

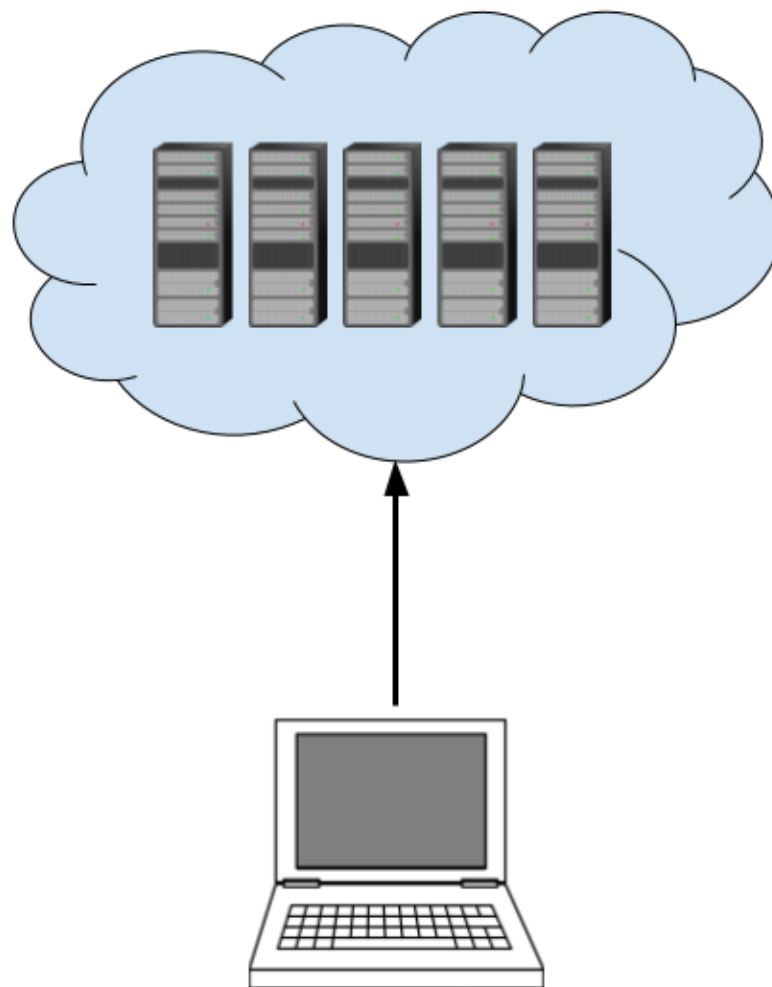
Getting started with Databricks

DATABRICKS CONCEPTS



Kevin Barlow
Data Practitioner

Compute cluster refresh



Create your first cluster

The first step is to create a cluster for your data processing!

Configuration options:

Sample Cluster

Policy 

Unrestricted

☒ Multi node ☐ Single node

Access mode 

Single user

Single user access 

kevin.curtis.barlow@gmail.com

Performance

Databricks runtime version 

Runtime: 12.2 LTS (Scala 2.12, Spark 3.3.2)

☐ Use Photon Acceleration 

Worker type 

Standard_DS3_v2

14 GB Memory, 4 Cores

Min workers

1

Max workers

1

☐ Spot instances

Driver type

Same as worker

14 GB Memory, 4 Cores

☒ Enable autoscaling 

☒ Terminate after 20 minutes of inactivity 

Tags

Add tags

Key

Value

Add

Create your first cluster

The first step is to create a cluster for your data processing!

Configuration options:

- Cluster policies and access

Sample Cluster

Policy

Unrestricted

☒ Multi node ☐ Single node

Access mode

Single user

Single user access

kevin.curtis.barlow@gmail.com

Performance

Databricks runtime version

Runtime: 12.2 LTS (Scala 2.12, Spark 3.3.2)

