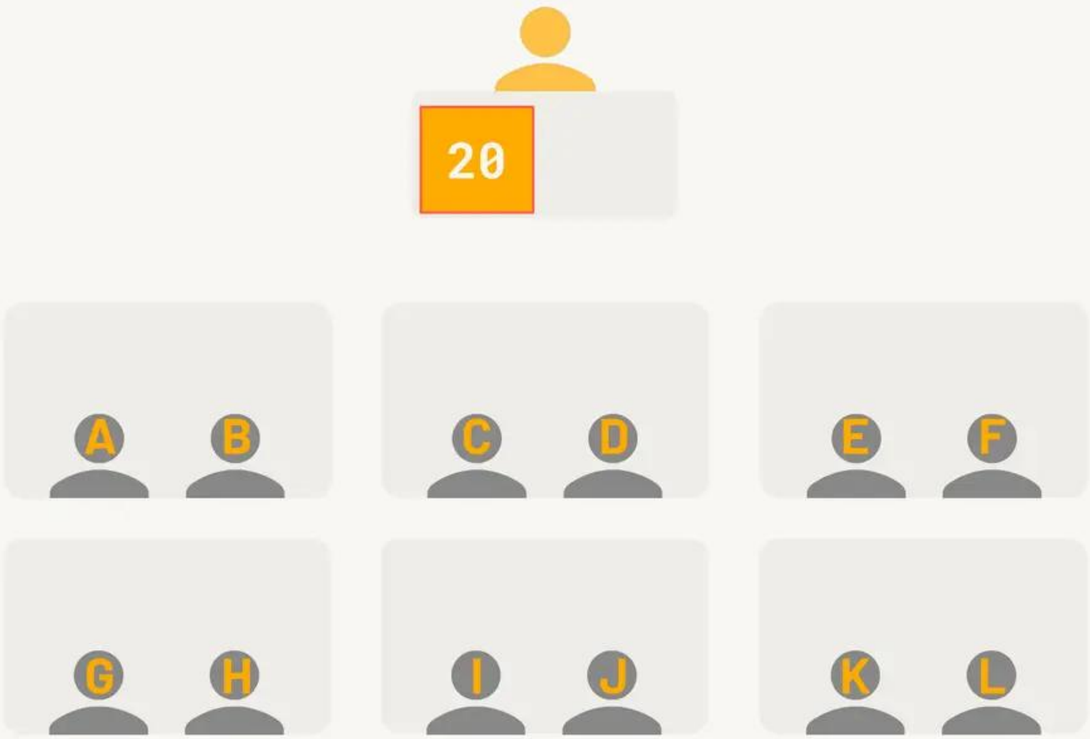


©2022 Databricks Inc. — All rights reserved

Stage 1: Local Count



Stage 2: Global Count



©2022 Databricks Inc. — All rights reserved

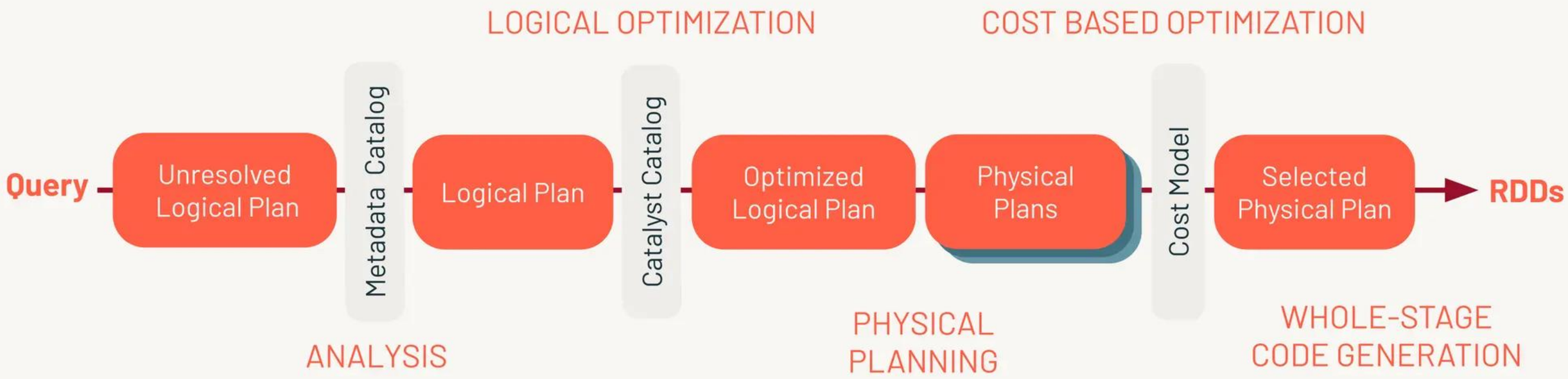
Query Optimization

Catalyst Optimizer

Adaptive Query Execution

©2022 Databricks Inc. — All rights reserved

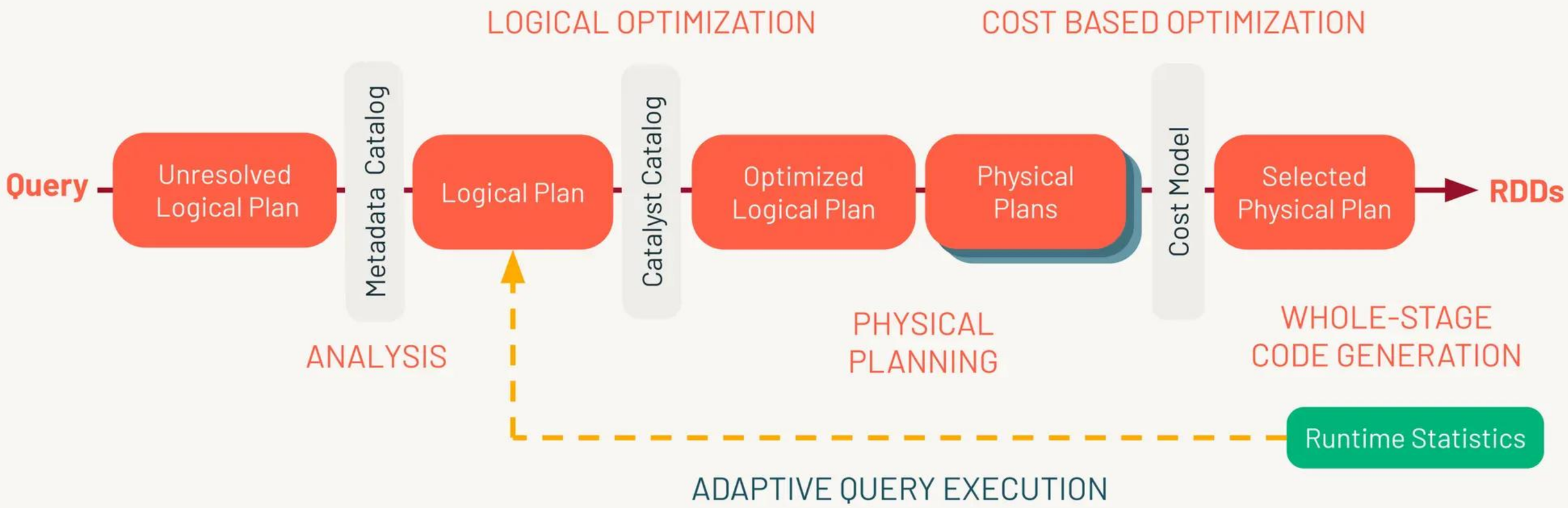
Query Optimization



©2022 Databricks Inc. — All rights reserved

Query Optimization with AQE