☐ Use Photon Acceleration 

Worker type

Standard_DS3_v2

14 GB Memory, 4 Cores

Min workers

1

Max workers

1

☐ Spot instances

Driver type

Same as worker

14 GB Memory, 4 Cores

☒ Enable autoscaling 

☒ Terminate after 20 minutes of inactivity 

Tags

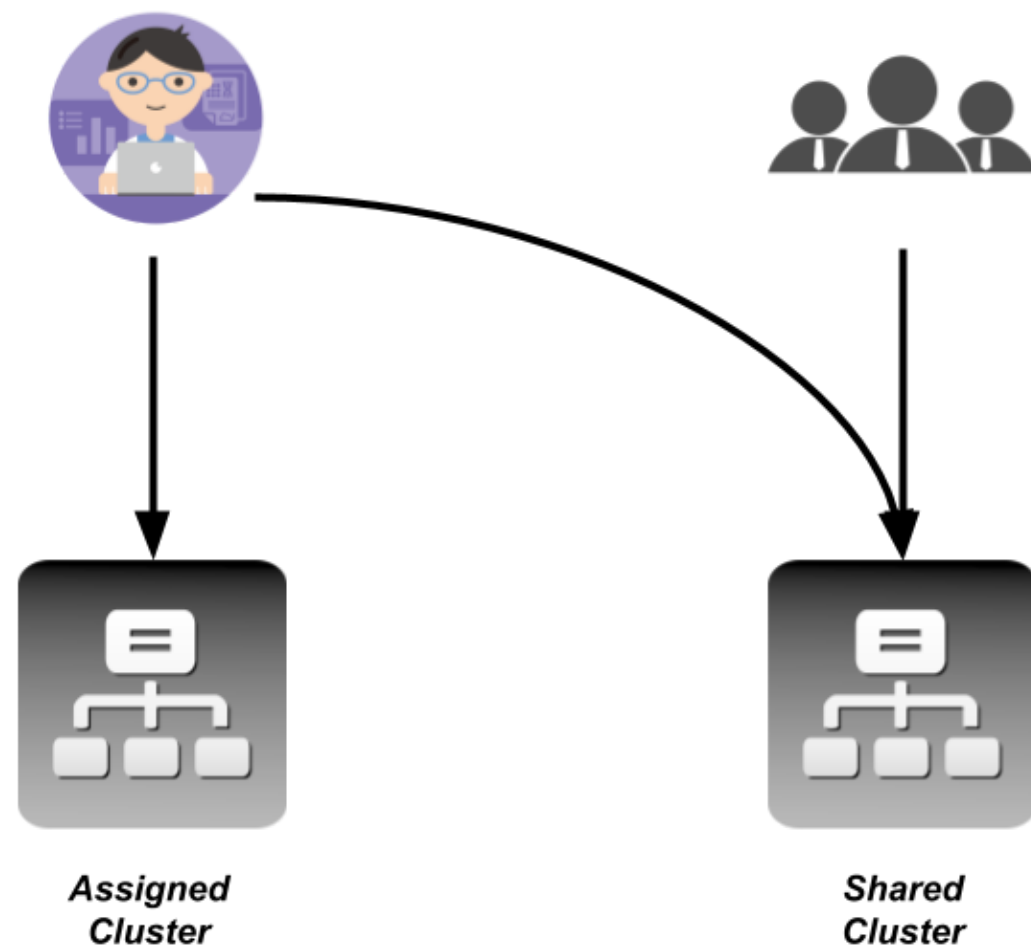
Add tags

Key

Value

Add

Cluster Access



Create your first cluster

The first step is to create a cluster for your data processing!

Configuration options:

- Cluster policies and access
- Databricks Runtime
- Photon Acceleration

Sample Cluster

Policy 

Unrestricted 

☒ Multi node ☐ Single node

Access mode 

Single user access 

Single user 

kevin.curtis.barlow@gmail.com 

Performance

Databricks runtime version 

Runtime: 12.2 LTS (Scala 2.12, Spark 3.3.2) 

☐ Use Photon Acceleration 

Worker type 

Standard_DS3_v2

14 GB Memory, 4 Cores 

Min workers

1

Max workers

1

☐ Spot instances

Driver type

Same as worker

14 GB Memory, 4 Cores 

☒ Enable autoscaling 

☒ Terminate after 20 minutes of inactivity 

Tags

Add tags

Key

Value

Add

Create your first cluster

The first step is to create a cluster for your data processing!

Configuration options:

- Cluster policies and access
- Databricks Runtime
- Photon Acceleration
- Node instance types and number
- Auto-scaling / Auto-termination

Sample Cluster

Policy 

Unrestricted 

☒ Multi node ☐ Single node

Access mode 

Single user access 

Single user 

kevin.curtis.barlow@gmail.com 

Performance

Databricks runtime version 

Runtime: 12.2 LTS (Scala 2.12, Spark 3.3.2) 

☐ Use Photon Acceleration 

Worker type 

Standard_DS3_v2

14 GB Memory, 4 Cores 

Min workers

1

Max workers

1

☐ Spot instances

Driver type

Same as worker

14 GB Memory, 4 Cores 

☒ Enable autoscaling 

☒ Terminate after 20 minutes of inactivity 

Tags

Add tags

Key

Value

Add

Data Explorer

Get familiar with the Data Explorer! In this UI, you can:

1. Browse available catalogs/schemas/tables
2. Look at sample data and summary statistics
3. View data lineage and history

You can also upload new data by clicking the "plus" icon!

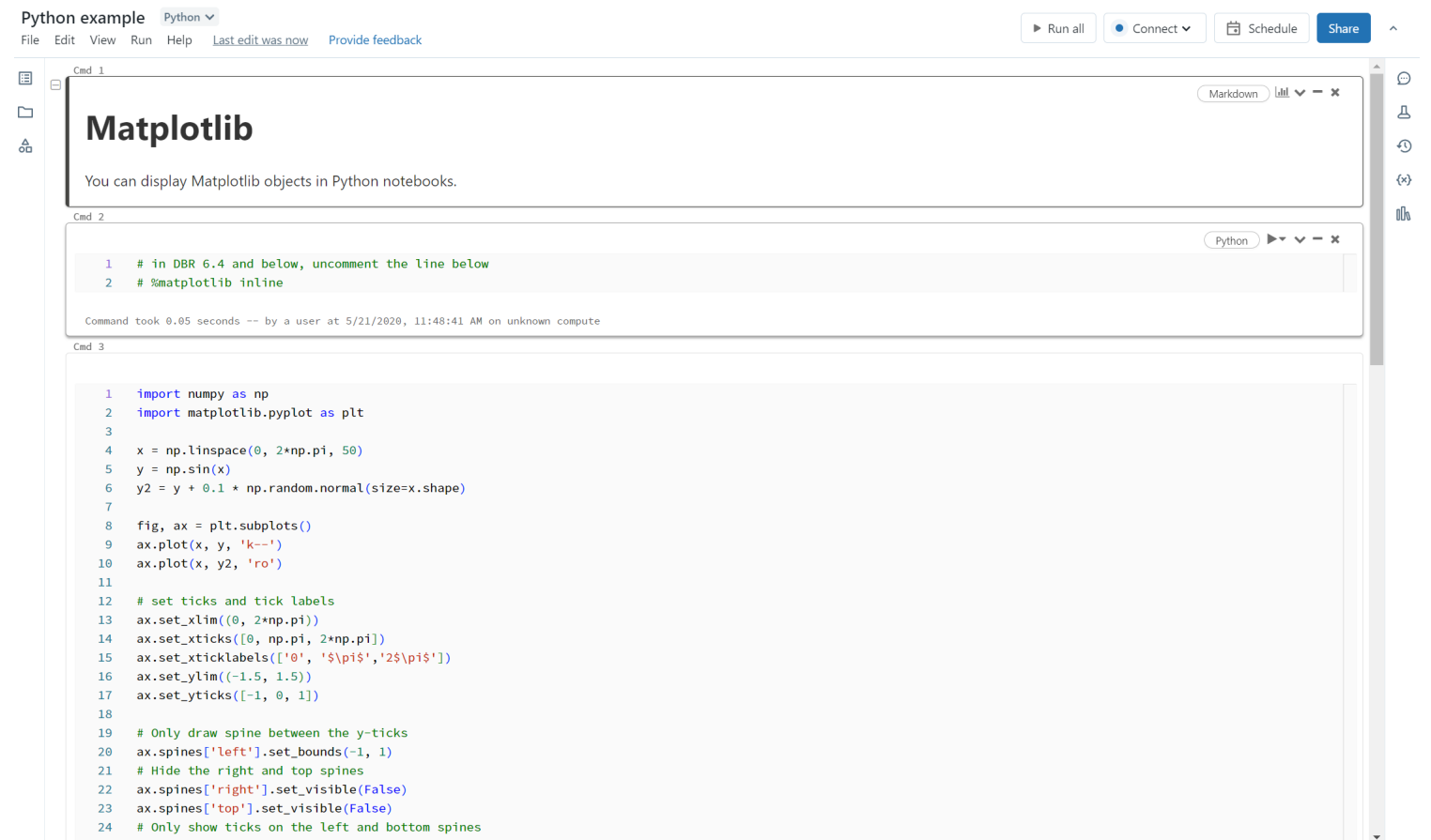


¹ Photo by Jakub Zerdzicki: <https://www.pexels.com/photo/magnifier-loupe-17284804/>

Create a notebook

Databricks notebooks:

- Standard interface for Databricks
- Improvements on open-source Jupyter
- Support for many languages
 - Python, R, Scala, SQL
 - Magic commands (%sql)
- Built-in visualizations
- Real-time commenting and collaboration



The screenshot displays a Databricks notebook titled "Python example". The interface includes a top bar with "File", "Edit", "View", "Run", and "Help" menus, along with buttons for "Run all", "Connect", "Schedule", and "Share". The notebook content is organized into cells. The first cell, labeled "Cmd 1", is a Markdown cell titled "Matplotlib" with the text "You can display Matplotlib objects in Python notebooks." The second cell, labeled "Cmd 2", is a Python code cell containing two lines of code: `1 # in DBR 6.4 and below, uncomment the line below` and `2 %matplotlib inline`. Below the code, a status bar indicates "Command took 0.05 seconds -- by a user at 5/21/2020, 11:48:41 AM on unknown compute". The third cell, labeled "Cmd 3", is a Python code cell containing a full Matplotlib script. The script imports numpy and matplotlib, generates random data, and creates a plot with two series: a dashed line and red circles. The plot is styled with a white background, black axes, and specific tick marks. The script concludes with `plt.show()`.