New in Spark 3.0, **enabled** by default as of Spark 3.2



©2022 Databricks Inc. — All rights reserved

Module 5

Structured Streaming

©2022 Databricks Inc. — All rights reserved

Structured Streaming

Streaming Query

Stream Aggregations

©2022 Databricks Inc. — All rights reserved

Streaming Query

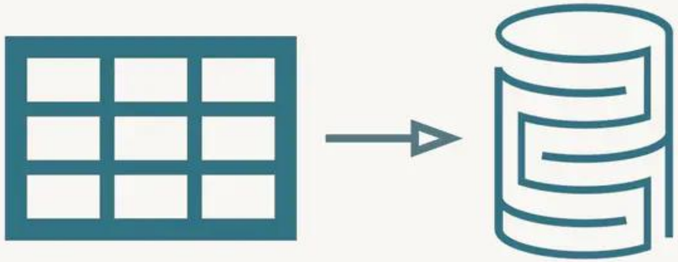
Advantages

Use Cases

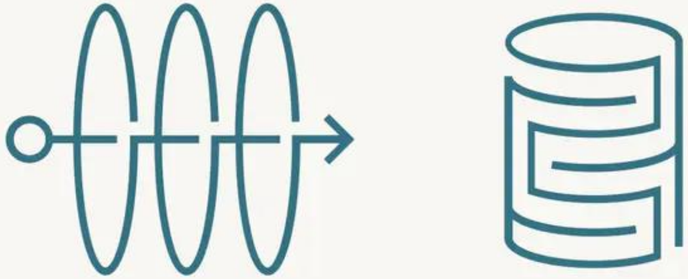
Sources

©2022 Databricks Inc. — All rights reserved

Batch Processing



Stream Processing