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 x = np.linspace(0, 2*np.pi, 50)
5 y = np.sin(x)
6 y2 = y + 0.1 * np.random.normal(size=x.shape)
7
8 fig, ax = plt.subplots()
9 ax.plot(x, y, 'k--')
10 ax.plot(x, y2, 'ro')
11
12 # set ticks and tick labels
13 ax.set_xlim((0, 2*np.pi))
14 ax.set_xticks([0, np.pi, 2*np.pi])
15 ax.set_xticklabels(['0', '$\pi$', '2$\pi$'])
16 ax.set_ylim((-1.5, 1.5))
17 ax.set_yticks([-1, 0, 1])
18
19 # Only draw spine between the y-ticks
20 ax.spines['left'].set_bounds(-1, 1)
21 # Hide the right and top spines
22 ax.spines['right'].set_visible(False)
23 ax.spines['top'].set_visible(False)
24 # Only show ticks on the left and bottom spines
```

Let's practice!
DATABRICKS CONCEPTS

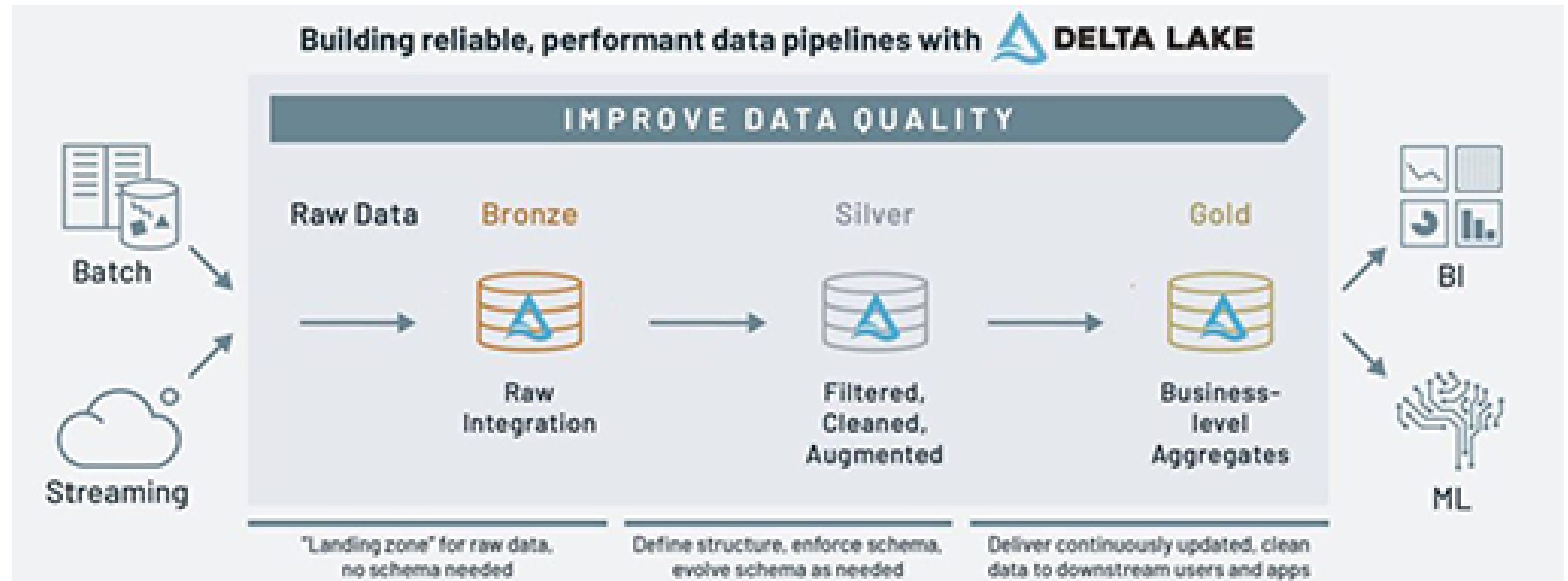
Data Engineering foundations in Databricks

DATABRICKS CONCEPTS



Kevin Barlow
Data Practitioner

Medallion architecture



Reading data

Spark is a highly flexible framework and can read from various data sources/types.

Common data sources and types:

- Delta tables
- File formats (CSV, JSON, Parquet, XML)
- Databases (MySQL, Postgres, EDW)
- Streaming data
- Images / Videos



Reading data

Spark is a highly flexible framework and can read from various data sources/types.

Common data sources and types:

- Delta tables
- File formats (CSV, JSON, Parquet, XML)
- Databases (MySQL, Postgres, EDW)
- Streaming data
- Images / Videos

```
#Delta table
spark.read.table()

#CSV files
spark.read.format('csv').load('*.*csv')

#Postgres table
spark.read.format("jdbc")
    .option("driver", driver)
    .option("url", url)
    .option("dbtable", table)
    .option("user", user)
    .option("password", password)
    .load()
```

Structure of a Delta table

A Delta table provides table-like qualities to an open file format.

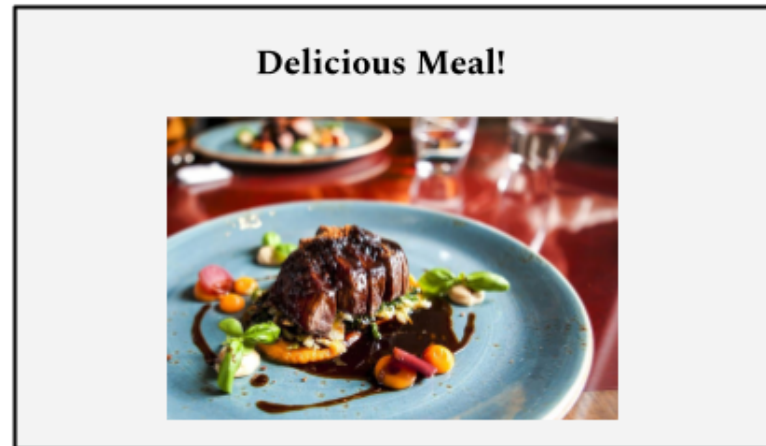
- Feels like a table when reading
- Access to underlying files (Parquet and JSON)

```
1 %fs ls /databricks-datasets/nyctaxi-with-zipcodes/subsampled
```

Table ▾ +

	path	name
1	dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled/_delta_log/	_delta_log/
2	dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled/nyc-zips-dataset-readme.txt	nyc-zips-dataset-readme.txt
3	dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled/part-00000-80b68cae-ce6a-41cf-87cd-2573d91b4c07-c000.snappy.parquet	part-00000-80b68cae-ce6a-41cf-87cd-2573d91b4c07-c000.snappy.parquet
4	dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled/part-00001-c883942d-366f-478a-be3b-f13fd4bee0ab-c000.snappy.parquet	part-00001-c883942d-366f-478a-be3b-f13fd4bee0ab-c000.snappy.parquet
5	dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled/part-00002-bbf9fd81-4b3a-46f3-943e-841b48ae743e-c000.snappy.parquet	part-00002-bbf9fd81-4b3a-46f3-943e-841b48ae743e-c000.snappy.parquet
6	dbfs:/databricks-datasets/nyctaxi-with-zipcodes/subsampled/part-00003-3d80435e-15f8-4154-92c7-515307e41c1b-c000.snappy.parquet	part-00003-3d80435e-15f8-4154-92c7-515307e41c1b-c000.snappy.parquet

Explaining the Delta Lake structure



Ingredients	Steps
• Ribeye Steak • Carrots • Red Wine •	1. Preheat oven 2. Mise en place 3. Sear meat 4.

DataFrames

DataFrames are two-dimensional representations of data.

- Look and feel similar to tables
- Similar concept for many different data tools
 - Spark (default), pandas, dplyr, SQL queries
- Underlying construct for most data processes

id	customerName	bookTitle
1	John Data	Guide to Spark
2	Sally Bricks	SQL for Data Engineering
3	Adam Delta	Keeping Data Clean

```
df = (spark.read
      .format("csv")
      .option("header", "true")
      .option("inferSchema", "true")
      .load("/data.csv"))
```

Writing data

Kinds of tables in Databricks

1. Managed tables
 - Default type
 - Stored with Unity Catalog
 - Databricks managed
2. External tables
 - Stored in another location
 - Set LOCATION
 - Customer managed