©2022 Databricks Inc. — All rights reserved

Advantages of Stream Processing

Lower latency

Efficient Updates

Automatic bookkeeping on new data

©2022 Databricks Inc. — All rights reserved

Stream Processing Use Cases

Notifications

Real-time reporting

Incremental ETL

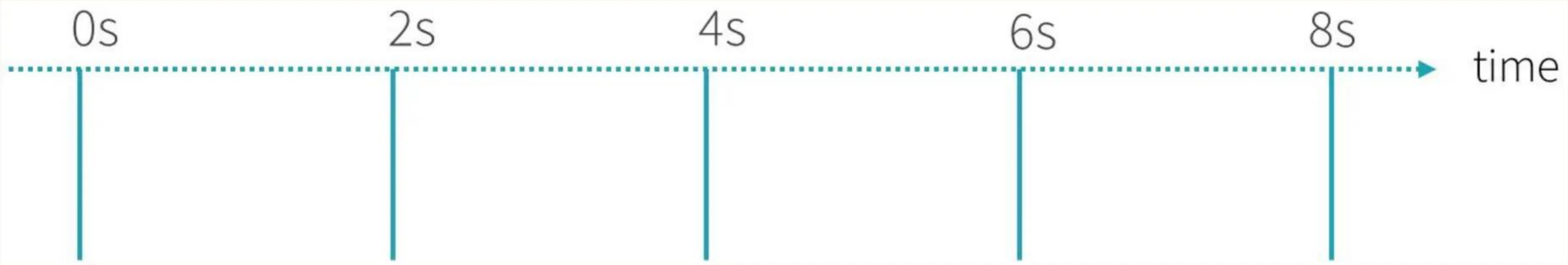
Update data to serve in real-time

Real-time decision making

Online ML

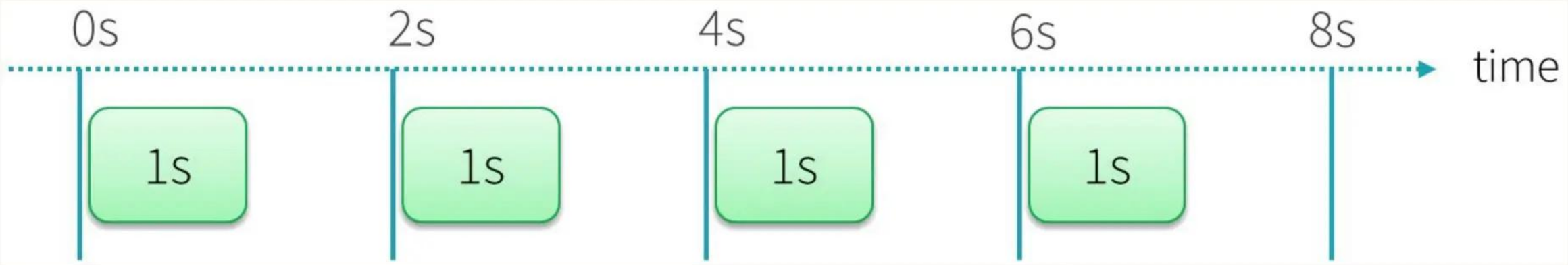
©2022 Databricks Inc. — All rights reserved