```
df.write.saveAsTable(table_name)
```

```
CREATE TABLE table_name  
USING delta  
AS ...
```

```
df.write  
  .location(' ').saveAsTable(table_name)
```

```
CREATE TABLE table_name  
USING delta  
LOCATION "<path>"  
AS ...
```

Let's practice!
DATABRICKS CONCEPTS

Data transformations in Databricks

DATABRICKS CONCEPTS



Kevin Barlow
Data Practitioner

SQL for data engineering

SQL

- Familiar for Database Administrators (DBAs)
- Great for standard manipulations
- Execute pre-defined UDFs

```
-- Creating a new table in SQL
```

```
CREATE TABLE table_name  
USING delta  
AS (  
    SELECT *  
    FROM source_table  
    WHERE date >= '2023-01-01'  
)
```

Other languages for data engineering

Python, R, Scala

- Familiar for software engineers
- Standard and complex transformations
- Use and define custom functions

#Creating a new table in Pyspark

```
spark
  .read
  .table('source_table')
  .filter(col('date') >= '2023-01-01')
  .write
  .saveAsTable('table_name')
```

Common transformations

Schema manipulation

- Add and remove columns
- Redefine columns

#Pyspark

```
df
    .withColumn(col('newCol'), ...)
    .drop(col('oldCol'))
```

Filtering

- Reduce DataFrame to subset of data
- Pass multiple criteria

#Pyspark

```
df
    .filter(col('date') >= target_date)
    .filter(col('id') IS NOT NULL)
```

Common transformations (continued)

Nested data

- Arrays or Struct data
- Expand or contract

```
df
    .explode(col('arrayCol')) #wide to long
    .flatten(col('items')) #long to wide
```

Aggregation

- Group data based on columns
- Calculate data summarizations

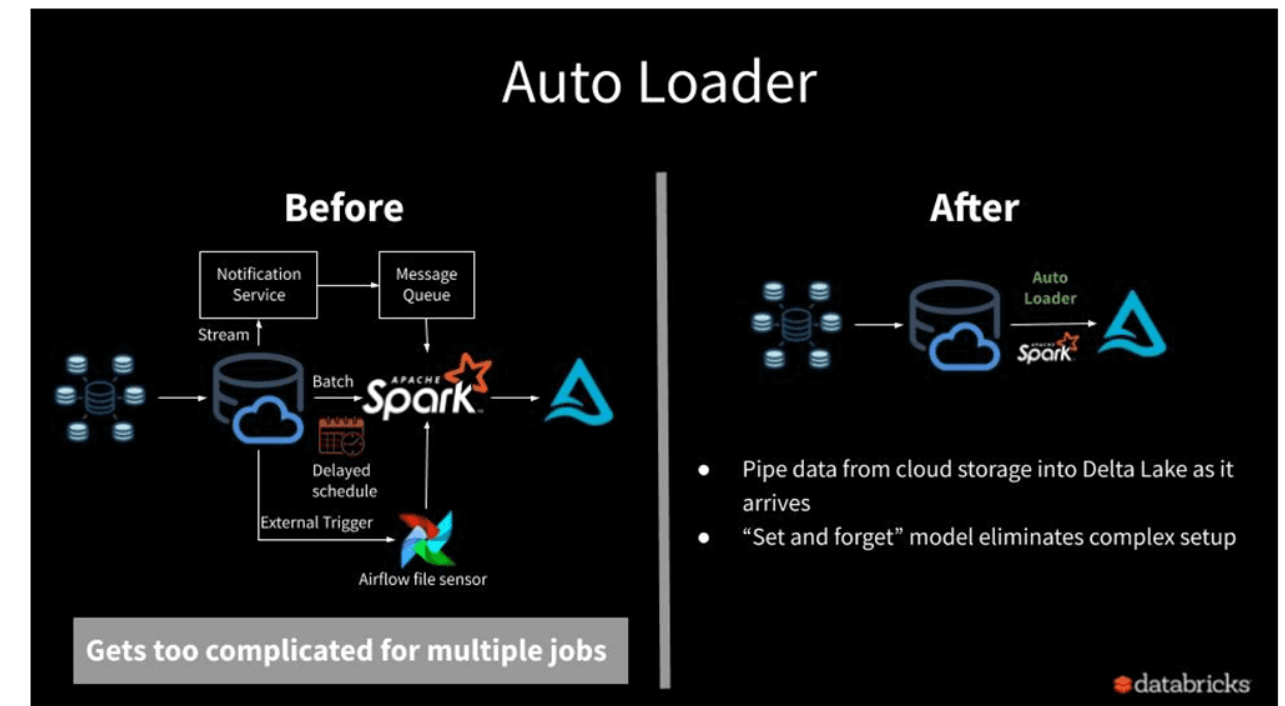
```
df
    .groupBy(col('region'))
    .agg(sum(col('sales')))
```

Auto Loader

Auto Loader processes new data files as they land in a data lake.

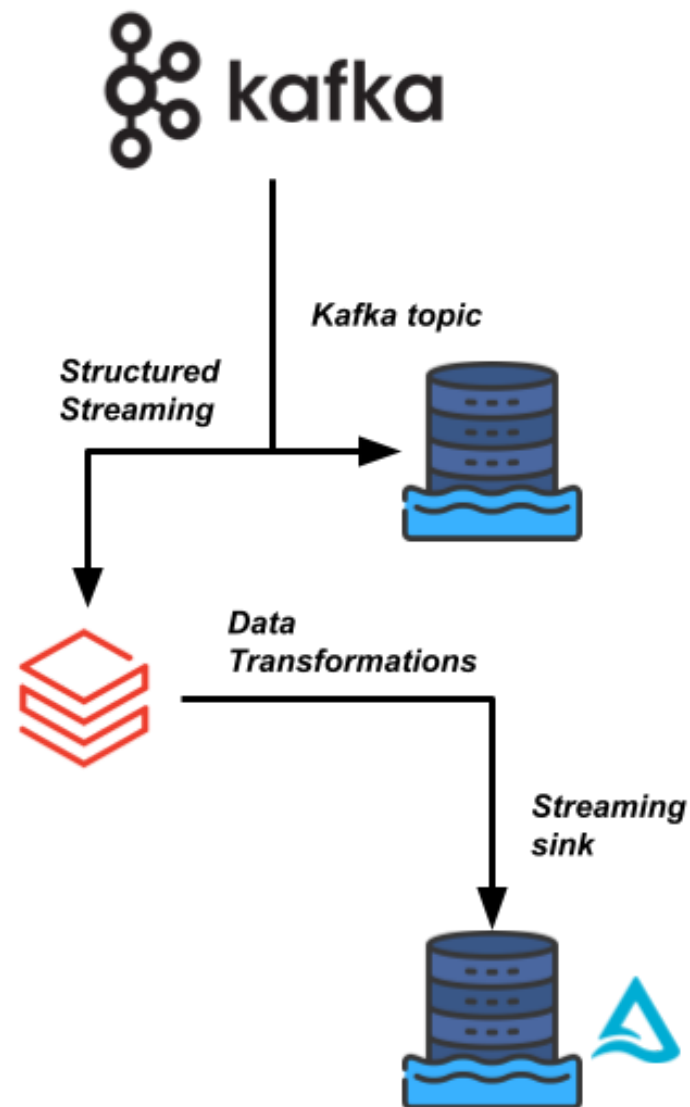
- Incremental processing
- Efficient processing
- Automatic