Micro-Batch Processing



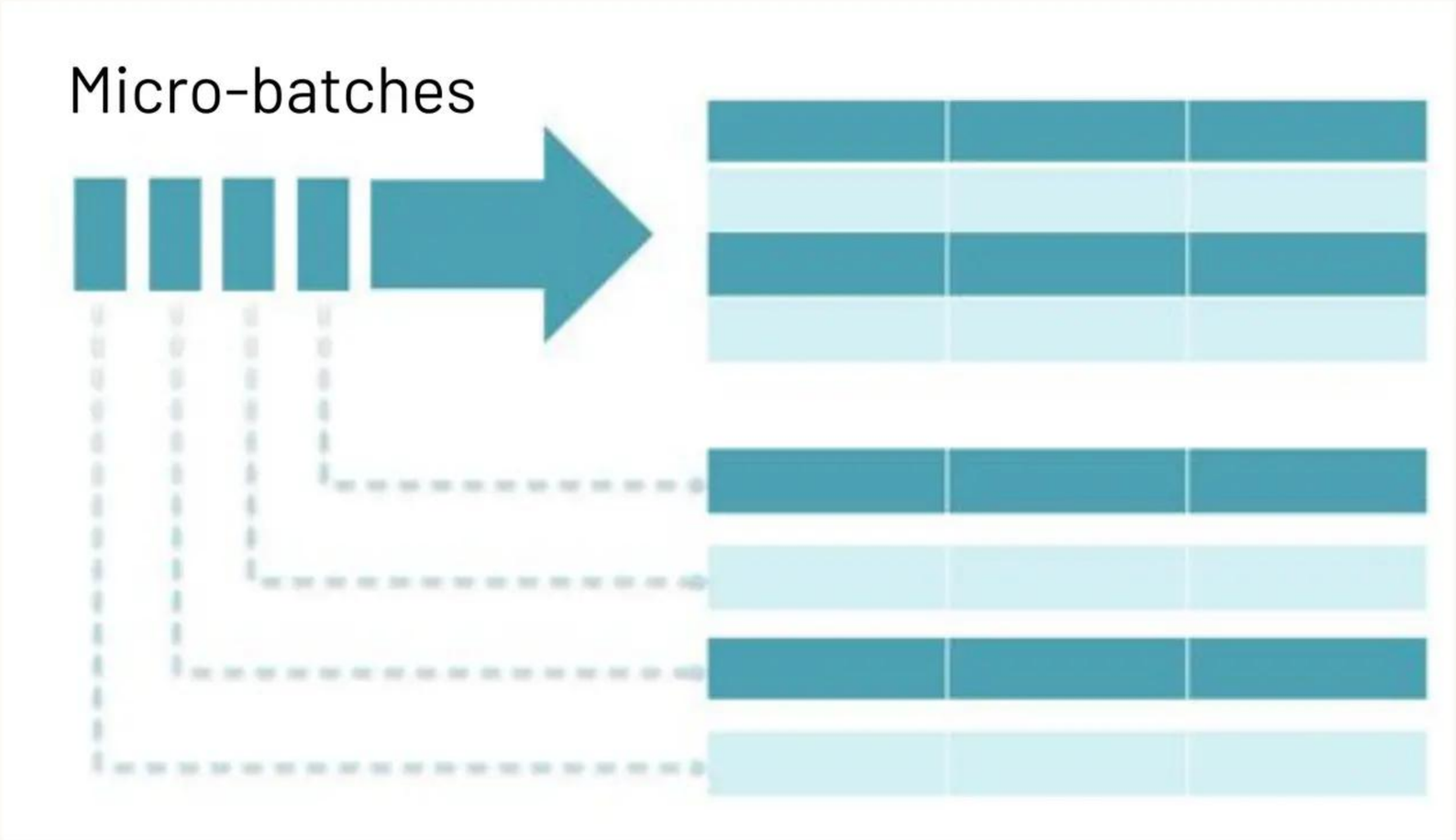
©2022 Databricks Inc. — All rights reserved

Micro-Batch Processing



©2022 Databricks Inc. — All rights reserved

Structured Streaming



= new rows appended to unbounded table

©2022 Databricks Inc. — All rights reserved

Input Sources

Kafka

Event Hubs

Files

Sockets

Generator

FOR TESTING

©2022 Databricks Inc. — All rights reserved

Sinks

Kafka

Event Hubs

Files

Foreach

Console

Memory

FOR DEBUGGING

©2022 Databricks Inc. — All rights reserved

Output Modes

APPEND

Add new records only

UPDATE

Update changed records in place

COMPLETE

Rewrite full output

©2022 Databricks Inc. — All rights reserved

Trigger Types

Default	Process each micro-batch as soon as the previous one has been processed
Fixed interval	Micro-batch processing kicked off at the user-specified interval
One-time	Process all of the available data as a single micro-batch and then automatically stop the query
Continuous Processing	Long-running tasks that continuously read, process, and write data as soon events are available **Experimental*

See Structured Streaming Programming Guide

©2022 Databricks Inc. — All rights reserved

End-to-end fault tolerance

Guaranteed in Structured Streaming by

Checkpointing and write-ahead logs

Idempotent sinks

Replayable data sources

©2022 Databricks Inc. — All rights reserved

Stream Aggregations

Aggregations

Windows

Watermarking

©2022 Databricks Inc. — All rights reserved

Real-time Aggregations

Errors in IoT data by device type

Anomalous behavior in server log files by country

Behavior analysis on messages by hashtags

©2022 Databricks Inc. — All rights reserved

Time-Based Windows

Tumbling Windows

No window overlap

Any given event gets aggregated into **only one** window group

e.g. 1:00–2:00 am, 2:00–3:00 am, 3:00-4:00 am, ...