```
spark.readStream  
  .format("cloudFiles")  
  .option("cloudFiles.format", "json")  
  .load(file_path)
```



¹ <https://www.databricks.com/blog/2020/02/24/introducing-databricks-ingest-easy-data-ingestion-into-delta-lake.html>

Structured Streaming



```
spark.readStream
  .format("kafka")
  .option("subscribe", "<topic>")
  .load()
  .join(table_df,
        on="<id>", how="left")
  .writeStream
  .format("kafka")
  .option("topic", "<topic>")
  .start()
```

Let's practice!

DATABRICKS CONCEPTS

Orchestration in Databricks

DATABRICKS CONCEPTS



Kevin Barlow

Data Analytics Practitioner

What is data orchestration?

- Data orchestration is a form of automation!



- Enables data engineers to automate the end-to-end data life cycle

Databricks Workflows

Databricks Workflows is a collection of built-in capabilities to orchestrate all your data processes, at no additional cost!

Example Databricks Workflow



¹ <https://docs.databricks.com/workflows>

What can we orchestrate?

Data engineers/data scientists

Data analysts

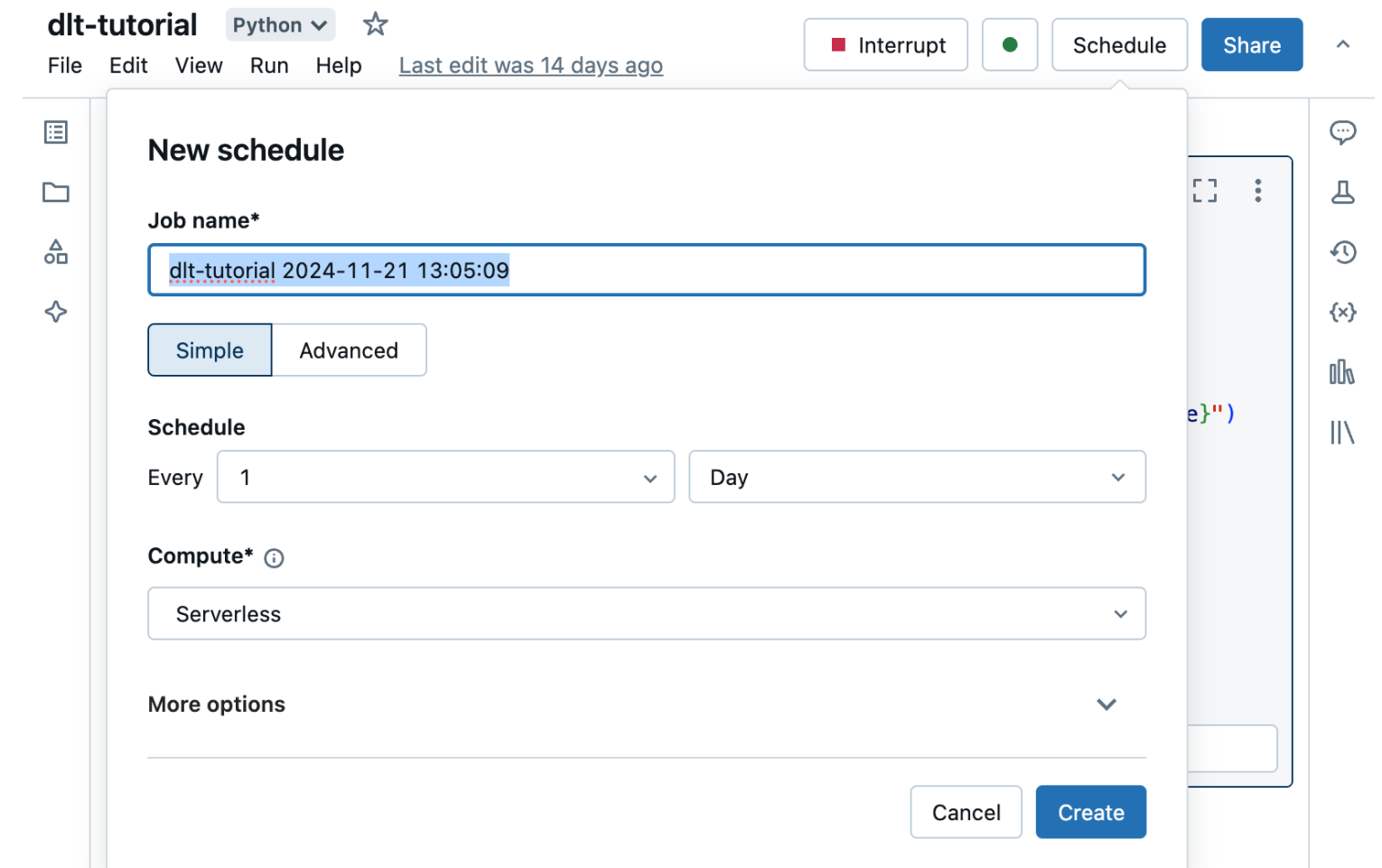


Databricks Jobs

Workflows UI

Users can create jobs directly from the Databricks UI:

- Directly from a notebook
- In the Workflows section



The screenshot shows the 'New schedule' dialog box in the Databricks UI. At the top, there's a header bar with 'dlt-tutorial', a 'Python' language selector, a star icon, and buttons for 'Interrupt', 'Schedule', and 'Share'. Below this is a menu bar with 'File', 'Edit', 'View', 'Run', and 'Help', along with a timestamp 'Last edit was 14 days ago'. The dialog box itself has a title 'New schedule'. It contains a 'Job name*' field with the text 'dlt-tutorial 2024-11-21 13:05:09'. Below the job name are two tabs: 'Simple' (selected) and 'Advanced'. The 'Schedule' section has 'Every' set to '1' and 'Day' selected from a dropdown. The 'Compute*' section has 'Serverless' selected from a dropdown. At the bottom, there's a 'More options' section with a downward arrow. Finally, there are 'Cancel' and 'Create' buttons at the bottom right.

¹ <https://docs.databricks.com/workflows/jobs>

Databricks Jobs

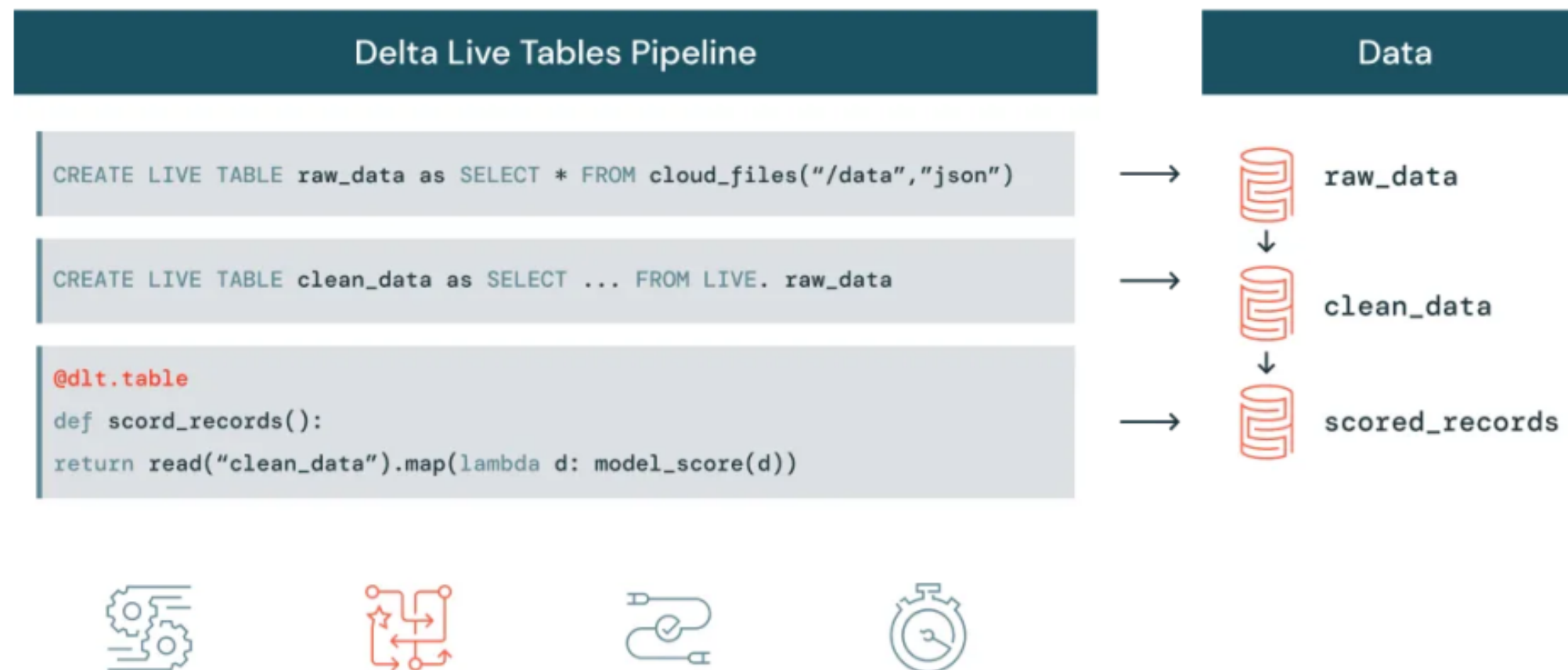
Programmatic

Users can also programmatically create jobs using the Jobs CLI or Jobs API with the Databricks platform.