Sliding Windows

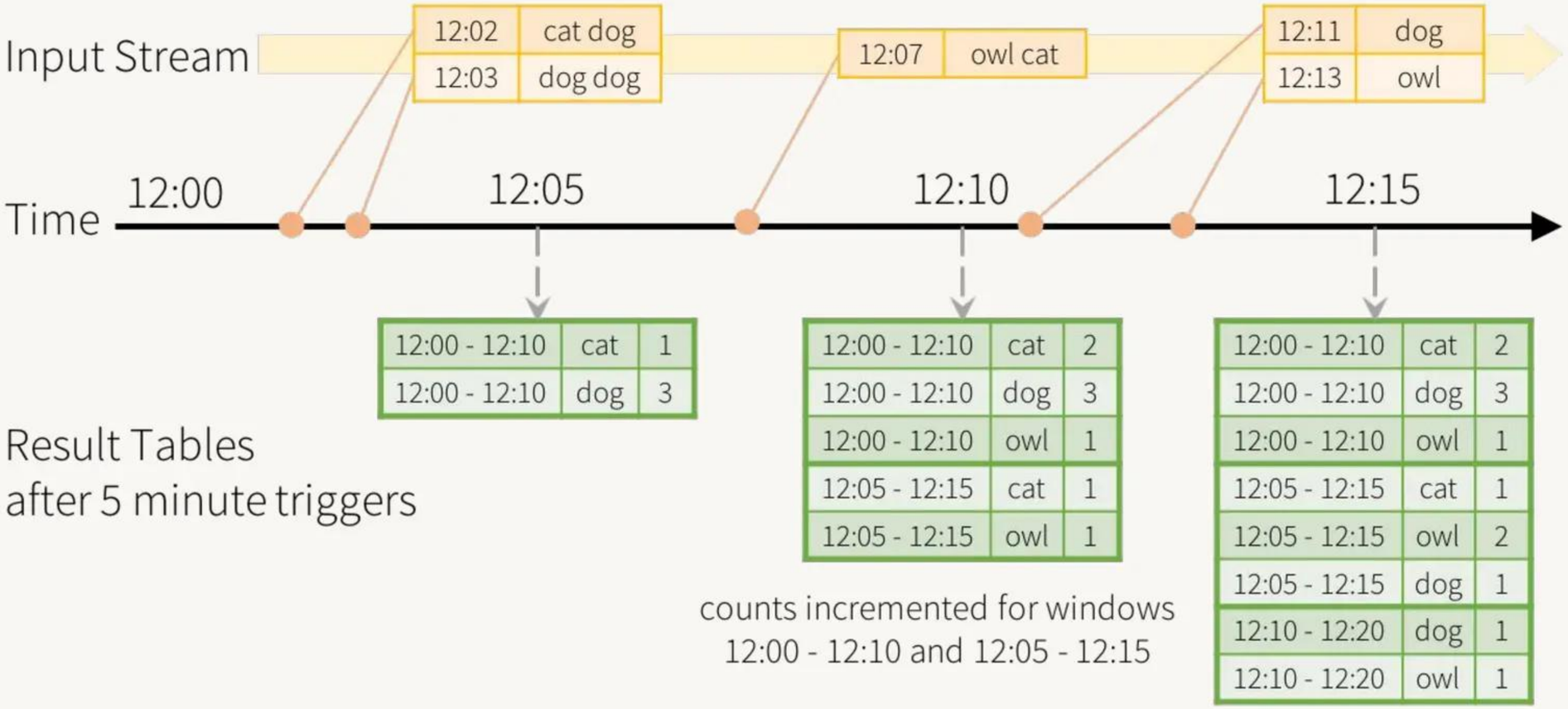
Windows overlap

Any given event gets aggregated into **multiple** window groups

e.g. 1:00-2:00 am, 1:30–2:30 am, 2:00–3:00 am, ...

©2022 Databricks Inc. — All rights reserved

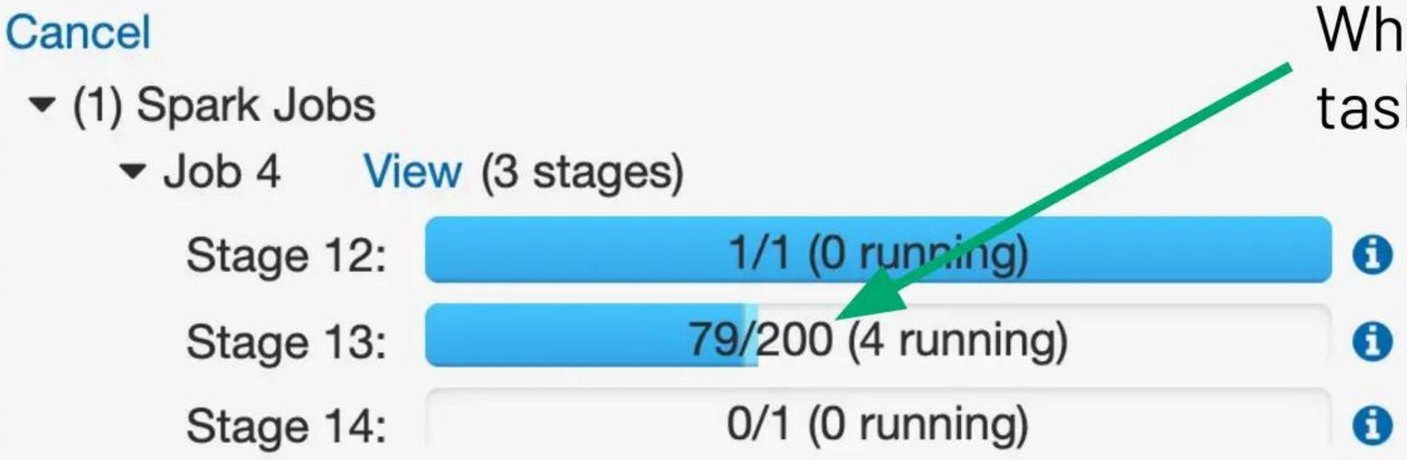
Sliding Windows Example



©2022 Databricks Inc. — All rights reserved

Windowing

```
(streamingDF
  .groupBy(col("device"),
           window(col("time"), "1 hour"))
  .count())
```



Why are we seeing 200 tasks for this stage?

©2022 Databricks Inc. — All rights reserved

Control the Shuffle Repartitioning

```
spark.conf.set("spark.sql.shuffle.partitions",spark.sparkContext.defaultParallelism)
```

©2022 Databricks Inc. — All rights reserved

Event-Time Processing

EVENT-TIME DATA

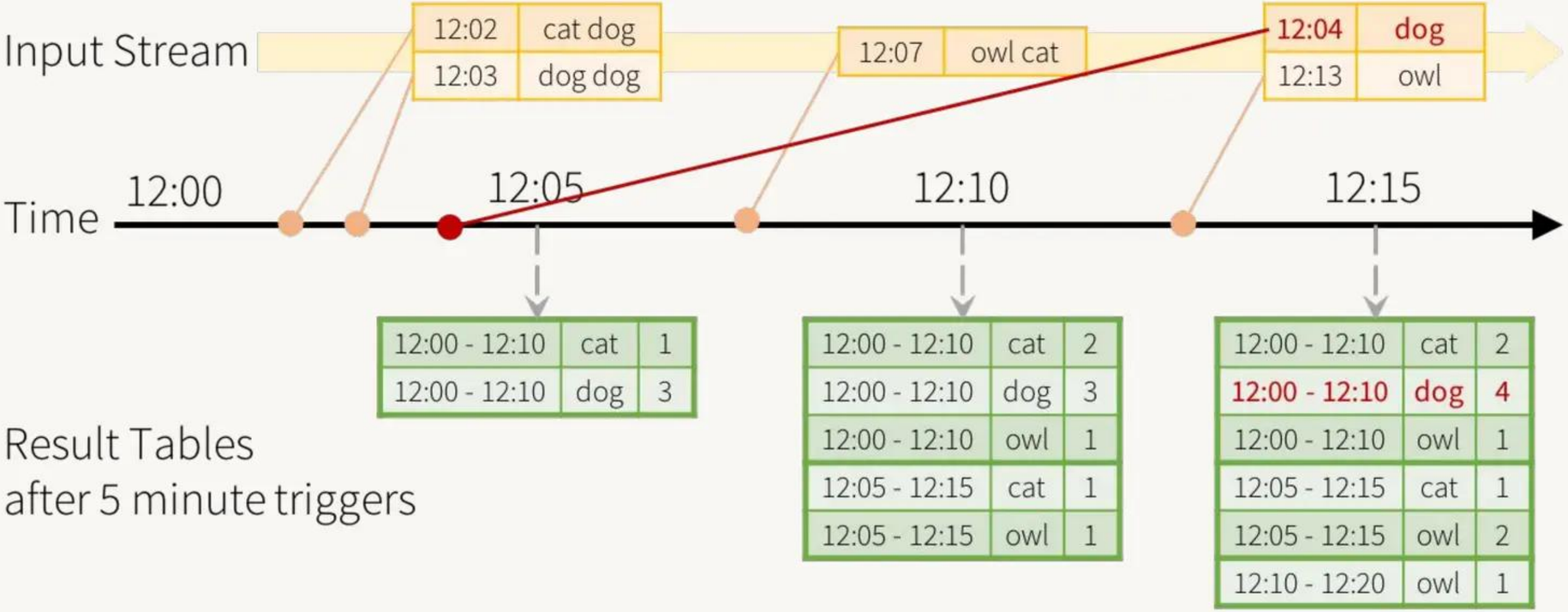
Process based on event-time (time fields embedded in data) rather than receipt time

WATERMARKS

Handle late data and limit how long to remember old data

©2022 Databricks Inc. — All rights reserved

Handling Late Data and Watermarking



©2022 Databricks Inc. — All rights reserved

Watermarking

```
(streamingDF
  .withWatermark("time", "2 hours")
  .groupBy(col("device"),
           window(col("time"), "1 hour"))
  .count()
)
```

©2022 Databricks Inc. — All rights reserved

Module 6

Delta Lake

©2022 Databricks Inc. — All rights reserved

Delta Lake

Using Spark with Delta Lake

©2022 Databricks Inc. — All rights reserved

Delta Lake

Delta Lake Concepts

©2022 Databricks Inc. — All rights reserved

What is Delta Lake?