```
{  
  "name": "A multitask job",  
  "tags": {},  
  "tasks": [],  
  "job_clusters": [],  
  "format": "MULTI_TASK",  
}
```

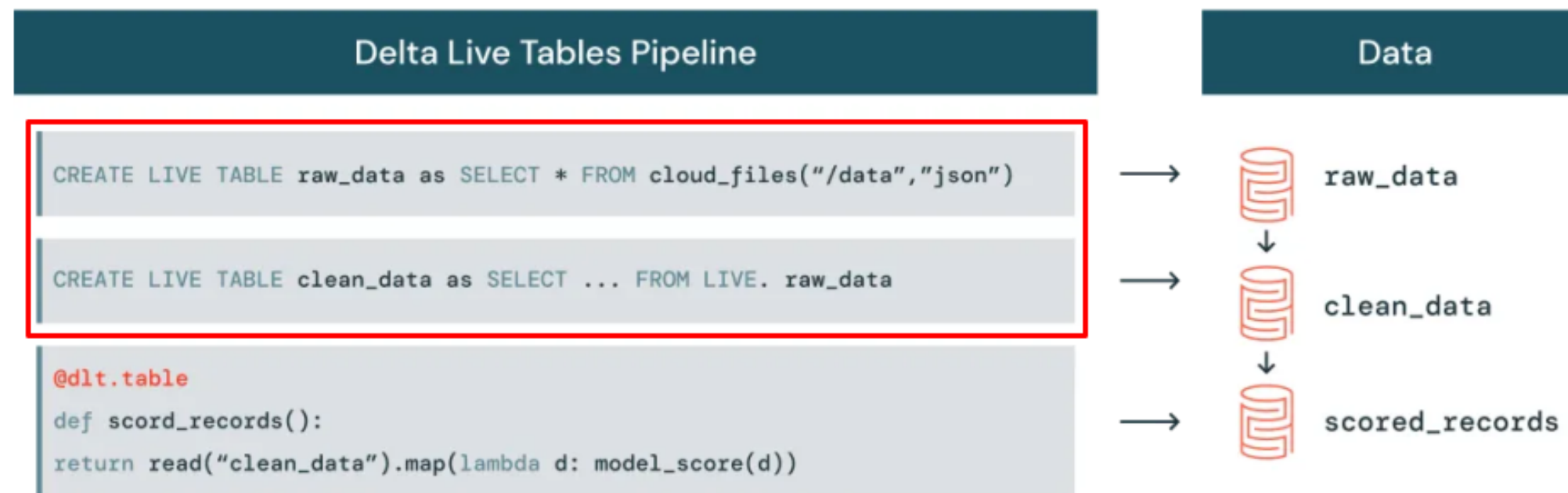
Delta Live Tables

Delta Live Tables understands and coordinates data flow between your queries



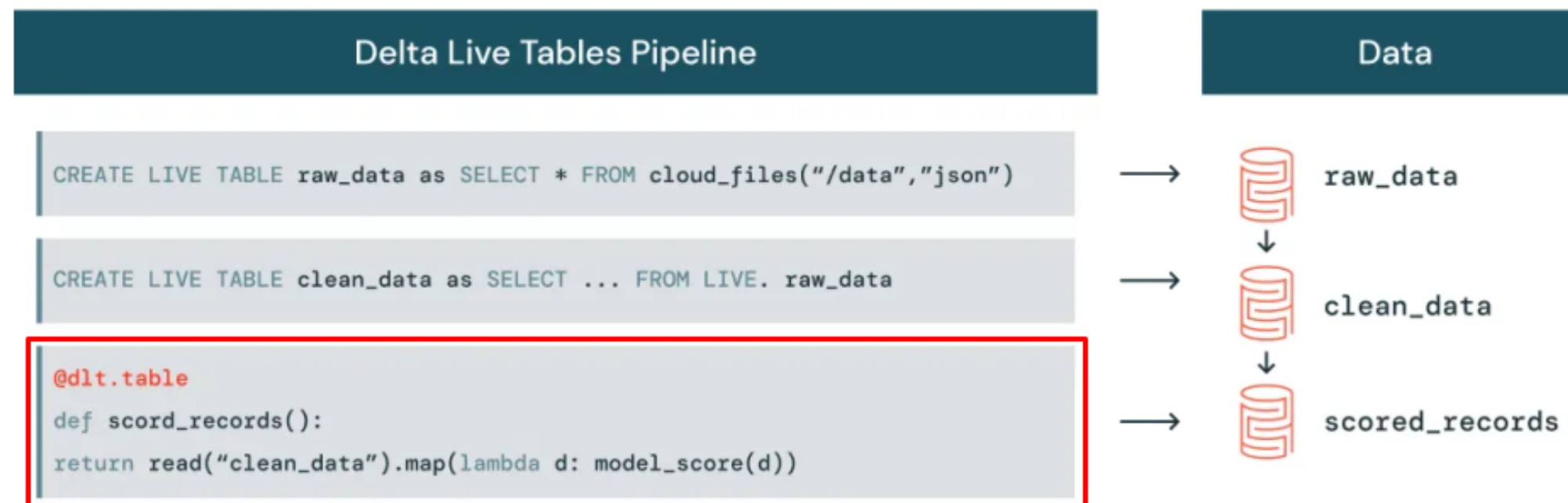
Delta Live Tables

Delta Live Tables understands and coordinates
data flow between your queries



Delta Live Tables

Delta Live Tables understands and coordinates
data flow between your queries



Let's practice!
DATABRICKS CONCEPTS

End-to-end data pipeline example in Databricks

DATABRICKS CONCEPTS



Kevin Barlow
Data Practitioner

Let's practice!

DATABRICKS CONCEPTS