- Technology designed to be used with Apache Spark to build robust data lakes
- Open source project at delta.io
- Databricks Delta Lake documentation

DELTA LAKE

©2022 Databricks Inc. — All rights reserved

Delta Lake features

- ACID transactions on Spark
- Scalable metadata handling
- Streaming and batch unification
- Schema enforcement
- Time travel
- Upserts and deletes
- Fully configurable/optimizable
- Structured streaming support

©2022 Databricks Inc. — All rights reserved

Delta Lake components

Delta Lake storage layer

Delta tables

Delta Engine

©2022 Databricks Inc. — All rights reserved

Delta Lake Storage Layer

- Highly performant and persistent
- Low-cost, easily scalable object storage
- Ensures consistency
- Allows for flexibility

©2022 Databricks Inc. — All rights reserved

Delta tables

- Contain data in Parquet files that are kept in object storage
- Keep transaction logs in object storage
- Can be registered in a metastore (optional)

©2022 Databricks Inc. — All rights reserved

Delta Engine

Databricks edge feature; not available in OS Apache Spark

- File management optimizations
- Auto-optimized writes
- Performance optimization via Delta caching

©2022 Databricks Inc. — All rights reserved

What is the Delta transaction log?

- Ordered record of the transactions performed on a Delta table
- Single source of truth for that table
- Mechanism that the Delta Engine uses to guarantee atomicity

©2022 Databricks Inc. — All rights reserved

How does the transaction log work?

- Delta Lake breaks operations down into one or more of these steps:
  - Add file
  - Remove file
  - Update metadata
  - Set transaction
  - Change protocol
  - Commit info

©2022 Databricks Inc. — All rights reserved

Delta transaction log at the file level

Transaction Log

Single Commits

(Optional) Partition Directories

Data Files

```
my_table/
  _delta_log/
    00000.json
    00001.json
  date=2019-01-01/
    file-1.parquet
```

©2022 Databricks Inc. — All rights reserved

Adding commits to the transaction log

©2022 Databricks Inc. — All rights reserved

`DESCRIBE` Command

- Returns the metadata of an existing table
  - Ex. Column names, data types, comments

`DESCRIBE HISTORY` Command

- Returns a more complete set of metadata for a Delta table
  - Operation, user, operation metrics

©2022 Databricks Inc. — All rights reserved

Delta Lake Time Travel

- Query an older snapshot of a Delta table
  - Re-creating analysis, reports or outputs
  - Writing complex temporal queries
  - Fixing mistakes in data
  - Providing snapshot isolation

©2022 Databricks Inc. — All rights reserved

Time Travel SQL syntax

```
SELECT * FROM events TIMESTAMP AS OF timestamp_expression
SELECT * FROM events VERSION AS OF version
```

©2022 Databricks Inc. — All rights reserved

Time Travel SQL syntax

```
SELECT * FROM events TIMESTAMP AS OF timestamp_expression
```

- `timestamp_expression` can be:
  - String that can be cast to a timestamp
  - String that can be cast to a date
  - Explicit timestamp type (the result of casting)
  - Simple time or date expressions (see the Databricks documentation)

©2022 Databricks Inc. — All rights reserved

Time Travel SQL syntax

```
SELECT * FROM events VERSION AS OF version
```

- `Version` can be obtained from the output of `DESCRIBE HISTORY` events.

©2022 Databricks Inc. — All rights reserved

Thank you! Congrats!

©2022 Databricks Inc. — All rights reserved